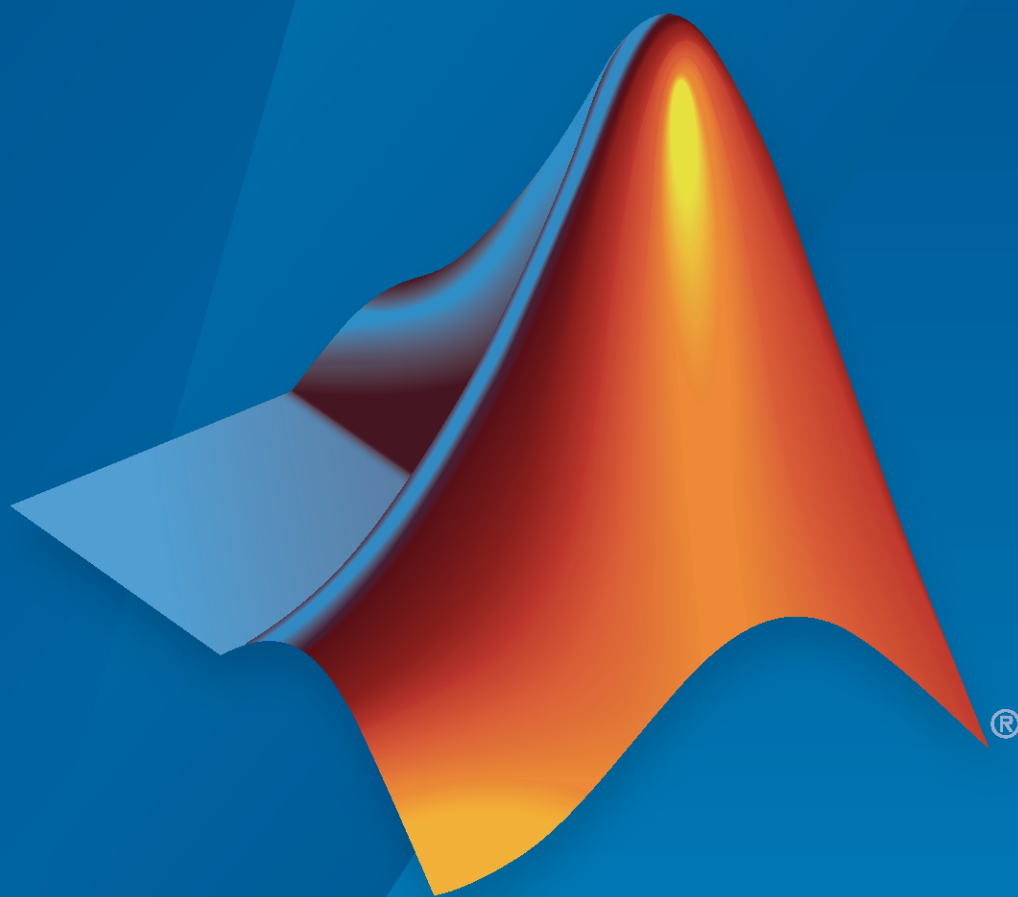


Polyspace® Bug Finder™

Installation Guide



R2023a



How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Polyspace® Bug Finder™ Installation Guide

© COPYRIGHT 2021–2023 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

September 2021	Online Only	Revised for Polyspace Bug Finder Version 3.5, Polyspace Bug Finder Server Version 3.5, and Polyspace Bug Finder Access Version 3.1 (Release 2021b)
March 2022	Online Only	Revised for Polyspace Bug Finder Version 3.6, Polyspace Bug Finder Server Version 3.6, and Polyspace Access Version 4.0 (Release 2022a)
September 2022	Online Only	Revised for Polyspace Bug Finder Version 3.7, Polyspace Bug Finder Server Version 3.7, and Polyspace Access Version 4.1 (Release 2022b)
March 2023	Online Only	Revised for Polyspace Bug Finder Version 3.8, Polyspace Bug Finder Server Version 3.8, and Polyspace Access Version 4.2 (Release 2023a)

Introduction

1	About This Installation Guide	1-2
----------	--	------------

Install Polyspace Bug Finder

2	Install Polyspace Desktop Products	2-2
	Workflow	2-2
	Product Installation	2-3
	Install Polyspace with Other MathWorks Products	2-4
	Install Polyspace Desktop Plugin for Eclipse	2-6
	Install Polyspace Plugin	2-6
	Uninstall Polyspace Plugin	2-7

Install Polyspace Bug Finder Server

3	Install Polyspace Server and Access Products	3-2
	What You Need to Install	3-2
	Product Installation	3-3
	Check Polyspace Installation	3-5
	Install Polyspace with Other MathWorks Products	3-5
	Install Polyspace Server Products at Command Line	3-7
	Prerequisites	3-7
	Installation Steps	3-7
	Install Products for Submitting Polyspace Analysis from Desktops to Remote Server	3-11
	Choose Between Local and Remote Analysis	3-11
	Requirements for Remote Analysis	3-11
	Configure and Start Server	3-14
	Configure Client	3-17
	Offload Polyspace Analysis from Desktop to Server	3-18
	Submit Analysis Jobs from Multiple Releases of Polyspace	3-19

Install Products for Submitting Polyspace Analysis from Desktops to Servers Hosted on AWS	3-21
Prepare Your Installation	3-21
Configure and Deploy Network License Manager Stack	3-22
Configure and Deploy MATLAB Parallel Server and Polyspace Server Stack	3-24
Configure Security Groups	3-26
Configure Client Machines	3-26
Update or Delete Resources	3-28
Install Products for Submitting Polyspace Analysis from Desktops to Servers Hosted on Azure	3-29
Prepare Your Installation	3-29
Configure and Deploy Network License Manager Resources	3-30
Configure and Deploy MATLAB Parallel Server and Polyspace Server Resources	3-32
Configure Client Machines	3-34
Update or Delete Resources	3-36

Install Polyspace Bug Finder Access Web Interface

4

System Requirements for Polyspace Access	4-2
Required Software	4-2
Windows Requirements	4-2
Hardware and Other Requirements	4-2
Storage and Port Configuration	4-4
Storage Configuration	4-4
Network Port Configuration	4-4
Create a Linux Virtual Machine by Using Hyper-V	4-6
Prerequisites	4-6
Create a Virtual Machine	4-6
Start and Configure the Virtual Machine	4-7
Prepare Your Installation	4-9
General Prerequisites	4-10
User Manager Prerequisites	4-11
Issue Tracker Prerequisites	4-11
Polyspace Access Support on WSL	4-12
Configure and Start the Cluster Admin	4-13
Prerequisites	4-13
Unzip Installation Image and Start Cluster Admin Agent	4-13
Choose Between HTTP and HTTPS Configuration for Polyspace Access ..	4-14
Enable HTTPS Between Polyspace Access Services	4-17
Open the Cluster Admin Interface	4-17
Configure User Manager	4-21
General User Manager Settings	4-21
Connect Your Organization LDAP Server to the User Manager	4-24

Configure Issue Tracker	4-29
Configure Jira Software Bug Tracking Tool	4-29
Configure Redmine Bug Tracking Tool	4-31
Add BTT Instance Configured by Using HTTPS	4-31
Configure Polyspace Access App Services	4-33
Start Polyspace Access and Manage Users	4-35
Configure Polyspace Access to Restart Automatically	4-36
Manage Users and Groups	4-36
Update List of Polyspace Access Users and Groups	4-42
Upload Examples and Open Polyspace Access Interface	4-44
Upload Examples	4-44
Open the Polyspace Access Web Interface	4-45
Register Polyspace Desktop User Interface	4-47
Generate a Client Keystore	4-48
Database Backup	4-51
Create Database Backup	4-51
Restore Database from Backup	4-51
Database Backup for Polyspace Access Versions R2020a and Earlier ..	4-54
Create Database Backup	4-54
Restore Database from Backup	4-54
Database Clean Up	4-56
Perform Database Vacuuming	4-56
Delete Project Runs or Entire Projects	4-57
Update or Uninstall Polyspace Access	4-59
Update Polyspace Access	4-59
Uninstall Polyspace Access	4-62

Install Polyspace as You Code

5

Install Polyspace as You Code Using Installer	5-2
Prepare Polyspace as You Code Installation	5-2
Install Polyspace as You Code Interactively	5-4
Install Polyspace as You Code Noninteractively	5-6
Install Polyspace as You Code Extension in Visual Studio	5-8
Interactive Installation	5-8
Command-Line Installation	5-8
Install Polyspace as You Code Extension in Visual Studio Code	5-10
Interactive Installation	5-10
Command-Line Installation	5-11

Install Polyspace as You Code Plugin in Eclipse	5-13
Interactive Installation	5-13
Command-line Installation	5-15
Configure Eclipse for Supported Java Version on Linux	5-16

Manage Polyspace Licenses

6

Configure Polyspace Access License	6-2
Configure NNU License	6-2
Install License Manager	6-5
Manage Named Users for Polyspace Access	6-7
Configure License Borrowing	6-9
Borrow a Polyspace License	6-9
Return Polyspace License	6-10

Update Polyspace

7

Update Polyspace Products	7-2
Install Update on Machine Connected to Internet	7-2
Install Update on Machine Without Internet	7-2
Check Update Installer Log and Update Version	7-3
Migrate Polyspace Projects After Product Upgrade	7-4
Account for Changes in Options	7-4
Account for Improvements in Results	7-5

Introduction

About This Installation Guide

This Installation Guide covers all Polyspace Bug Finder products:

- Polyspace Bug Finder™
- Polyspace Bug Finder Server™
- Polyspace Access™

Depending on how you set up a Bug Finder run, you might be running an analysis from one of these locations:

- **Desktop:** If you are running an analysis and reviewing the results on your desktop, you use Polyspace Bug Finder. To begin installation, see “Install Polyspace Bug Finder”.
- **Server:** If you are running an analysis on a server or reviewing the results from a server run on a web browser, you use:
 - Polyspace Bug Finder Server to run the analysis.
 - Polyspace Access to host the analysis results (for review on a web browser).

To begin installation, see “Install Polyspace Bug Finder Server” and “Install Polyspace Bug Finder Access Web Interface”.

- **IDE:** If you are running an analysis on the current file in your Integration Development Environment (IDE), you use Polyspace as You Code. Polyspace as You Code is a feature available with Polyspace Access. To begin installation, see “Install Polyspace as You Code”.

Install Polyspace Bug Finder

Install Polyspace Desktop Products

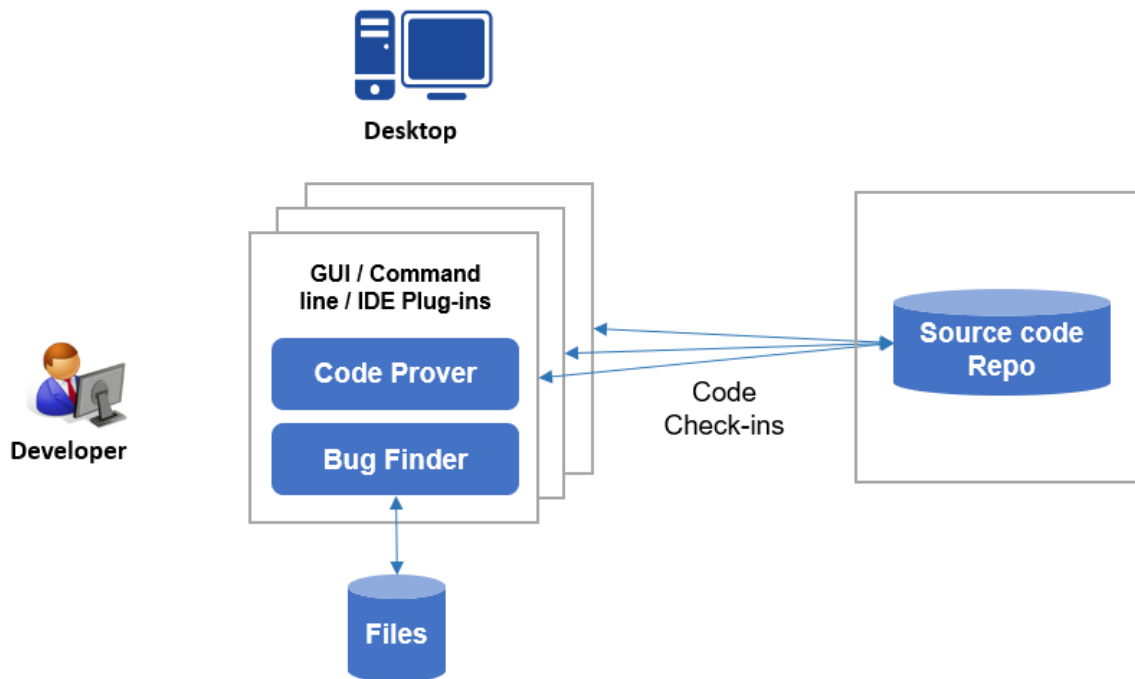
Polyspace checks C/C++ code for bugs, run-time errors, coding standard violations, and other issues by using static analysis. With the desktop products, Polyspace Bug Finder and Polyspace Code Prover, you can perform the checks on individual desktops prior to code submission.

For an overview of all Polyspace products, see “Polyspace Products and Software Development Workflows”.

Workflow

Using the Polyspace desktop products, individual developers can check their code for bugs and run-time errors during development.

For uniform standards across a project or team, all developers in the project or team can use a predefined set of checks. Developers can qualify their code for submission based on these predefined checks. After code submission to a shared repository, a more extensive post-submission analysis can run on a server by using the Polyspace server products.



The workflow consists of these steps:

- Running Polyspace analysis:

During development, individual developers start the analysis from their IDEs by using scripts or from the user interface of the desktop products.

For code generated from:

- Simulink® models, you can start the analysis directly from Simulink after code generation.
- MATLAB® code, you can start the analysis directly in the MATLAB Coder App after code generation.

To save processing power on the developer's desktop, you can offload the analysis to a server. After analysis, the results are downloaded to the desktop for review.

- Reviewing Polyspace results:

After analysis, developers review the results (bugs, run-time errors, coding standard violations, and so on) in the desktop product user interface.

If using Eclipse™ or an IDE based on Eclipse, developers can review the results directly in the IDE.

These steps describe a workflow prior to code submission. After code submission, a build automation tool can run a Polyspace analysis on a server. The tool can also upload the analysis results to a web browser for collaborative review by developers or quality engineers. See “Install Polyspace Server and Access Products” on page 3-2.

Product Installation

For this workflow, you must install the following products on individual desktops.

Polyspace Products to Run Analysis

Install Polyspace Bug Finder or Polyspace Code Prover to run the analysis.

For more efficient analysis, the computer that runs the analysis should have at least four physical cores, with at least 4 GB of memory per core. A Code Prover analysis can be parallelized to use up to four cores. A Bug Finder analysis can be parallelized to use the maximum number of available cores.

Installation

Run the MathWorks® installer. Choose a license for Polyspace desktop products. You can get the installer and license by purchasing the product or requesting a trial. For detailed instructions, see “Installation and Licensing”.

To install the products non-interactively or silently at the command line, see “Install Noninteractively”.

You require Polyspace Bug Finder to install Polyspace Code Prover.

Installation Folder

A default installation folder is used based on your operating system and the release version. During installation, you can change this default folder if needed.

For instance, the default installation folder for release R2023a is listed here.

Operating System	Default Installation Folder
Windows®	C:\Program Files\Polyspace\R2023a
Linux® (most distributions)	/usr/local/Polyspace/R2023a

Operating System	Default Installation Folder
Mac	/Applications/Polyspace/R2023a

Post-Installation Steps

After you install a Polyspace desktop product, you can open the Polyspace user interface or run command-line executables. You can start an analysis in the user interface or from the Windows or Linux command line. Note that all Polyspace commands are available in the folder *polyspaceroot* \polyspace\bin, where *polyspaceroot* is the Polyspace installation folder. To avoid typing the full path to the commands, add this location to the PATH environment variable in your operating system.

To start the analysis from other environments, perform these post-installation steps:

- To run Polyspace from Eclipse or an IDE based on Eclipse, install the Polyspace plugin. See “Install Polyspace Desktop Plugin for Eclipse” on page 2-6.
- To run Polyspace with MATLAB scripts, install MATLAB. Then, perform a one-time setup to link your Polyspace and MATLAB installations. See “Integrate Polyspace with MATLAB and Simulink”.
- To run Polyspace from Simulink, install MATLAB, Simulink, and Embedded Coder®. Then, perform a one-time setup to link your Polyspace and Simulink installations. See “Integrate Polyspace with MATLAB and Simulink”.
- To run Polyspace from the MATLAB Coder App, install MATLAB and Embedded Coder. Then, perform a one-time setup to link your Polyspace and MATLAB installations. See “Integrate Polyspace with MATLAB and Simulink”.
- To offload the analysis to a server, install Polyspace Bug Finder only on your desktop. On the server side, install the Polyspace server products and MATLAB Parallel Server™ to handle analysis jobs from multiple desktops. See “Install Products for Submitting Polyspace Analysis from Desktops to Remote Server” on page 3-11.

Polyspace Products to Review Results

The Polyspace Bug Finder or Polyspace Code Prover installation is sufficient to review the results.

You can review Bug Finder and Code Prover results with only Bug Finder desktop. For instance, if you offload the analysis to a server and only review the downloaded analysis results on your desktop, you require Bug Finder only.

In Eclipse or IDEs based on Eclipse, if you install the Polyspace plugin, you can see the results directly in the IDE.

Install Polyspace with Other MathWorks Products

To install Polyspace with other MathWorks products such as MATLAB, run the MathWorks installer twice.

- In the first run, choose the license that corresponds to the other MathWorks products, such as MATLAB, Simulink, or Embedded Coder.
- In the second run, choose the license that corresponds to the Polyspace products.

In this workflow, products such as MATLAB and Simulink are installed in a different root folder than the Polyspace products. You can link the two installations and use MATLAB scripts to run Polyspace. See “Integrate Polyspace with MATLAB and Simulink”.

If you install the Polyspace desktop and server products, you also have to run the installer twice with separate licenses. The desktop and server products are installed in separate root folders. For instance, in Windows, the default root folders for an R2023a installation are:

- Polyspace desktop products: C:\Program Files\Polyspace\R2023a.

This folder contains executables to run analysis with the products, Polyspace Bug Finder and Polyspace Code Prover.

- Polyspace server products: C:\Program Files\Polyspace Server\R2023a.

This folder contains executables to run analysis with the products, Polyspace Bug Finder Server and Polyspace Code Prover Server.

See Also

More About

- “Run Polyspace Bug Finder on Desktop”
- “Review Polyspace Bug Finder Results in Polyspace User Interface”
- “Install Polyspace Server and Access Products” on page 3-2
- “Update or Uninstall Polyspace Access” on page 4-59

Install Polyspace Desktop Plugin for Eclipse

This topic describes how to install the Polyspace desktop plugin for analysis of complete Eclipse projects. To install the Polyspace as You Code plugin, see “Run Polyspace as You Code in Eclipse and Review Results”.

After you install the Polyspace desktop products, you can proceed to install the Polyspace plugin for Eclipse.

Install Polyspace Plugin

The Polyspace plugin is supported for Eclipse versions 4.20 to 4.23. You can install the Polyspace plugin only after you:

- Install and set up Eclipse Integrated Development Environment (IDE). For more information, see the Eclipse documentation at www.eclipse.org.
- Install a Java® version between 7 and 15. See available Java versions here. You must install a compatible Java version to use the Polyspace plugin.

If you run into issues because of incompatible Java versions, see “Troubleshoot Java Incompatibility in Polyspace Plugin for Eclipse”.

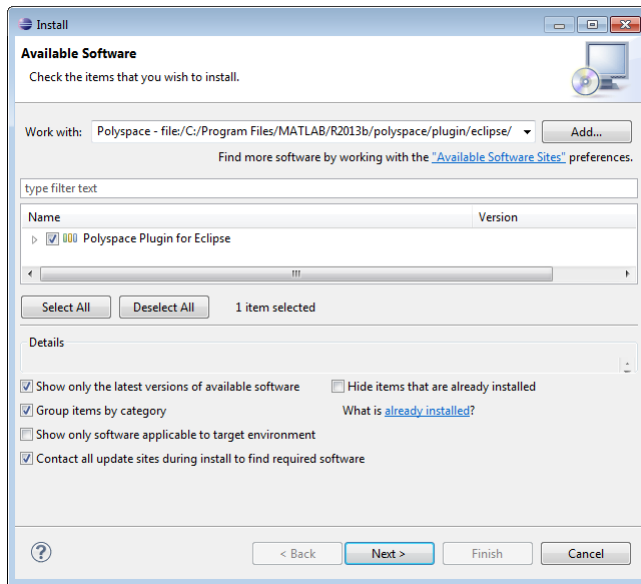
- Uninstall any previous Polyspace plugins. For more information, see “Uninstall Polyspace Plugin” on page 2-7.

To install the Polyspace plugin:

- 1 From the Eclipse editor, select **Help > Install New Software**. The Install wizard opens, displaying the Available Software page.
- 2 Click **Add** to open the Add Repository dialog box.
- 3 In the **Name** field, specify a name for your Polyspace site, for example, `Polyspace_Eclipse_Plugin`.
- 4 Click **Local**, to open the Browse for Folder dialog box.
- 5 Navigate to the `polyspaceroot\polyspace\plugin\eclipse` folder. Then click **OK**.

`polyspaceroot` is the installation folder for the Polyspace product.

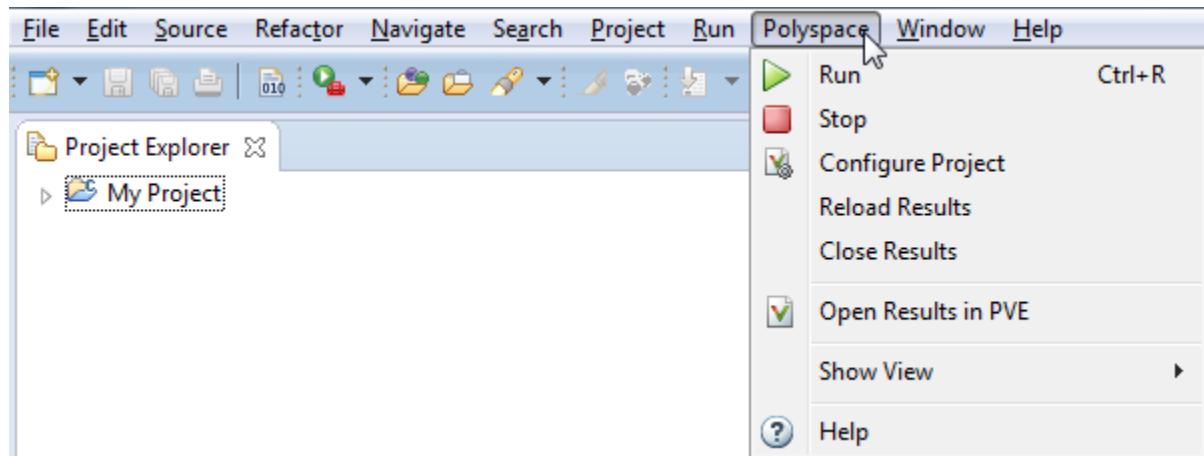
- 6 Click **OK** to close the Add Repository dialog box.
- 7 On the Available Software page, select **Polyspace Plugin for Eclipse**.



- 8 Click **Next**.
- 9 On the Install Details page, click **Next**.
- 10 On the Review Licenses page, review and accept the license agreement. Then click **Finish**.
- 11 Restart Eclipse. You might get this prompt: Select the folder where Polyspace Bug Finder was installed. Enter the path to your Polyspace installation, for instance, C:\Program Files\Polyspace\R2023a.

Once you install the plugin, in the Eclipse editor, you'll see:

- A **Polyspace** menu
- A **Polyspace Run - Bug Finder, Results List - Bug Finder**, and **Result Details** view.



Uninstall Polyspace Plugin

Before installing a new Polyspace plugin, you must uninstall any previous Polyspace plugins:

- 1 In Eclipse, select **Help > About Eclipse**.

- 2 Select **Installation Details**.
- 3 Select the Polyspace plugin and select **Uninstall**.

Follow the uninstall wizard to remove the Polyspace plugin. You must restart Eclipse for changes to take effect.

See Also

More About

- “Run Polyspace Analysis on Eclipse Projects”

Install Polyspace Bug Finder Server

Install Polyspace Server and Access Products

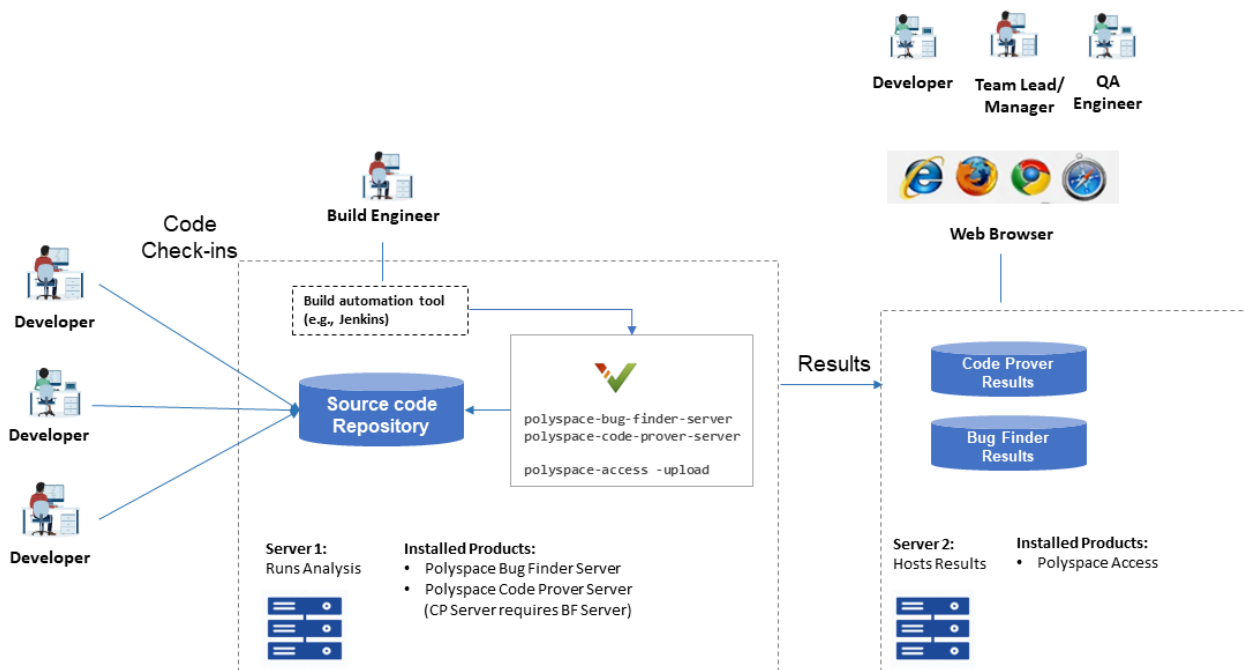
Polyspace checks C/C++ code for bugs, run-time errors, coding standard violations, and other issues by using static analysis. You can run a Polyspace analysis on server class machines and review the analysis results on a web browser with these products:

- Polyspace Bug Finder Server for finding bugs and coding rule violations
- Polyspace Code Prover Server for exhaustive code verification
- Polyspace Access for web browser based review of analysis results (from either Polyspace Bug Finder Server or Polyspace Code Prover Server)

By using these products with a build automation tool, you can incorporate a Polyspace analysis into continuous integration systems.

What You Need to Install

In a continuous integration process, developers submit code to a shared repository. An automated build system builds and tests each code submission at regular intervals or based on predefined triggers, and then integrates the code. You can run a Polyspace analysis as part of this process.



Note:

- Depending on the specifications, the same computer can serve as both Server 1 and Server 2.
- Though a server hosts the components for Polyspace web interface, each reviewer requires a PolyspaceAccess license to login to the interface.

The workflow consists of three steps:

1 Triggering Polyspace analysis:

When developers submit their code, a build automation tool such as Jenkins triggers the Polyspace analysis.

2 Running Polyspace analysis:

The analysis is run by using the products, Polyspace Bug Finder Server or Polyspace Code Prover Server. The results are uploaded to a server that hosts the Polyspace Access web interface. The same computer can act as the server that runs the analysis and hosts the uploaded results.

3 Reviewing analysis results:

To see the uploaded results, each reviewer must have a Polyspace Access license to login to the Polyspace Access web interface.

These steps describe a workflow after code submission. Before code submission, individual developers can:

- Run a Polyspace analysis and review the results on their desktops. See “Install Polyspace Desktop Products” on page 2-2.
- Offload a Polyspace analysis from their desktops to a remote server and review the downloaded results on their desktops. See “Install Products for Submitting Polyspace Analysis from Desktops to Remote Server” on page 3-11.

For an overview of all Polyspace products, see “Polyspace Products and Software Development Workflows”.

Product Installation

For this workflow, you must install the following products on the computer or computers that act as the server.

Polyspace Products to Run Analysis

Install Polyspace Bug Finder Server or Polyspace Code Prover Server to run the analysis and upload results to a web interface for review.

Installation

- 1** Run the MathWorks installer. Choose a license for Polyspace server products. You can get the installer and license by purchasing the product or requesting a trial. For detailed instructions, see “Installation and Licensing”.

Along with the Polyspace Server products, install the network license manager to manage licenses (unless it is already installed). Configure the license manager to start automatically on reboot. For instance, in Windows, you can configure the license manager to start as a service.

- 2** At the end of the installation, start the network license manager. See “Start Network License Manager”.

To install the products non-interactively or silently at the command line, see Install Noninteractively.

You require Polyspace Bug Finder Server to install Polyspace Code Prover Server. For more efficient analysis, the computer that runs the analysis should have at least four physical cores, with at least 4 GB of memory per core. Beyond four cores, a Code Prover analysis cannot be further parallelized. A Bug Finder analysis can use the maximum number of available cores.

Installation Folder

A default installation folder is used based on your operating system and the release version. During installation, you can change this default folder if needed.

For instance, the default installation folder for release R2023a is listed here.

Operating System	Default Installation Folder
Windows	C:\Program Files\Polyspace Server\R2023a
Linux (most distributions)	/usr/local/Polyspace_Server/R2023a
Mac	/Applications/Polyspace_Server/R2023a

Note that all Polyspace commands are available in the folder *polyspaceroot*\polyspace\bin, where *polyspaceroot* is the Polyspace Server installation folder. To avoid typing the full path to the commands, add this location to the PATH environment variable in your operating system.

Polyspace Products to Review Results

Install the components to host the web interface of Polyspace Access.

Each reviewer must have a Polyspace Access license to view the results on the web interface.

Installation

See “Install Polyspace Access for Web Reviews”.

Products to Schedule Polyspace Analysis

In a continuous integration workflow, a build automation tool schedules the Polyspace analysis as part of a build process. The tool is not needed to run the Polyspace analysis or review the results but only to integrate the analysis into an overall software build and test workflow.

For instance, Jenkins is a popular open source tool that can execute build scripts. For scripting convenience, a Polyspace plugin, available in Jenkins, provides a build environment for automation of Polyspace static analysis and simplifies post-analysis e-mail notification with Polyspace results.

Installation

Install a tool for continuous integration or build automation.

For instance, to install Jenkins, see the Jenkins website. In the Jenkins interface, select **Manage Jenkins** on the left. Select **Manage Plugin**. Search for the Polyspace plugin, and then download and install the plugin.

Release Compatibility

When you upload Polyspace analysis results to Polyspace Access, you must use a version of Polyspace Server that is the same as or older than the version of Polyspace Access. For instance, if you upload a result to an **R2023a** version of Polyspace Access, you must use an **R2023a** or earlier version of Polyspace Server.

More specifically, when you upload an analysis result to Polyspace Access:

- The analysis must have run using the same or an older version of Polyspace Server.
- The upload command must come from the same an older version of Polyspace Server.

The same considerations apply to:

- Exporting results and report generation from Polyspace Access.
- Uploading results to Polyspace Access from a Polyspace desktop product.

Check Polyspace Installation

To check if the installation of Polyspace Bug Finder Server was successful:

- 1 Open a command window. Navigate to *polyspaceserverroot*\polyspace\bin. Here, *polyspaceserverroot* is the Polyspace Bug Finder Server installation folder, for instance, C:\Program Files\Polyspace Server\R2023a.
- 2 Enter:

```
polyspace-bug-finder-server -help
```

You should see the list of options allowed for a Bug Finder analysis. To check if the installation of Polyspace Code Prover Server was successful, you can perform the same steps for Polyspace Code Prover Server. Replace `polyspace-bug-finder-server` with `polyspace-code-prover-server`.

To check if the Polyspace Access web interface is set up for upload:

- 1 Navigate again to *polyspaceserverroot*\polyspace\bin.
- 2 Enter:

```
polyspace-access -host hostName -port portNumber -create-project testProject
```

Here, *hostName* is the name of the server hosting the Polyspace Access web server. For a locally hosted server, use `localhost`. *portNumber* is the optional port number of the server. If you omit the port number, 9443 is used.

If the connection is successful, a project called `testProject` should be created in the Polyspace Access web interface.

- 3 Open this URL in a web browser:

```
https://hostName:portNumber/metrics/index.html
```

Here, *hostName* and *portNumber* are the host name and port number from the previous step.

In the **Project Explorer** pane on the Polyspace Access web interface, you should see the newly created project `testProject`.

Install Polyspace with Other MathWorks Products

To install Polyspace with other MathWorks products such as MATLAB, run the MathWorks installer twice.

- In the first run, choose the license that corresponds to the other MathWorks products, such as MATLAB, Simulink, or Embedded Coder.

- In the second run, choose the license that corresponds to the Polyspace products.

In this workflow, products such as MATLAB and Simulink are installed in a different root folder than the Polyspace products. You can link the two installations and use MATLAB scripts to run Polyspace. See “Integrate Polyspace Server Products with MATLAB”.

If you install the Polyspace desktop and server products, you also have to run the installer twice with separate licenses. The desktop and server products are installed in separate root folders. For instance, in Windows, the default root folders for an R2023a installation are:

- Polyspace desktop products: C:\Program Files\Polyspace\R2023a.

This folder contains executables to run analysis with Polyspace Bug Finder and Polyspace Code Prover.

- Polyspace server products: C:\Program Files\Polyspace Server\R2023a.

This folder contains executables to run analysis with Polyspace Bug Finder Server and Polyspace Code Prover Server.

See Also

More About

- “Run Polyspace Bug Finder on Server and Upload Results to Web Interface”
- “Send Email Notifications with Polyspace Bug Finder Server Results”
- “Install Polyspace Desktop Products” on page 2-2
- “Update or Uninstall Polyspace Access” on page 4-59

Install Polyspace Server Products at Command Line

This topic describes how to perform an installation of the Polyspace Server products entirely at the command line.

The Polyspace Server products support automated Polyspace runs on a Continuous Integration (CI) server. You might be accessing this server through a terminal or using cloud computing platforms such as Azure® or AWS®. In these situations, you might not be able to install Polyspace Bug Finder Server or Polyspace Code Prover Server using a graphical installer. Use the steps below to perform a command-line installation.

Prerequisites

Before beginning installation from the command line, note the following:

- You must have a license file (.lic file) from MathWorks and a File Installation Key or FIK for this license. To obtain the license file and FIK, activate Polyspace in License Center.
- Some of the initial steps of the installation occur in a graphical user interface. If you cannot open a graphical interface on your target machine, you must have access to a machine where you can open a graphical user interface and then transfer files from this machine to the target machine.
- On the target machine, you must have the Linux Standard Base or `lsb` package installed. On most standard installations, this package is already installed. The machine must also have `unzip` or an equivalent package to extract files from a .zip file.

Installation Steps

To run the Polyspace Server products, you have to install the products and the network license manager for managing licenses.

Create Installer

You can download a light installer from the web and create the full installer required for command-line installation (or obtain the ISO image of the full installer directly from https://www.mathworks.com/downloads/web_downloads/download_iso).

To create the full installer from the light installer download from the web:

- 1** Download a light installer (.zip file) from the web on a machine, where you can open a graphical user interface.

For instance, you can download the installer from <https://www.mathworks.com/downloads>. The machine where you download the installer can have a different operating system from the machine where you will eventually install the products. For instance, you can download a light installer for Linux on a Windows machine, create the full installer and then transfer the full installer to the target Linux machine.

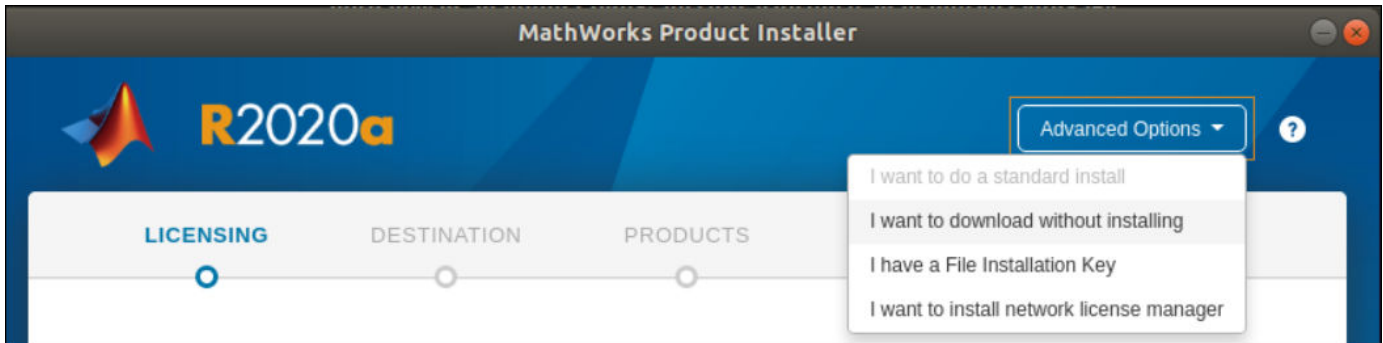
- 2** Unzip and run the installer to download additional product files so that you have a full installer.

For instance, if you downloaded the light installer to a Linux machine:

- a** Extract files from the zipped installer.
- b** In a terminal, `cd` to the extracted folder and enter:

```
./install
```

- c In the installer interface, after signing in to your MathWorks Account and accepting the License Agreement, from the **Advanced Options** menu, select **I want to download without installing**.



- d Follow the prompts to complete the download.

Your destination folder, for instance, one that ends with `MathWorks/Rxxxx`, contains all the files required for full installation.

At this point, the light installer is no longer required.

Install Product

Once you have the full installer, you are ready to install the Polyspace Server products on any machine.

- 1 Transfer the files that you downloaded in the previous step to the machine without a graphical user interface.
- 2 On this machine, run the full installer with an input file as argument. In this file, you specify the various installation specifics such as installation folder, File Installation Key or FIK, and path to license file.

For instance, if you are installing on a Linux machine:

- a Make a writable copy of the file `installer_input.txt` that is provided with the installer files.
- b Read the instructions in the file and enter the required information such as destination folder, license agreement, File Installation Key or FIK, and product selection. Unlike a desktop product installation, you also need to enter the path to the license file that you obtained from MathWorks.
- c In a terminal, `cd` to the folder containing the installer files and enter the following:

```
./install -inputFile installer_input_copy.txt
```

where `installer_input_copy.txt` is the writable copy of the file `installer_input.txt` that you made earlier (the above command assumes that the copy is saved in the same location as the original file).

Depending on where you install, you might require superuser privileges for the installation. If so, prepend `sudo` to the installation command.

If you create the full installer in Windows and then transfer to a Linux machine for installation, you might see this error message during command-line installation: `Command not found`. Extract the contents of the zip file `matlab_Rxxxx_glnxa64.zip` so that necessary files are available for command-line installation.

After installation, `cd` to the subfolder `polyspace/bin` in the installation folder and enter:

```
./polyspace-bug-finder-server -help
```

If you do not see this subfolder or run into other errors, the installation might not have completed successfully. Check the installation log. You specified this log in the input file that you provided with the `-inputFile` option. Typically, the file path is `/tmp/mathworks_username.log`.

To run Polyspace Bug Finder Server or Polyspace Code Prover Server on source files, you must also install and start the network license manager that handles checking out of licenses. Otherwise, if you try to run `polyspace-bug-finder-server` (or `polyspace-code-prover-server`) on a C source file:

```
./polyspace-bug-finder-server -sources aFile.c
```

you see a license checkout error.

Install Network License Manager

The network license manager controls checkout of licenses. To allow a Polyspace Server installation to communicate with a network license manager, you have to specify the host name and host ID of the server that runs the network license manager in your Polyspace Server installation.

To install the network license manager:

- 1 Download the network license manager in zipped format from https://www.mathworks.com/support/install/license_manager_files.html.

If you want to start the network license manager on a machine without a graphical user interface, transfer the zipped file to that machine.

- 2 Extract the network license manager files to a folder.

Modify these license files to allow communication between the Polyspace Server installation and the network license manager:

- License file (`.lic` file) provided by MathWorks: Add the following lines at the beginning of the license file:

```
SERVER hostname hostid port_number
DAEMON MLM path_to_mlm
```

where:

- *hostname* is the host name of the server hosting the network license manager. To find the host name, in a terminal, enter:
`hostname`
- *hostid* is the MAC address of the server. You provided this MAC address to obtain the license file (and this address should be already in a comment in the license file).

- *port_number* is the port number on the server used for communication.
- *path_to_mlm* is the path to the MLM binary provided with the network license manager. On Linux, the binary is typically located in the subfolder `etc/glnxa64` in the network license manager folder. Example specification: `/usr/local/license_manager/etc/glnxa64/MLM`.
- `network.lic` file in the subfolder `licenses` in the Polyspace Server installation folder: Modify the line starting with `SERVER` to exactly replicate the `SERVER` line in the previous license file:

```
SERVER hostname hostid port_number
```

To start the network license manager, on a terminal, `cd` to the subfolder `etc/glnxa64` in the network license manager folder and enter:

```
./lmgrd -c path_to_license
```

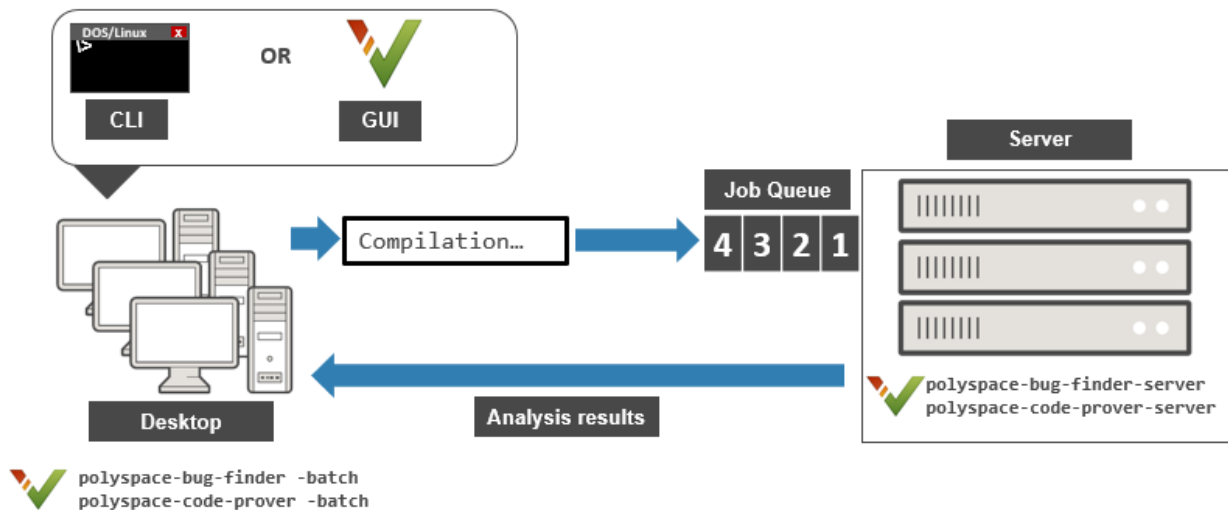
where *path_to_license* is the path to the license file (`.lic` file) provided by MathWorks. You can use the option `-l` to pipe the output to a log file.

You can also configure the network license manager to start automatically. It is recommended that you do not start the license manager as a root user since it is a security risk to run any program that does not require root permissions and the network license manager does not require root permissions. For instance, you can create a dedicated user with restricted permissions and start the network license manager as this user. See “Start Network License Manager”.

Now, you should be able to run the `polyspace-bug-finder-server` (or `polyspace-code-prover-server`) command on a source file without errors. See “Run Polyspace Bug Finder on Server and Upload Results to Web Interface” or “Run Polyspace Code Prover on Server and Upload Results to Web Interface” (Polyspace Code Prover).

Install Products for Submitting Polyspace Analysis from Desktops to Remote Server

You can perform a Polyspace analysis locally on your desktop or offload the analysis to one or more dedicated remote servers. This topic shows how to set up the dispatch of Polyspace analysis from desktop clients to remote servers. Once configured, you can send the Polyspace analysis to a remote server and view the downloaded results on your desktop.



Choose Between Local and Remote Analysis

To determine when to use local or remote analysis, use the rules listed in this table.

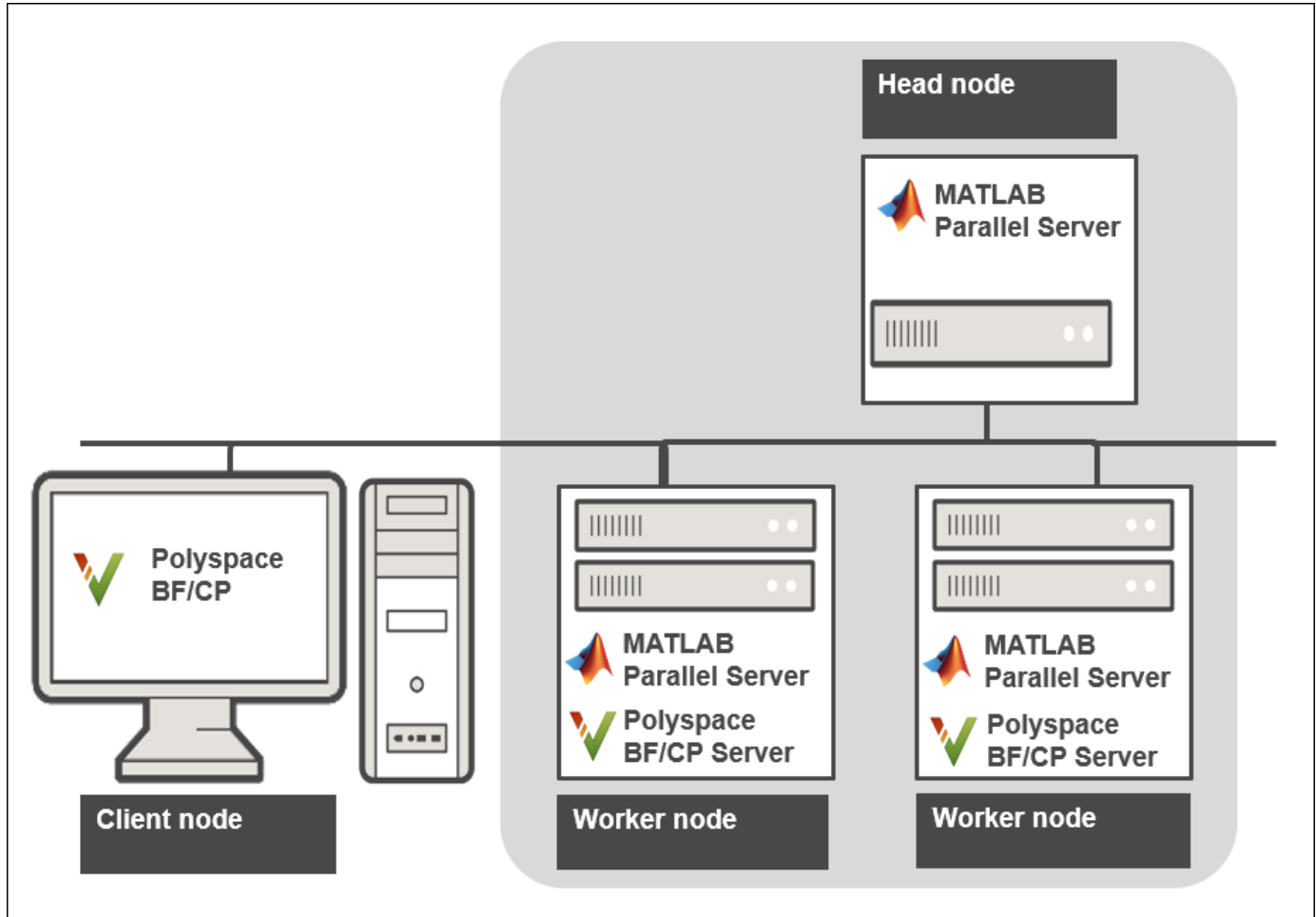
Type	When to Use
Remote	Source files are large and execution time of analysis is lengthy. Typically, a Code Prover analysis takes significantly longer than a Bug Finder analysis and benefits from running on a dedicated server.
Local	Source files are small and execution time of analysis is short.

Requirements for Remote Analysis

A typical distributed network for running remote analysis consists of these parts:

- **Client nodes:** On the client node, you configure your Polyspace project or scripts, and then submit a job that runs Polyspace.
- **Head node:** The head node distributes the submitted jobs to worker nodes.
- **Worker node(s):** The Polyspace analysis runs on a worker node.

In this workflow, you install the product MATLAB Parallel Server to manage submissions from multiple clients. An analysis job is created for each submission and placed in a queue. As soon as a worker node is available, the next analysis job from the queue is run on the worker.



This table lists the product requirements for remote analysis.

Location	Requirements	Installation
Client node	Polyspace Bug Finder A Polyspace Bug Finder license is sufficient to trigger a Bug Finder or Code Prover analysis on the server and review the downloaded analysis results.	Run the MathWorks installer on the client desktops. Choose a license for Polyspace desktop products. For detailed instructions, see "Installation and Licensing".

Location	Requirements	Installation
Head node	MATLAB Parallel Server (earlier called MATLAB Distributed Computing Server)	Run the MathWorks installer on the server(s). Choose a license for MATLAB Parallel Server installation. For detailed instructions, see “Install and Configure MATLAB Parallel Server for MATLAB Job Scheduler and Network License Manager” (MATLAB Parallel Server).
Worker nodes	<ul style="list-style-type: none"> • MATLAB Parallel Server (earlier called MATLAB Distributed Computing Server) • Polyspace Bug Finder Server • Polyspace Code Prover Server (if you choose to run Code Prover) 	<p>To install:</p> <ul style="list-style-type: none"> • MATLAB Parallel Server, run the MathWorks installer on the server(s). Choose a license for MATLAB Parallel Server installation. • Polyspace Bug Finder Server and/or Polyspace Code Prover Server, run the MathWorks installer. Choose a license for Polyspace server products.

In the simplest remote analysis configuration, the same computer can serve as the head node and worker node.

Selecting Number of Workers

To select the number of worker nodes to allocate, consider the following:

- Polyspace does not distribute a single analysis over multiple workers and executes one analysis entirely on one worker node. Do not expect an individual analysis to speed up if you use multiple workers (unless you run a file by file verification in Code Prover).

Only if the head node processes more than one analysis job at the same time can you benefit from multiple workers. This scenario is likely if multiple clients submit analysis jobs within a short time window.

- Each worker node must have a license available for executing the Polyspace analysis. Otherwise, a worker node might wait for a license to be released and you might not benefit from multiple workers.

Make sure that you do not allocate more workers than the number of licenses available.

Using MATLAB Parallel Server for distributing Polyspace jobs is different from other uses of MATLAB Parallel Server where a single job can be parallelized over multiple workers.

Configure and Start Server

On the computers that act as the worker nodes of the server, you install MATLAB Parallel Server and the Polyspace server products in two separate folders. The MATLAB Parallel Server installation must know where the Polyspace server products are located so that it can route the Polyspace analysis. To link the two installations, specify the paths to the root folder of the Polyspace server products in your MATLAB Parallel Server installations.

Then configure and start MATLAB Parallel Server (the `mjs` service) on all computers that act as the head node and worker nodes.

Configure `mjs` Service Settings

Before starting services, you must configure the `mjs` service settings.

- 1 Navigate to `matlabroot\toolbox\parallel\bin`, where `matlabroot` is the MATLAB Parallel Server installation folder, for instance, `C:\Program Files\MATLAB\R2023a`.
- 2 Modify the file `mjs_def.bat` (Windows) or `mjs_def.sh` (Linux). To edit and save the file, you have to open your editor in administrator mode.

Read the instructions in the file and uncomment the lines as needed. At a minimum, you might have to uncomment these lines:

- Hostname:

```
REM set HOSTNAME=%strHostname%.%strDomain%
```

in Windows or

```
#HOSTNAME=`hostname -f`
```

in Linux. Explicitly specify your computer host name.

- Security level:

```
REM set SECURITY_LEVEL=
```

in Windows or

```
#SECURITY_LEVEL=""
```

in Linux. Explicitly specify a security level.

If you do this step later, make sure to first stop the `mjs` services with:

```
mjs stop -clean
```

Then make the modifications and later restart the services with:

```
mjs restart -clean
```

Otherwise, you might see an error later when starting the job scheduler.

Specify Polyspace Installation Paths

When you offload an analysis using a Polyspace desktop product installation, the server must run the analysis using a Polyspace server product installation from the same release. For instance, if you offload an analysis from an R2023a desktop product, the analysis must run using the R2023a server

product. To ensure that the correct Polyspace server product is used, you must specify the installation paths of the Polyspace server products in your MATLAB Parallel Server installations.

To specify the Polyspace installation paths:

- 1 Navigate to *matlabroot*\toolbox\parallel\bin\. Here, *matlabroot* is the MATLAB installation folder, for instance, C:\Program Files\MATLAB\R2023a.
- 2 Uncomment and modify the following line in the file *mjs_polyspace.conf*. To edit and save the file, you have to open your editor in administrator mode.

```
POLYSPACE_SERVER_ROOT=polyspaceserverroot
```

Here, *polyspaceserverroot* is the installation path of the server products, for instance:

```
C:\Program Files\Polyspace Server\R2023a
```

If you use multiple releases of Polyspace desktop and server products, the MATLAB Parallel Server release must be the later one. For instance, if you offload analysis jobs using both R2019a and R2019b Polyspace desktop and server products, the MATLAB Parallel Server installation must be an R2019b one. See also “Submit Analysis Jobs from Multiple Releases of Polyspace” on page 3-19.

Start mjs Service and Assign as Head Node or Worker Node

To configure a server with multiple workers, start the service that runs a job scheduler (the *mjs* service) on the computer that acts as the head node and all computers that act as worker nodes. In the simplest configuration, the same computer can act as the head node and a worker node.

To set up a cluster with one head node and several workers, on the computer that acts as the head node:

- 1 Open the **Admin Center** window. Navigate to *matlabroot*\toolbox\parallel\bin\ and execute the file *admincenter.bat* (Windows) or *admincenter.sh* (Linux). Here, *matlabroot* is the MATLAB installation folder, for instance, C:\Program Files\MATLAB\R2023a.

The screenshot shows the Admin Center interface with three main sections:

- Hosts:** A table with columns: Hostname, Reachable, Cores, Status, Up Since, MATLAB Job Scheduler Name, and Workers Count. It lists node1 (6 cores, running) and node2 (2 cores, running).
- MATLAB Job Scheduler:** A table with columns: Name, Hostname, Status, Up Since, and Workers. It shows jobmanager on node1 (running, 4 workers).
- Workers:** A table with columns: Name, Hostname, Status, Up Since, Connection, and Hostname. It lists worker1-4 on node1 and node2, all with 'idle' status and 'connected' to jobmanager.

Callouts in the image point to the 'jobmanager' entry in the MATLAB Job Scheduler section as the 'Head Node' and the 'worker1-4' entries in the Workers section as 'Worker Nodes'.

- 2 In the **Hosts** section, add the host names of all computers that you want to use as head and worker nodes of the cluster. Start the `mjs` service.

The service uses the settings specified in the file `mjs_def.bat` (Windows) or `mjs_def.sh` (Linux).

- 3 Right-click each host. Select either **Start MJS** (head node) or **Start Workers** (worker nodes).

The hosts appear in the **MATLAB Job Scheduler** or **Workers** section. In each section, select the host and click **Start** to start the MATLAB Job Scheduler or the workers.

Selecting a computer as host starts the `mjs` service on that computer. You must have permission to start services on other computers in the network. For instance, on Windows, you must be in the Administrators group for other computers where you want to start the `mjs` service. Otherwise, you have to start the `mjs` services individually on each computer that acts as a worker.

For more details and command-line workflows, see:

- “Install and Configure MATLAB Parallel Server for MATLAB Job Scheduler and Network License Manager” (MATLAB Parallel Server)
- `mjs` command for starting services, and `nodestatus` command for checking status of nodes (for instance, to check if the node can handle submissions from multiple releases).

For an example of command-line workflows to start the `mjs` services, see “Send Bug Finder Analysis from Desktop to Locally Hosted Server”.

Configure Client

Configure the client node so that it can communicate with the computer that serves as the head node of the MATLAB Parallel Server cluster. For scheduling jobs, you can use the job scheduler that comes with MATLAB Parallel Server (MATLAB Job Scheduler). If you were already using a third-party scheduler such as HPC Server or PBS Professional, you can continue to use that scheduler.

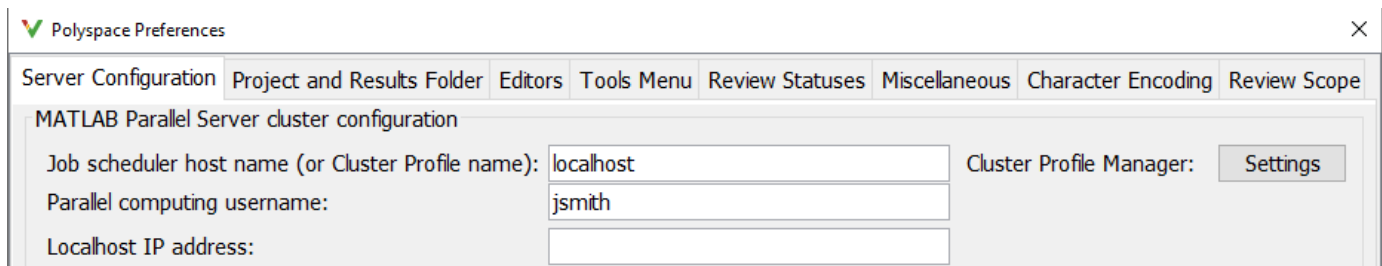
Use MATLAB Job Scheduler

Configure the client node in the user interface of the Polyspace desktop products:

- 1 Select **Tools > Preferences**.
- 2 Click the **Server Configuration** tab. Under **MATLAB Parallel Server cluster configuration**:
 - a In the **Job scheduler host name** field, specify the computer for the head node of the cluster. This computer hosts the MATLAB job scheduler.

If the port used on the computer hosting the MATLAB job scheduler is different from 27350, enter the port name explicitly with the notation *hostName:portNumber*.

- b Due to the network setting, the job scheduler may be unable to connect back to your local computer. If so, enter the IP address of the client computer in the **Localhost IP address** field.



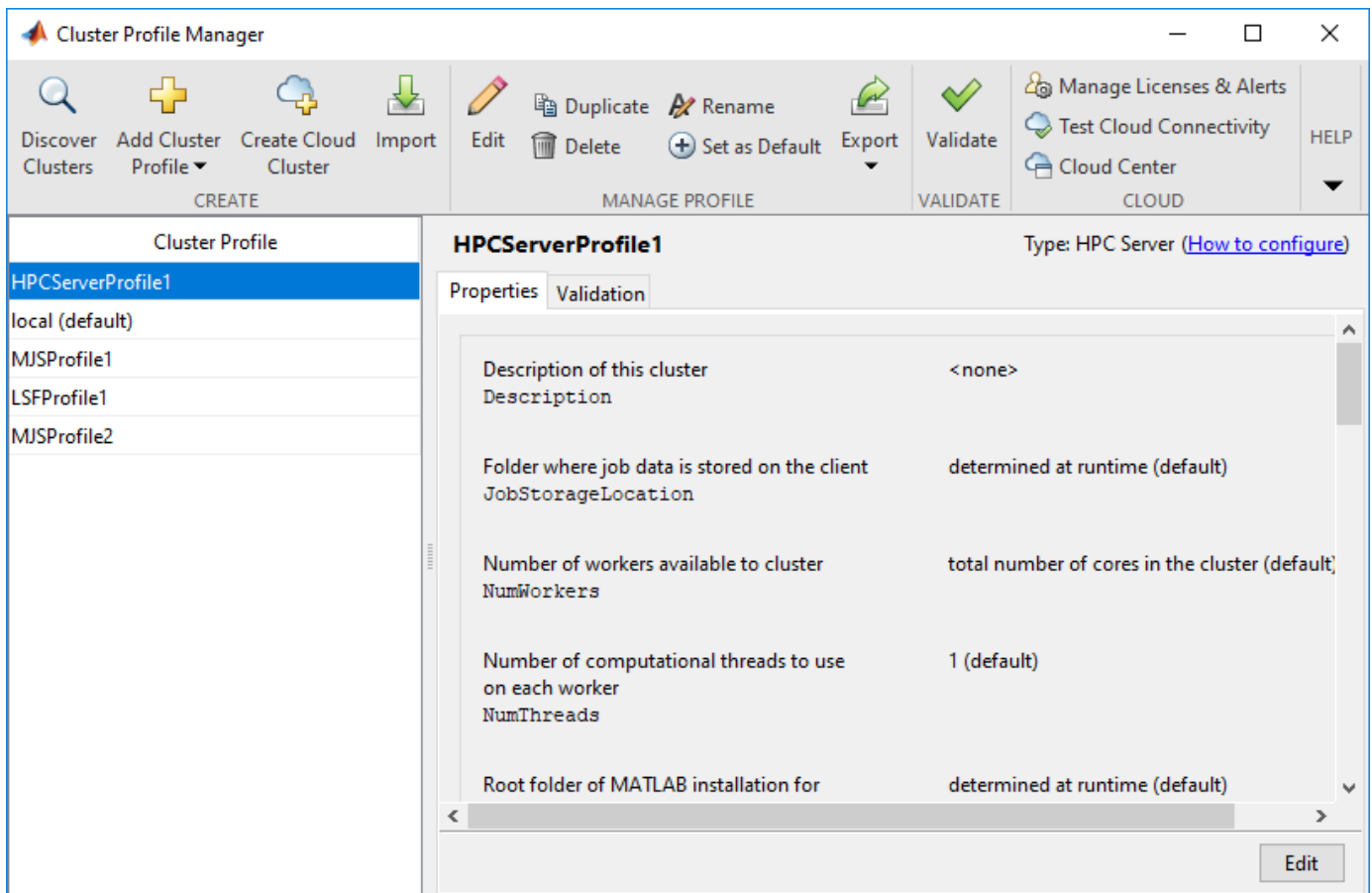
Create a Cluster Profile for Job Scheduling

Cluster profiles let you define certain properties for your cluster, such as number of threads per worker, timeouts, callbacks, and so on. For more information on cluster profiles, see “Discover Clusters and Use Cluster Profiles” (Parallel Computing Toolbox).

For instance, instead of the job scheduler provided with MATLAB Parallel Server, you can use a third party scheduler such as HPC Server or PBS Professional. You can create a cluster profile to use the third-party scheduler. You can also create a cluster profile that uses the MATLAB job scheduler but with nondefault properties. For instance, if you start the job manager with a certificate using the `-certificate` flag, you can create a cluster profile that uses this job manager.

To create and use a cluster profile:

- 1 In the Polyspace user interface, select **Tools > Preferences**.
- 2 In the Polyspace Preferences window, click the **Settings** button beside **Cluster Profile Manager**.
- 3 In the Cluster Profile Manager window, click **Add Cluster Profile**. Select a third-party scheduler. You see a default cluster profile using this scheduler.



- 4 Specify the properties of the cluster profile.
 - a Click **Edit**. Modify the default properties as needed.
 - b Click **Validate**. At a minimum, the cluster connection test and the job test must complete.
- 5 In the Polyspace Preferences window, for **Job scheduler host name**, specify the cluster profile name that you just created (for instance, HPCServerProfile1 in the example above).

Offload Polyspace Analysis from Desktop to Server

Once the configuration is over, you can offload an analysis from a Polyspace desktop product installation to a remote server. You can do one of the following:

- Start a remote analysis from the user interface of the Polyspace desktop products.

See “Send Polyspace Analysis from Desktop to Remote Servers”.

- Start a remote analysis with Windows or Linux scripts.

See “Send Polyspace Analysis from Desktop to Remote Servers Using Scripts”. In the simplest configuration, the same computer can be used as a client and server. For a simple tutorial that uses this configuration and walks through all the steps for offloading a Polyspace analysis, see “Send Bug Finder Analysis from Desktop to Locally Hosted Server”.

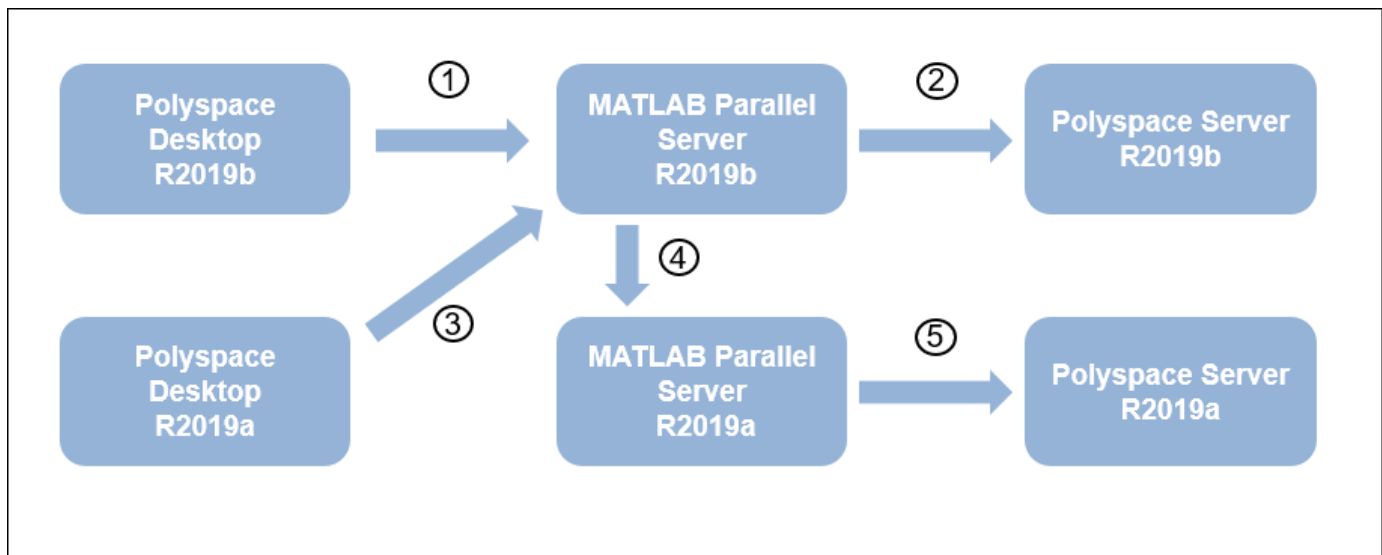
- Start a remote analysis with MATLAB scripts.

See “Run Analysis on Server”.

Submit Analysis Jobs from Multiple Releases of Polyspace

If you upgrade to a newer release of Polyspace products, you can continue to submit jobs from both the older and the new releases. For instance, suppose that you were using the R2019a release of Polyspace products and MATLAB Parallel Server. When you upgrade to the R2019b releases of the two sets of products, you can continue to submit jobs from the older releases.

The setup for handling multiple releases of Polyspace, for instance, R2019a and R2019b, looks like this:



To support submissions from multiple releases with a single MATLAB Parallel Server cluster, you have to link the various installations of Polyspace products and MATLAB Parallel Server. Of the links shown in the preceding figure, the links numbered 1, 2 and 5 are part of a regular server-client installation using all products from the same release. To create all the links, do these steps:

- 1 In the R2019b installation of the Polyspace desktop products, open the user interface and select **Tools > Preferences**. Specify the computer that acts as the head node of the R2019b MATLAB Parallel Server cluster, as described earlier.

If you are able to submit jobs from R2019b Polyspace desktop products, you have already done this step.

- 2 In the R2019b installation of the MATLAB Parallel Server products, edit the `mjs_polyspace.conf` file to point to the R2019b installation of the Polyspace Server products, as described earlier.

If your R2019b installation of MATLAB Parallel Server can run an analysis using R2019b Polyspace Server products, you have already done this step.

- 3 In the R2019a installation of the Polyspace desktop products, open the user interface and select **Tools > Preferences**. Specify the computer that acts as the head node of the R2019b MATLAB Parallel Server cluster.

If you install the R2019b version of MATLAB Parallel Server on the same computer as the R2019a version, you have already done this step. Otherwise, this step might be new in the multi-release workflow.

- 4 In the R2019b installation of the MATLAB Parallel Server products, edit the file `mjs_def.bat` or `mjs_def.sh` (located in `matlabroot\toolbox\parallel\bin\`) to refer to the earlier release. Find the line with `MJS_ADDITIONAL_MATLABROOTS` and edit it as follows. To edit and save the file, you have to open your editor in administrator mode.

```
set MJS_ADDITIONAL_MATLABROOTS=othermatlabroot
```

Here, *othermatlabroot* is the installation path of the MATLAB Parallel Server installation from the earlier release, for instance:

```
C:\Program Files\MATLAB\R2019a
```

This step is new in the multi-release workflow.

- 5 In the R2019a installation of the MATLAB Parallel Server products, edit the `mjs_polyspace.conf` file to point to the R2019a installation of the Polyspace Server products, as described earlier.

If your R2019a installation of MATLAB Parallel Server can run an analysis using R2019a Polyspace Server products, you have already done this step.

Note

- The lowest release of MATLAB Parallel Server that can be mixed with the current release using this workflow is R2016a.
 - For releases from R2016a to R2018b, the server side configuration is different from releases R2019a and later. Consult the documentation for older releases for step 5 above. However, if you are already submitting jobs from a pre-R2019a Polyspace installation using a pre-R2019a MATLAB Parallel Server installation, you have presumably done this step.
-

See Also

Related Examples

- “Send Polyspace Analysis from Desktop to Remote Servers”
- “Send Polyspace Analysis from Desktop to Remote Servers Using Scripts”
- “Send Bug Finder Analysis from Desktop to Locally Hosted Server”
- “Fix Error: Job Manager Cannot Write to Database”
- “Install and Configure MATLAB Parallel Server for Third-Party Schedulers” (MATLAB Parallel Server)
- “Troubleshoot Common Problems” (MATLAB Parallel Server)

Install Products for Submitting Polyspace Analysis from Desktops to Servers Hosted on AWS

Cloud platforms enable you to easily scale your resources to meet your requirements and to securely access those resources from anywhere. This topic describes installation and configuration steps so that you can offload a Polyspace analysis from a client machine to a more powerful machine hosted on an Amazon® Web Services (AWS) cloud platform.

Note These instructions are valid only for Polyspace Server products version R2021b and later. To deploy versions R2021a and earlier to AWS, contact MathWorks Technical Support.

Prepare Your Installation

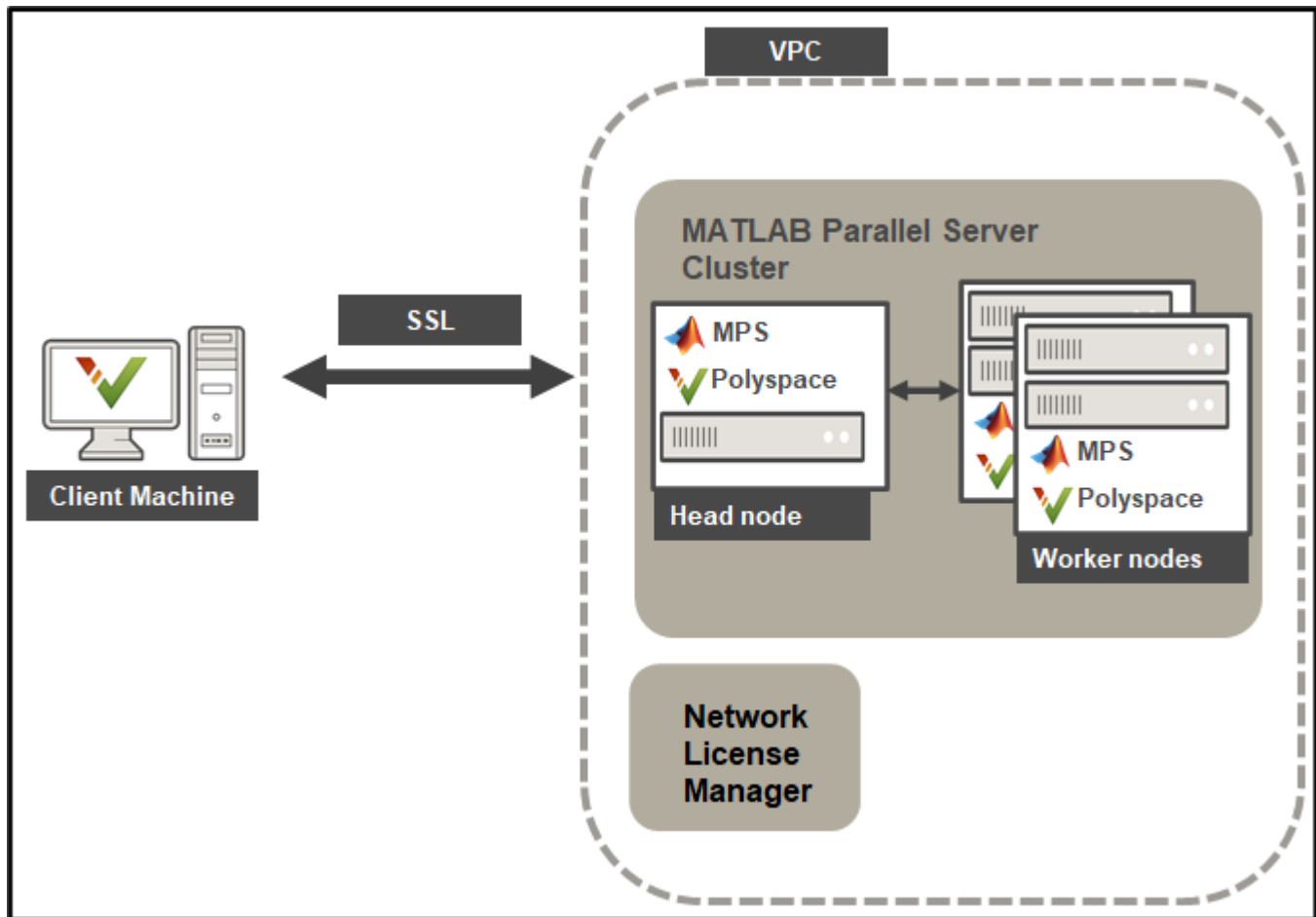
Before you begin your installation, make sure that you have:

- An AWS account
- A valid license for Polyspace Bug Finder or Code Prover Server and MATLAB Parallel Server.
- An SSH keypair for your region. To generate the keypair file, follow these instructions.
- A client machine with a Polyspace Desktop product already installed. See “Install Polyspace Desktop Products” on page 2-2.

To ease cloud deployments, MathWorks provides reference architectures that enable you to start a cluster in a few clicks by using a template. The clusters run on Linux virtual machines on EC2 resources in AWS. To complete this installation:

- 1** Install a license manager server. See “Configure and Deploy Network License Manager Stack” on page 3-22. This step is optional if you already have a license manager server.
- 2** Deploy the MATLAB Parallel Server and Polyspace resources to AWS. See “Configure and Deploy MATLAB Parallel Server and Polyspace Server Stack” on page 3-24.
- 3** Configure the inbound traffic rules to enable communication between the license server and the MATLAB Parallel Server cluster. See “Configure Security Groups” on page 3-26.
- 4** Configure the client machines that offload the analysis to the AWS cluster. See “Configure Client Machines” on page 3-26.

Polyspace does not support online licensing. To manage license checkouts, install a license server on the cloud or use an onsite license server.



Configure and Deploy Network License Manager Stack

If you already have a License Manager Server running and that server is accessible from the cloud, skip this section.

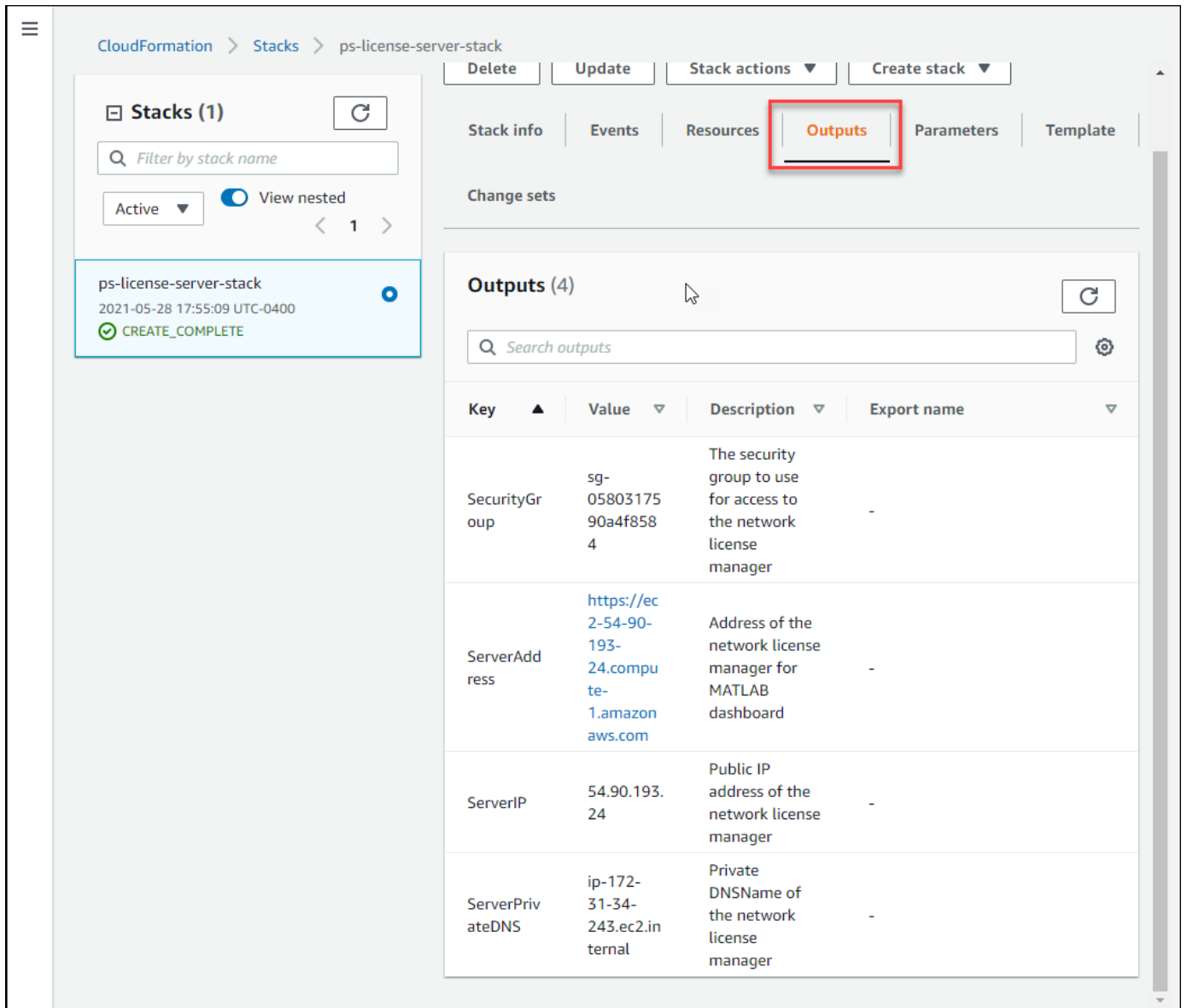
To deploy a network license manager on AWS, use the `license-manager-for-matlab-on-aws` architecture:

- 1 Select a release.
- 2 On the page for the release that you selected, click **Launch Stack** for your region. For example, `us-east-1`.
- 3 Complete the template configuration.

Parameter	Description
Stack name	Specify a name to identify the license manager instance. For example, <code>ps-license-server-stack</code> .
VPC to deploy this stack to	Select the ID of a virtual private cloud (VPC) from the drop-down list.

Parameter	Description
Subnet for the license server	Select an existing subnet from the drop-down list.
CIDR IP address range of client	<p>Specify public IP address that is allowed to connect to the server hosting the license manager from outside the VPC. Typically, this is the public IP of your client machine.</p> <p>To find this IP address, from the client machine, search “What’s my IP address” in a search engine or contact your IT administrator.</p> <p>For example, if your IP is 123.456.78, enter 123.456.78/32.</p>
Name of SSH key	Select an EC2 KeyPair from the drop-down list. To generate a keypair, see these instructions.
Password	Password you use to log into the license manager dashboard. Specify a password then confirm that password in the Confirm Password field.

- 4 Click **Create stack**. The process takes a few moments to complete. Keep the page open or bookmark it. You need information from the **Outputs** and **Parameters** tabs on this page to complete later configuration steps.



- 5 Once the status of the stack is **CREATE_COMPLETE**, go to the **Outputs** tab and click the **ServerAddress** URL to open the license manager dashboard.

Use the password from step 3 to log into the dashboard and follow the **Getting started** instructions to activate your license and start the license manager.

Configure and Deploy MATLAB Parallel Server and Polyspace Server Stack

To deploy the MATLAB Parallel Server, which includes Polyspace, on AWS, use the matlab-parallel-server-on-aws reference architecture.

- 1 Select a release. Polyspace products are available in the reference architecture only in R2021b and later releases.

On the page for the release that you selected, click **Launch Stack** for your region. For example, us-east-1.

- 2 Complete the template configuration.

Parameter	Description
Stack name	Specify a name to identify the MATLAB Parallel Server instance. For example, ps-mps-cloud.
Cluster name	Specify a name to identify the cluster. You use this name in the cluster configuration of Polyspace on the client machines.
Number of worker nodes	Specify a number equal to or smaller than the number of available licenses.
Number of workers to start on each node	Specify 1 worker per node. Make sure that the Instance type for the worker nodes is set to an appropriate instance type that meets or exceeds the minimum Polyspace hardware requirements (16 GB RAM and 4 cores). See Amazon EC2 Instance Types.
Name of SSH key	Select an EC2 KeyPair from the drop-down list. To generate a keypair, see these instructions. If you deployed a license manager on the cloud, this must match the value of the SSHKeyName on the Parameters tab of the license manager stack.
VPC to deploy this stack to	Select the ID of a virtual private cloud (VPC) from the drop-down list. If you deployed a license manager on the cloud, this must match the value of the VPC on the Parameters tab of the license manager stack.
Subnets for the head node and worker nodes	Select an existing subnet from the drop-down list. If you deployed a license manager on the cloud, this must match the value of the Subnet on the Parameters tab of the license manager stack.
CIDR IP address range of client	Specify public IP address that is allowed to connect to the server hosting the license manager from outside the VPC. Typically, this is the public IP of your client machine. To find this IP address, from the client machine, search “What’s my IP address” in a search engine or contact your IT administrator. For example, if your IP is 123.456.78, enter 123.456.78/32.
Additional security group to place instances in	If you deployed a license manager on the cloud, this must match the value of the SecurityGroup on the Outputs tab of the license manager stack. Otherwise, you can leave this field blank.

Parameter	Description
License Manager for MATLAB connection string	Specify the address of the license manager in the format <code>port@hostname</code> , for example, <code>27000@ip-123-45-67-89.ec2.internal</code> . Typically, the port number is 27000. If you deployed a license manager on the cloud, <code>hostname</code> must match the value of the ServerPrivateDNS on the Outputs tab of the license manager stack. Otherwise, if you use an onsite license manager, <code>hostname</code> corresponds to the hostname of the server that hosts the license manager.

- 3 Click **Create stack**. The process takes a few moments to complete.

Once the status of the stack is **CREATE_COMPLETE**, go to the **Outputs** tab and click the **BucketURL** URL.

- 4 On the following page, on the **Objects** tab, select the settings object and click **download** to download the cluster profile `.settings` file. You use this file when you configure the client machines to enable communication between the cluster and the client machine.

Configure Security Groups

After you start the License Manager and the MATLAB Parallel Server stacks, add an inbound rule to their respective security groups to allow each stack to communicate with the other stack. The security group defines the ports that are opened for inbound and outbound traffic. See Security groups for your VPC.

To edit the security groups:

- 1 From the **EC2** service page, click **Security Groups** on the left pane then, on the security groups page, click the **Security group ID** of the license manager stack. You can use the **Description** field to identify the security group of the license manager.
- 2 On the next page, select the **Inbound rules** tab and click **Edit inbound rules**. You might need to expand the bottom section of the page to see all the rules.
- 3 Scroll to the bottom of the page, click **Add rule**, and then select All TCP from the **Type** drop-down list. Add the MATLAB Parallel Server security group to the **Source** field by clicking inside that field and selecting the security group from the menu.
- 4 Click **Save rules** and then select the MATLAB Parallel Server security group and add a rule for the license manager security group by repeating steps 2 and 3.

If you did not deploy the license manager on AWS, make sure that the MATLAB Parallel Server security group has the necessary rules to communicate with your license manager.

Configure Client Machines

After you deploy the MATLAB Parallel Server and Polyspace Server on the cloud, configure the client machines that you use to offload the analyses to Polyspace in the cloud.

- 1 From the client machine, open the Polyspace desktop interface, go to **Tools > Preferences**, select the **Server Configuration** tab, and click **Settings**.

- 2 On the **Cluster Profile Manager** window toolstrip, click **Import**. Select the cluster profile `.settings` file that you downloaded in step 4 of “Configure and Deploy MATLAB Parallel Server and Polyspace Server Stack” on page 3-24. If you did not setup the MATLAB Parallel Server stack on AWS, contact your AWS administrator to obtain the cluster profile file.

Copy the values of the **MJSName** and **Host** fields from the **properties** tab. Click **Edit** at the bottom right of the window to make the field text selectable. You use these values in later configuration steps.

- 3 Click **Validate** on the toolstrip to check the communication with the AWS cluster. Some issues that might cause the validation to fail are:
 - Incorrect CIDR IP address. Check the stack settings or contact your cluster administrator.
 - Incorrect security group configuration. Check the security group settings or contact your cluster administrator.
 - Invalid MATLAB Parallel Server license. Check that you are using the correct license or contact your license administrator.

This validation step does not check the validity of your Polyspace Server product license. To check that you are using a valid license, see “Check the Polyspace Server License” on page 3-27.

- 4 Close the **Cluster Profile Manager** and enter the **MJSName** that you obtained in step 2 in the **Job Scheduler host name** field.
- 5 Restart the Polyspace desktop interface to apply the new settings.

To start offloading analysis to Polyspace in the cloud, in the **Configuration** pane of the Polyspace desktop interface, select the **Run Settings** node and enable analysis on a remote cluster. See also “Bug Finder Analysis on Clusters”.

Check the Polyspace Server License

The validation step that you perform in the **Cluster Profile Manager** does not check the validity of the Polyspace Server license. To check the Polyspace Server license, at the command line on the client machine, navigate to the Polyspace desktop installation folder, for instance `C:\Program Files\Polyspace\R2023a`, and run this command:

- Windows

```
polyspace\bin\polyspace-bug-finder.exe -ver -batch -scheduler MJSName@host
```

- Linux

```
polyspace/bin/polyspace-bug-finder -ver -batch -scheduler MJSName@host
```

where *MJSName* and *host* are the values that you obtained from the **Cluster Profile Manager** in the step 2 of “Configure Client Machines” on page 3-26.

The command outputs system information about the machine where the Polyspace Server products are deployed. If you have a valid license, you see a message similar to this message in the output:

Polyspace related licenses and license feature names:

```
Polyspace Bug Finder Server: Polyspace_BF_Server
Polyspace Code Prover Server: Polyspace_CP_Server
```

If you get a license error message or if the output does not list any Polyspace products under the line `Polyspace related licenses and license feature names:`, your Polyspace license might be invalid. Contact your Polyspace license administrator.

Update or Delete Resources

To upgrade to a newer release of the reference architectures, you must delete the existing resources and deploy the new version.

To delete the existing resources, in your AWS account, go to **CloudFormation**, select a stack, and click **Delete**.

If you update the license manager to a newer version, the new resource has a different host ID. Even if you choose to keep the same license manager stack, you must update your license file when you update to a new version of Polyspace.

Install Products for Submitting Polyspace Analysis from Desktops to Servers Hosted on Azure

Cloud platforms enable you to easily scale your resources to meet your requirements and to securely access those resources from anywhere. This topic describes installation and configuration steps so that you can offload a Polyspace analysis from a client machine to a more powerful machine hosted on a Microsoft® Azure cloud platform.

Note These instructions are valid only for Polyspace Server products version R2021b and later. To deploy versions R2021a and earlier to Azure, contact MathWorks Technical Support.

Prepare Your Installation

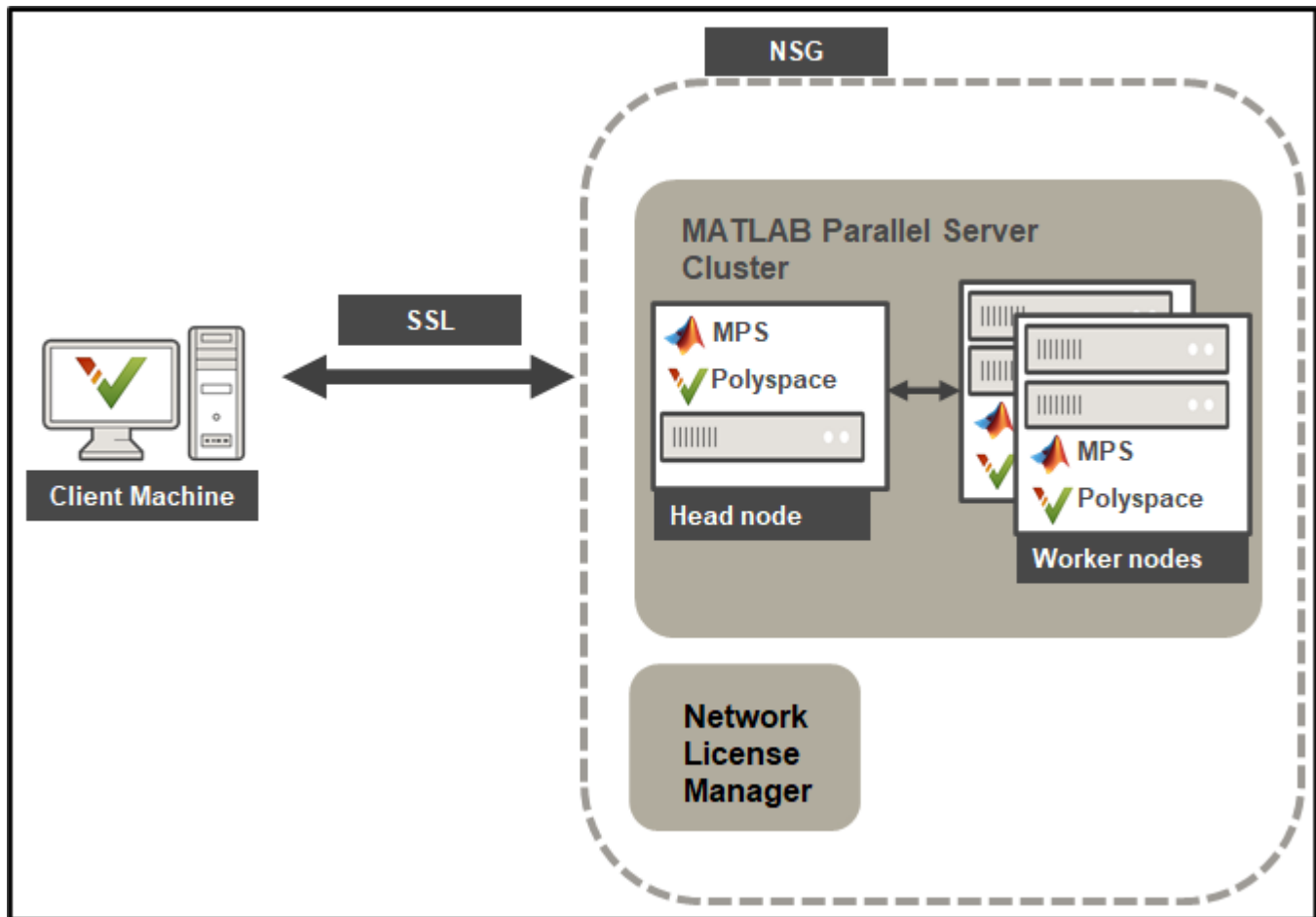
Before you begin your installation, make sure that you have:

- A Microsoft Azure account.
- A valid license for Polyspace Bug Finder or Code Prover Server and MATLAB Parallel Server.
- A client machine with a Polyspace Desktop product already installed. See “Install Polyspace Desktop Products” on page 2-2.

To ease cloud deployments, MathWorks provides reference architectures that enable you to start a cluster in a few clicks by using a template. The clusters run on Windows virtual machines. To complete this installation:

- 1** Install a license manager. See “Configure and Deploy Network License Manager Resources” on page 3-30. This step is optional if you already have a license manager server.
- 2** Deploy the MATLAB Parallel Server and Polyspace resources on Azure. See “Configure and Deploy MATLAB Parallel Server and Polyspace Server Resources” on page 3-32.
- 3** Configure the client machines that offload the analysis to the Azure cluster. See “Configure Client Machines” on page 3-34.

Polyspace does not support online licensing. To manage license checkouts, install a license server on the cloud or use an onsite license server.



Configure and Deploy Network License Manager Resources

If you already have a License Manager Server running and that server is accessible from the cloud, skip this section.

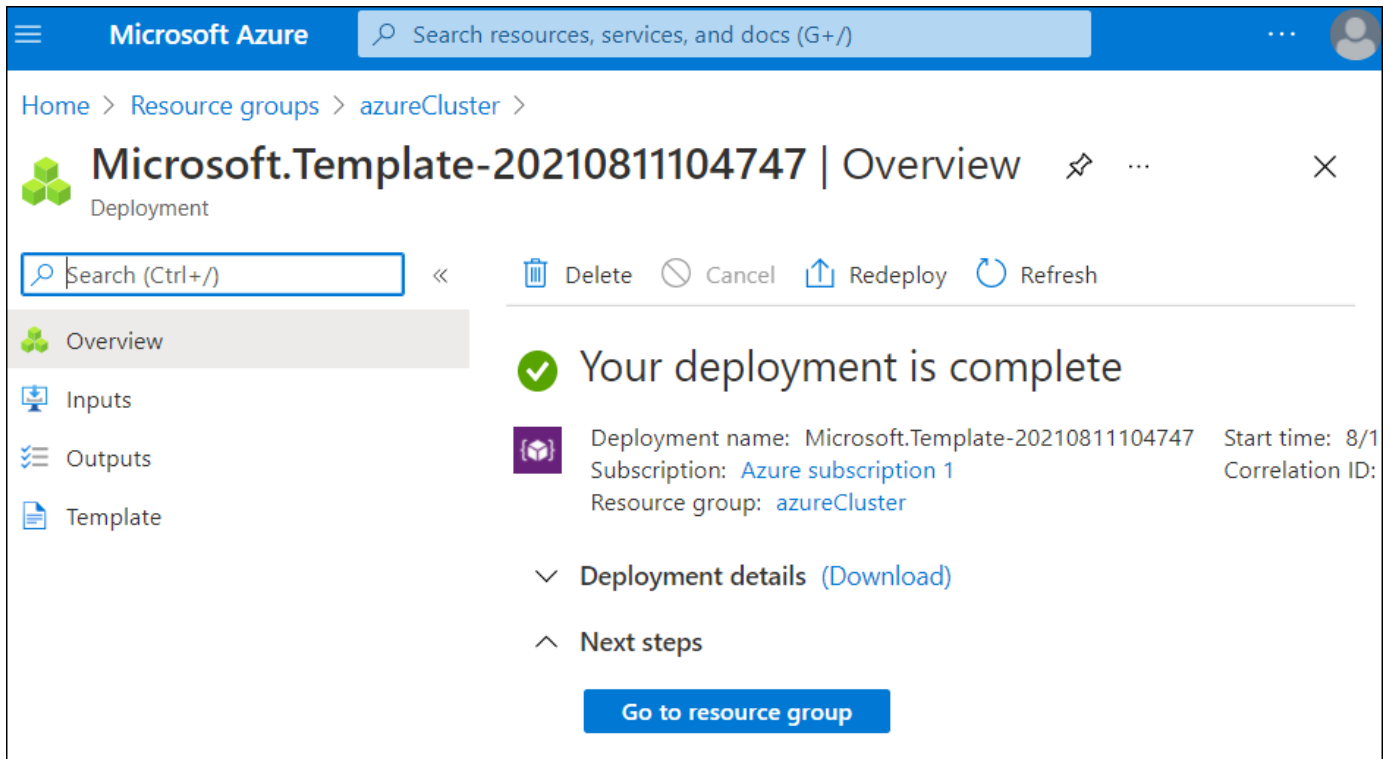
To deploy a network license manager on Microsoft Azure, use the `license-manager-for-matlab-on-azure` architecture:

- 1 Select a release. Polyspace products are available in the reference architecture only in R2021b and later releases.
- 2 On the page for the release that you selected, click **Deploy to Azure**. You can deploy the resources in a new virtual network or in an existing virtual network. You might be asked to log into your Azure account.
- 3 Complete the template configuration.

Parameter	Description
Resources group	Select a resource group from the drop-down list or click Create new to create a group. See What is a resource group .

Parameter	Description
Region	Select a region from the drop-down list. This field is read-only if you use an existing resource group.
Client IP address	<p>Specify a public IP address that is allowed to connect to the server hosting the license manager from outside the network security group (NSG). Typically, this is the public IP of your client machine.</p> <p>To find this IP address, from the client machine, search “What’s my IP address” in a search engine or contact your IT administrator.</p> <p>For example, if your IP is 123.456.78, enter 123.456.78/32.</p>
Admin Password	Password you use to log into the license manager dashboard.
Virtual Network Resource ID	<p>Specify the resource ID of an existing virtual network. To find the resource ID of a virtual network:</p> <ol style="list-style-type: none"> a From the Azure portal, click Resource groups and select the resource group of the virtual network on the next page. b On the resource group page, click the name of the virtual network. c Click Properties in the left pane and copy the Resource ID field. <p>You specify this parameter only if you deploy the license server to an existing virtual network.</p>
Subnet Name	<p>Specify the name of an existing virtual network subnet. To find the subnet name, follow steps a and b from the previous row, then click Subnets in the left pane and copy the Name field.</p> <p>You specify this parameter only if you deploy the license server to an existing virtual network.</p>

- 4** Click **Review + create**. After the final validation completes, click **Create**. The creation of the resources takes a few moments to complete.
- 5** Once the deployment is complete, click **Outputs** in the left pane and copy the **networkLicenseManagerAddress** URL. You use this URL to access the license manager dashboard.



Use the password from step 3 to log into the dashboard and follow the **Getting started** instructions to activate your license and start the license manager.

Configure and Deploy MATLAB Parallel Server and Polyspace Server Resources

To deploy the MATLAB Parallel Server, which includes Polyspace, on Microsoft Azure, use the matlab-parallel-server-on-azure reference architecture.

- 1 Select a release. Polyspace products are available in the reference architecture only in R2021b and later releases.
- 2 On the page for the release that you selected, click **Deploy to Azure**. You can deploy the resources in a new virtual network or in an existing virtual network. If you deployed your license manager on Azure, deploy MATLAB Parallel Server within the same virtual network as the license manager.
- 3 Complete the template configuration.

Parameter	Description
Resources group	Select a resource group from the drop-down list or click Create new to create a new group. See What is a resource group. If you deployed the license manager on Azure, select the resource group of the license manager.
Region	Select a region from the drop-down list. This field is read-only if you use an existing resource group.

Parameter	Description
Cluster Name	Specify a name for the MATLAB Parallel Server cluster. You use this name to specify the cluster when you configure client machines.
Num Worker Nodes	Specify the number of Azure instances to start for worker nodes to run on, for instance 1.
Num Workers Per Node	Specify the number of workers to start on each node. It is recommended to specify one worker for every two vCPUs. For more on the number of vCPUs for each type of Azure instance, see Sizes for virtual machines in Azure.
Head Node Vm Size	Specify the Azure instance type that you use for the head node. This node runs only the job manager and can be a smaller instance size. See Sizes for virtual machines in Azure.
Worker Vm Size	Specify the Azure instance type that you use for the worker nodes. Make sure that the instance type meets or exceeds the minimum Polyspace hardware requirements (16 GB RAM and 4 cores). See Sizes for virtual machines in Azure.
Client IP address	Specify a public IP address that is allowed to connect to the server hosting the license manager from outside the network security group (NSG). Typically, this is the public IP of your client machine. To find this IP address, from the client machine, search "What's my IP address" in a search engine or contact your IT administrator. For example, if your IP is 123.456.78, enter 123.456.78/32.
Admin Password	Password you use in combination with username "matlab" to log into any instance using remote desktop protocol.
Virtual Network Resource ID	Specify the resource ID of an existing virtual network. To find the resource ID of a virtual network: <ul style="list-style-type: none"> a From the Azure portal, click Resource groups and select the resource group of the virtual network on the next page. b On the resource group page, click the name of the virtual network. c Click Properties in the left pane and copy the Resource ID field. <p>You specify this parameter only if you deploy MATLAB Parallel Server and Polyspace to an existing virtual network.</p>
Subnet Name	Specify the name of an existing virtual network subnet, for instance netlm-subnet. To find the subnet name, follow steps a and b from the previous row, then click Subnets in the left pane and copy the Name field. You specify this parameter only if you deploy MATLAB Parallel Server and Polyspace to an existing virtual network.

Parameter	Description
License Server	Specify the address of the license manager in the format port@hostname, for example, 27000@netlm-server. The port and hostname that you specify must be accessible from the virtual network and subnet where you deploy the MATLAB Parallel Server resources.

- 4 Click **Review + create**. After the final validation completes, click **Create** . The creation of the resources takes a few moments to complete.
- 5 Once the deployment is complete, click **Go to resource group** and then, on the resource group page, click the storage account.

Showing 1 to 14 of 14 records. Show hidden types ⓘ No grouping ▾

<input type="checkbox"/> Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/> mpsazu-headnode_OsDisk_1_b8c6bbcc12eb4402b62b82ba59a2ad49	Disk	East US
<input type="checkbox"/> mpsazu-headnodenic	Network interface	East US
<input type="checkbox"/> mpsazurecluster-nsg	Network security group	East US
<input type="checkbox"/> mpsazureclusterergzstorage	Storage account	East US
<input type="checkbox"/> mpsazureclusterpub	Public IP address	East US
<input type="checkbox"/> netlm-nsg	Network security group	East US
<input type="checkbox"/> netlm-server	Virtual machine	East US
<input type="checkbox"/> netlm-server-ip	Public IP address	East US

- 6 On the storage account page:
 - a Click **File shares** in the left pane.
 - b Click **shared** and then **cluster**.
 - c Download the cluster profile `.settings` file. You use this file when you configure the client machines to enable communication between the cluster and the client machine.

Configure Client Machines

After you deploy the MATLAB Parallel Server and Polyspace Server to the cloud, configure the client machines that you use to offload the analyses to Polyspace in the cloud.

- 1 From the client machine, open the Polyspace desktop interface, go to **Tools > Preferences**, select the **Server Configuration** tab, and click **Settings**.
- 2 On the **Cluster Profile Manager** window toolbar, click **Import**. Select the cluster profile `.settings` file that you downloaded in step 6 of “Configure and Deploy MATLAB Parallel Server and Polyspace Server Resources” on page 3-32. If you did not configure and deploy the MATLAB Parallel Server resources on Azure, contact your Azure administrator to obtain the cluster profile file.

Copy the values of the **MJSName** and **Host** fields from the **properties** tab. Click **Edit** at the bottom right of the window to make the field text selectable. You use these values in later configuration steps.

- 3 Click **Validate** on the toolstrip to check the communication with the Azure cluster. Some issues that might cause the validation to fail are:
 - Incorrect client IP address. Check the resources settings or contact your cluster administrator.
 - Incorrect virtual network port configuration. Check the virtual network settings or contact your cluster administrator. See also Port requirements for accessing MATLAB Parallel Server.
 - Invalid MATLAB Parallel Server license. Check that you are using the correct license or contact your license administrator.

This validation step does not check the validity of your Polyspace Server product license. To check that you are using a valid license, see “Check the Polyspace Server License” on page 3-35.

- 4 Close the **Cluster Profile Manager** and enter the **MJSName** that you obtained in step 2 in the **Job Scheduler host name** field.
- 5 Restart the Polyspace desktop interface to apply the new settings.

To start offloading an analysis to Polyspace in the cloud, in the **Configuration** pane of the Polyspace desktop interface, select the **Run Settings** node and enable analysis on a remote cluster. See also “Bug Finder Analysis on Clusters”.

Check the Polyspace Server License

The validation step that you perform in the **Cluster Profile Manager** does not check the validity of the Polyspace Server license. To check the Polyspace Server license, at the command line on the client machine, navigate to the Polyspace desktop installation folder, for instance `C:\Program Files\Polyspace\R2023a`, and run this command:

- Windows

```
polyspace\bin\polyspace-bug-finder.exe -ver -batch -scheduler MJSName@host
```

- Linux

```
polyspace/bin/polyspace-bug-finder -ver -batch -scheduler MJSName@host
```

where *MJSName* and *host* are the values that you obtained from the **Cluster Profile Manager** in step 2 of “Configure Client Machines” on page 3-34.

The command outputs system information about the machine where the Polyspace Server products are deployed. If you have a valid license, you see a message similar to this message in the output:

```
Polyspace related licenses and license feature names:
```

```
Polyspace Bug Finder Server: Polyspace_BF_Server
Polyspace Code Prover Server: Polyspace_CP_Server
```

If you get a license error message or if the output does not list any Polyspace products under the line `Polyspace related licenses and license feature names:`, your Polyspace license might be invalid. Contact your Polyspace license administrator.

Update or Delete Resources

To upgrade to a newer release of the reference architectures, you must delete the existing resources and deploy the new version.

To delete the existing resources:

- 1 From the Azure portal, click **Resource groups**, and then click the name of the resource group that you want to do delete.
- 2 On the resource group page, click **Delete resource group** and follow the instructions.

If you update the license manager to a newer version, the new resource has a different host ID. Even if you choose to keep the same license manager stack, you must update your license file when you update to a new version of Polyspace.

Install Polyspace Bug Finder Access Web Interface

System Requirements for Polyspace Access

Required Software

- The installation of Polyspace Access components requires Docker version 19.03 or later.
- Polyspace Access is compatible with Docker Engine on Linux.

To install Docker Engine on your machine, click one of these Docker supported Linux platforms and follow the installation instructions. The installation of Polyspace Access on a Linux platform is recommended.

- Ubuntu
- Debian
- CentOS
- Fedora

Docker Engine is also available on other Linux distributions.. For Docker on SUSE and Red Hat, see Docker EE on Linux distros.

Tip You can use Kubernetes[®] to deploy Polyspace Access. For more information, contact MathWorks Technical Support.

- The configuration of some Polyspace Access services requires the `openssl` toolkit to generate public and private keys. This toolkit is also required to configure Polyspace Access with HTTPS.
- To connect Polyspace Access with Polyspace desktop user interfaces over HTTPS, you must use the Java Platform, Standard Edition Development Kit (JDK). You use the JDK to generate a Java Key Store file.
- Polyspace Access licenses are network named user (NNU) licenses that require a license manager. Polyspace uses the FlexNet[®] Publisher (FLEXlm[®]) license manager. See “Install License Manager” on page 6-5.

Windows Requirements

To install Polyspace Access inside Linux virtual machines on Windows Server 2016 and 2019, you must:

- Enable virtualization in your BIOS.
- Install and enable Hyper-V.
- Create a virtual switch for Hyper-V.

You must also enable nested virtualization if you run Hyper-V inside of a Hyper-V VM.

The installation of Polyspace Access is not supported on Windows Subsystem for Linux (WSL) and WSL2. See “Polyspace Access Support on WSL” on page 4-12.

Hardware and Other Requirements

- The minimum hardware configuration that is recommended for Polyspace Access is:

- 4 cores
- 32 GB of RAM
- 500 GB of disk space

Frequent uploads might affect the performance of Polyspace Access. For instance, if you have a continuous integration process that regularly uploads more than 10 analysis results per hour, consider setting up multiple instances of Polyspace Access. For more information, contact MathWorks Technical Support.

- Data transfers between the server and client machines require a high speed network connection. A gigabit network connection is recommended.
- The Polyspace Access Cluster Admin does not support the Internet Explorer web browser.
- It is recommended to install Polyspace Access on the local drive of a physical machine.
 - The Polyspace Access database might run less efficiently and stored data might be corrupted if you install on a network drive.
 - The installation of Polyspace Access on a virtual machine might result in up to a 50% overhead during I/O operations.

See Also

More About

- “Install Polyspace Access for Web Reviews”

Storage and Port Configuration

Storage Configuration

- To ensure optimal data storage performance, use physical drives instead of networked storage solutions.
- The database is stored under *polyspaceAccessRoot/appdata/polyspace-access* where *polyspaceAccessRoot* is the folder where you unzip the Polyspace Access installation image. Make sure that you have adequate disk space for the Database mount point.
- It is a best practice to secure the database mount point with a RAID array and to back up the mount point regularly.
- Allocate this recommended amount of disk space for the mount points of the working directories of the Polyspace Access processes:

- **Temporary upload directory:** 40 GB

Uploaded files are stored in this folder while they are transferred to the web server mount point.

- **Upload directory:** 10 GB

Once the transfer to the web server mount point is complete, files are moved to this directory. The path to this directory must be the same for the extract-transform-load (ETL) and web server services. If the services are on different machines, the paths to this directory must point to the same hard drive.

- **Storage directory:** 20 GB

The ETL (import process) looks for files in the upload directory and stores them in the storage directory. Files that are successfully uploaded to the database are deleted. Files that fail to upload are sent to the invalid results directory.

- **Working directory:** 10 GB

The ETL (import process) uses this directory to process files from the storage directory. Files are treated in the order in which they are received. The data is prepared to be sent to the database.

- **Invalid results directory:** 50 GB

Files that fail to upload are stored in this directory. You can recover and analyze the files to determine why the upload failed. Back up this folder regularly. Set up a policy to determine the amount of time after which older data can be deleted.

- Make sure that all users have read and write permissions for the directories you specify under **Polyspace Access ETL** and the **Temporary upload directory** in the **Cluster Admin** settings.

Network Port Configuration

When you configure Polyspace Access, you specify a port number that client machines use to communicate with the Polyspace Access services. To avoid installation errors, and to verify that the services are accessible, make sure that the port that you specify is open. To check whether a port *portNumber* is open, use these commands:

Windows PowerShell	<code>netstat -na find "portNumber"</code>
Linux	<code>netstat -na grep portNumber</code>

If the output of the command is empty, the port is not in use. If the port is in use, specify a different port or stop the process currently using the port.

Check the availability of these ports if you install Polyspace Access on a single node with a default configuration:

- 9443 — Polyspace Access default port.
- 27000 — Inbound port of license manager daemon that is required to manage license checkouts.
- Make sure that you also check the availability of the vendor daemon (MLM) port. If you did not specify a port for the vendor daemon, check the license manager log file to find the assigned port. To specify a port for the vendor daemon, see step 2 of “Configure Polyspace Access License” on page 6-2.

If you do not use a default configuration, for instance if you specify the `--force-exposing-ports` option when you start the `admin-docker-agent` binary, check the availability of all the ports that you specify on the **Services** tab of the **Nodes** settings.

See Also

More About

- “Install Polyspace Access for Web Reviews”
- “Database Backup” on page 4-51
- “Configure and Start the Cluster Admin” on page 4-13

Create a Linux Virtual Machine by Using Hyper-V

You can install Polyspace Access on Windows Server® 2016 and 2019 by creating a virtual machine (VM) that runs a Linux distribution, and then installing Polyspace Access inside that VM.

Warning The use of Polyspace Access inside a VM might result in up to a 50% overhead during I/O operations compared to using Polyspace Access on a physical machine.

Prerequisites

Before you create the VM:

- Make sure that Hyper-V is enabled on your machine.

Open PowerShell by pressing the Windows+X keys and clicking **Windows PowerShell (Admin)**.

In the PowerShell command prompt, enter:

```
(Get-WindowsOptionalFeature -featurename Microsoft-hyper-v -online).state
```

If the command does not return **Enabled**, enter:

```
Install-WindowsFeature -Name Hyper-V -IncludeManagementTools -Restart
```

The command enables Hyper-V and restarts your machine.

Open the Hyper-V Manager by pressing the Windows key and typing HyperV, then click **Action > Connect to Server** and select **Local computer**.

- Make sure that an external virtual switch has been created in Hyper-V.

In a PowerShell command prompt, enter:

```
Get-VMSwitch | where SwitchType -eq 'External'
```

If the command does not return anything, follow these instructions to create an external virtual switch. Running this command might require administrator privileges.

- Download an ISO image for a Linux distribution that is supported by Docker, for instance Ubuntu Server. For a list of Linux distributions that are available for Docker Engine or Docker Engine Enterprise (EE), see supported platforms for Docker Engine and Docker EE on Linux distros.
- Download and install the network license manager. See “Install License Manager” on page 6-5.

Create a Virtual Machine

To create a virtual machine, open the Hyper-V manager. In the **Actions** pane, click **New > Virtual Machine**.

Follow the prompts in the **New Virtual Machine Wizard** window.

- For the **Specify Generation** step, select **Generation 2**.
- For the **Assign Memory** step, allocate enough memory to meet the requirements on page 4-2 for Polyspace Access. The recommended minimum memory is 32 GB.
- For the **Configure Networking** step, select the switch that corresponds to the external connection type.

- For the **Connect Virtual Hard Disk** step, the size of the virtual hard drive must meet the requirements on page 4-2 of the Polyspace Access database. The recommended minimum disk size is 500 GB.
- For the **Installation Options** step, select **Install an operating system from a bootable image file**, and provide the path to the Linux ISO image that you downloaded.

After you click **Finish** and the wizard closes, right-click the newly created VM in the **Virtual Machines** pane and click **Settings**. In the settings window, click **Security** in the left pane, select **Enable Secure Boot** and, choose **Microsoft UEFI Certificate Authority** from the **Template** drop-down. Secure boot helps preventing the loading utility of the operating system from running unauthorized code at boot time. For a list of Linux distributions that Microsoft supports for secure boot, see Supported Linux and FreeBSD virtual machines for Hyper-V on Windows.

Start and Configure the Virtual Machine

To start the virtual machine (VM), in the Hyper-V manager, right-click the VM name in the **Virtual Machines** pane, and then click **Connect**. If this is the first time that you are starting the VM, follow the prompts to install the Linux distribution you specified in the **Installation Options** step when you created the VM.

During this installation process, you specify a hostname for the Linux machine and a username and password to log into the Linux machine. Enter this password when you use the `sudo` command in later configuration steps.

After you install the Linux distribution, restart the VM and open a Linux command-line terminal.

- Install the Docker engine. For installation instructions, see the Docker documentation ,for instance [Get Docker Engine - Community for Ubuntu](#).

Once you install the Docker engine, add the current user to the `docker` group. Only users that are in the `docker` group can run Docker commands. In the terminal, enter:

```
sudo usermod -aG docker $USER
```

- Install the `openssl` utility. The utility allows you to generate public/private key pairs to configure the **User Manager** service and to generate the necessary certificates if you enable HTTPS for Polyspace Access. For instance, on Ubuntu, enter this command:

```
sudo apt install openssl
```

If `openssl` is already installed, this command has no effect.

- Install the `openssh-server` server and make sure that port 22 is enabled in the firewall configuration. You can then remote into the Linux machine by using SSH or securely transfer files to the Linux machine. For instance, on Ubuntu, enter these commands:

```
sudo apt install openssh-server
sudo ufw allow 22
```

If `openssh-server` is already installed, the `install` command has no effect. Once you complete this step, you can use a command such as `scp` to securely transfer files between your Windows Server 2016 machine and the Linux VM.

For example, if you use username `accessUser` to log into the Linux VM with hostname `access-vm-lnx`, you can transfer file `myFile.txt` by entering this command from the Windows Server machine:

```
scp pathT0\myFile.txt accessUser@access-vm-lnx:~
```

The command copies the file to folder /home/accessUser on the Linux VM.

pathT0 is the path to myFile.txt.

- Once you complete the previous configuration steps, restart the VM.

To install Polyspace Access, see “Install Polyspace Access for Web Reviews” and “Manage Polyspace NNU Licenses”.

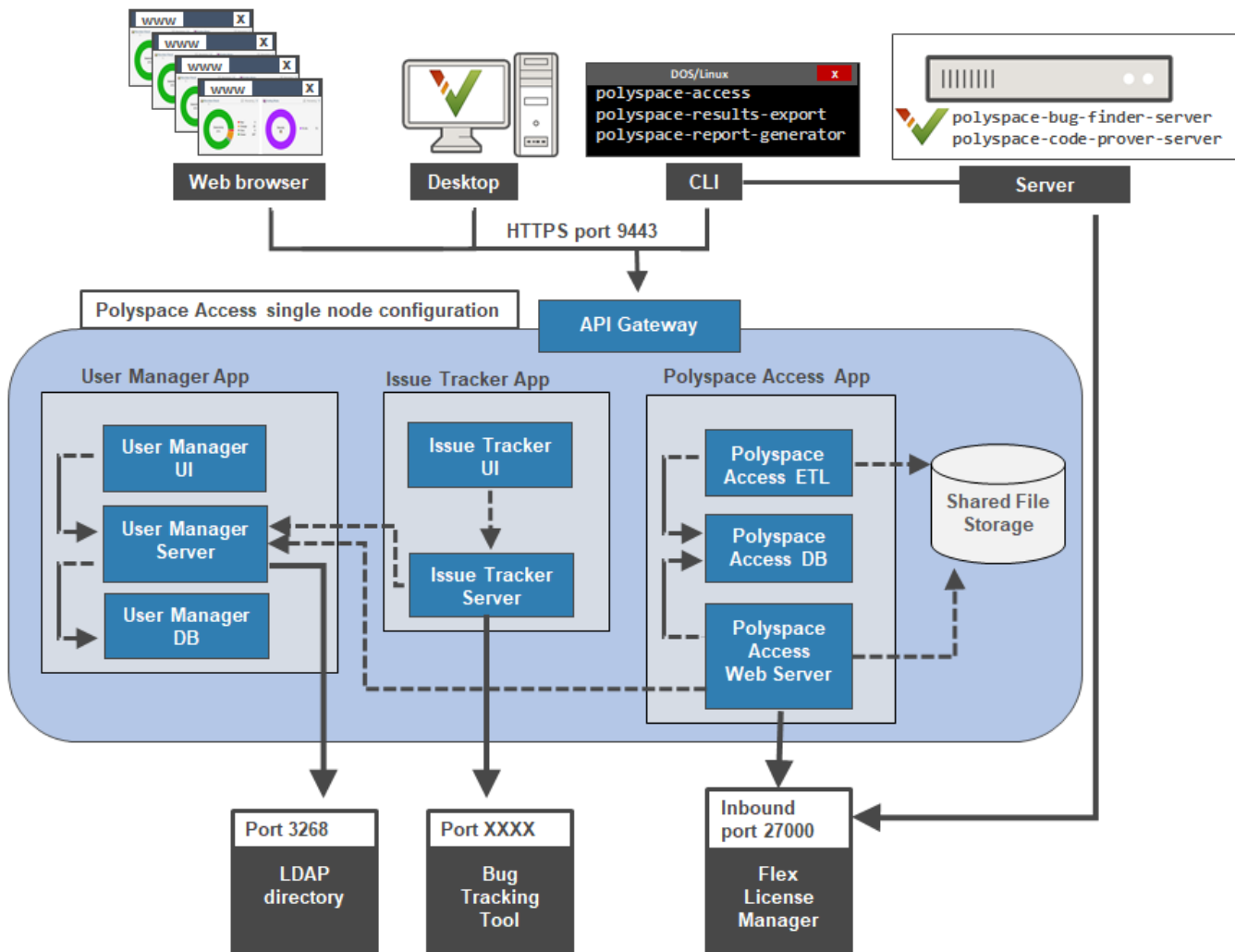
Prepare Your Installation

Polyspace Access provides a web browser interface so that you can review Polyspace analysis results that are hosted on a centralized database. When you install Polyspace Access, you install these apps:

- **User Manager App:** Authenticates user logins against your company Lightweight Directory Access Protocol (LDAP) or against a custom internal database of users. The app issues signed JSON Web Tokens to authenticated users and provides a user interface to manage the custom database of users.
- **Issue Tracker App:** Manages the communication between Polyspace Access and your bug tracking tool (BTT) software and provides a user interface to create BTT tickets.
- **Polyspace Access App:** Manages results uploads to the Polyspace Access database and results exports from that database, and provides a user web interface to review results.

The Polyspace Access ETL (Extract-Transform-Load) service handles operations such as results uploads, results downloads, and updates to the data displayed in the user web interface.

Each app contains services that are deployed inside docker containers. A separate **Gateway** service handles communications between Polyspace Access and client machines.



Before you begin your installation, decide whether to use the HTTPS protocol and how you to configure user authentication and bug tracking tool integration.

Tip You can use Kubernetes to deploy Polyspace Access. For more information, contact MathWorks Technical Support.

General Prerequisites

- Install the required software and make sure that your system meets the minimum hardware requirements. See “System Requirements for Polyspace Access” on page 4-2.
- Verify that you have enough data storage available and that the ports that you use are available and not blocked by your firewall. See “Storage and Port Configuration” on page 4-4.
- Check that the license manager is installed and running, and that the license manager options file includes the users to whom you grant right-to-use privileges for Polyspace Access. See “Manage Polyspace NNU Licenses”.

- To avoid any potential issues with license file operation, consider upgrading the network license manager software whenever you upgrade Polyspace Access. See “Update Network License Manager Software”.
- Create and configure a Linux virtual machine (VM) if you are installing Polyspace Access inside a Linux VM on Windows Server 2016 or 2019. See “Create a Linux Virtual Machine by Using Hyper-V” on page 4-6.
- If you enable HTTPS to encrypt communications between Polyspace Access and client machines, obtain an SSL private key and a signed certificate from a certificate authority or use self-signed certificates. See “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14
- If you configure the Polyspace Access to use the HTTPS protocol, you must generate a Java Key Store (JKS) file to enable communications between Polyspace Access and the Polyspace desktop interface, or the `polyspace-results-export` and `polyspace-report-generator` binaries. See “Generate a Client Keystore” on page 4-48.
- Check that the docker network does not conflict with an existing network. For more information on docker networks, see Networking overview.

To check if your docker network conflicts with an existing network, run this command:

```
docker network inspect networkName
```

and check that the IP range listed in the `IPAM.Config` node is not used by other services. You might need to contact your network administrator to determine if the IP range is used by other services.

To find *networkName*, use the command `docker network ls`. If the command returns more than one network, inspect all the bridge networks (see the `DRIVER` column in the output).

Create a network and specify the subnet and gateway to avoid conflicts with existing networks. See `docker network create`. To use the new network, use the `--network-name` option when you start the `admin-docker-agent` binary.

User Manager Prerequisites

- The **User Manager** configuration requires the generation of an SSL private key file. See “Configure User Manager” on page 4-21. This private key must be different from the private key that you use for the HTTPS configurations.
- If you use your company LDAP to authenticate user logins, contact your LDAP administrator to:
 - Obtain the LDAP URL and LDAP base that your organization uses.
 - Obtain LDAP login credentials if access to the LDAP server is password-protected.
 - Discuss an LDAP search filter that enables you to retrieve specific subsets of users from the LDAP database. See LDAP filters.
- If you use LDAP configured over SSL (LDAPS), you must add the LDAP SSL certificate to the certificate trust store file that you use for Polyspace Access. See “Configure the User Manager for LDAP over SSL” on page 4-27.

Issue Tracker Prerequisites

- The **Issue Tracker** configuration requires the URL that you use to connect to your BTT interface.

- If you use the Jira software with the OAuth authentication method, you must first create an application link in Jira. See the first step on this page.
- If you use the Redmine BTT, contact your Redmine administrator to obtain the Redmine API key.
- If you use a BTT configured by using HTTPS, you must add the BTT SSL certificate to the certificate trust store file that you use for Polyspace Access. See “Add BTT Instance Configured by Using HTTPS” on page 4-31.

Polyspace Access Support on WSL

The installation of Polyspace Access on the Windows Subsystem for Linux (WSL) is not supported.

WSL is a Windows feature that enables you to execute Linux command-line tools on a Linux file system from Windows while using fewer resources than a regular virtual machine (VM). While you can use WSL for development, this feature lacks some of the functionalities of a regular Linux operating system. See [Can I use WDL for production scenarios?](#) .

Note that WSL2 offers improved performance but relies on virtualized networking components and does not perform well across OS file systems. Both of these issues affect the performance of Polyspace Access. See [Comparing WSL Versions](#).

Configure and Start the Cluster Admin

The Cluster Admin is an agent that enables you to install, configure, and start the Docker containers for the different Polyspace Access services.

Prerequisites

Before configuring and starting the Cluster Admin, make sure that:

- You have downloaded the Polyspace Access installation image.

To download the installation image, go to the MathWorks download page, expand the **Get Polyspace Access** section and click **Download**. Make sure that the text under the download button lists the release and update number (when applicable) that you want to get.

- Docker is running on your machine. At the command line, type:

```
docker stats --no-stream
```

If you get an error message, run the command `sudo systemctl start docker`. If `systemctl` is not available, use `service` instead.

After you start Docker, you must be logged in as a member of the `docker` group to run Docker commands. To see a list of current members of this group, use the command:

```
grep 'docker' /etc/group
```

To add the current user to the `docker` group, use the command:

```
sudo usermod -aG docker $USER
```

Unzip Installation Image and Start Cluster Admin Agent

The Cluster Admin `admin-docker-agent` binary is included with the `polyspace-access-VERSION.zip` installation image for Polyspace Access. *VERSION* is the release version, for instance R2023a. After you download the installation image, unzip it to extract these files and folders:

```
admin-docker-agent*
admin-docker-agent.exe*
admin.tar
appdata/
download/
gateway.tar
issuetracker-server-main.tar
issuetracker.tar
issuetracker-ui-main.tar
lm/
polyspace-access-db-main.tar
polyspace-access-download-main.tar
polyspace-access-etl-main.tar
polyspace-access.tar
polyspace-access-web-server-main.tar
products/
usermanager-db-main.tar
usermanager-server-main.tar
usermanager.tar
usermanager-ui-main.tar
VERSION
```

To start the `admin-docker-agent` binary, from the command line, navigate to the installation folder where you extracted the contents of the zip installation image. Once inside this folder, at the command-line, enter:

```
admin-docker-agent
```

The command line outputs messages indicating that the agent is downloading image layers. After the download is complete, you see a message with information on how to connect to the agent:

```
time="2020-07-10T14:23:11Z" level=info msg="Cluster Admin started. You can now connect to the Cluster Admin through your web browser at http://localhost:9443/admin using the initial password randomPass
```

randomPass is a randomly generated initial password. Copy this password. The command-line output shows the password only the first time you start **Cluster Admin**.

By default, the **Cluster Admin** uses the HTTP protocol and starts with hostname localhost and port 9443.

- To configure the **Cluster Admin** with HTTPS, see “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14.
- If the default port is already in use, you get a `Permission denied` error message. Use the flag `--port` to specify a different port number, for instance:

```
admin-docker-agent --port 9999
```

To generate a Polyspace Access URL that does not require specifying a port number, for instance `https://accessServerHost.com` instead of `https://accessServerHost:9443.com`, use the `--port` flag and select port 443 for HTTPS or port 80 for HTTP.

To reset the password, press **CTRL+C** to stop the `admin-docker-agent` binary and enter this command:

```
admin-docker-agent --reset-password
```

To view the new password, restart the binary.

The **Cluster Admin** agent creates a `settings.json` file the first time it starts, and stores this file in the same folder as the `admin-docker-agent` binary by default. Ensure that only the user who starts the `admin-docker-agent` has read/write permissions on the `settings.json` file.

Choose Between HTTP and HTTPS Configuration for Polyspace Access

HTTP Configuration for Polyspace Access

By default, the **Cluster Admin** uses the HTTP protocol. When you start the `admin-docker-agent` binary, you do not need to specify any additional flags. The communications between Polyspace Access and client machines are not encrypted.

HTTPS Configuration for Polyspace Access

To encrypt the data between Polyspace Access and client machines, configure the **Cluster Admin** with the HTTPS protocol.

To complete the configuration:

- 1 Generate a private key and a certificate signing request (CSR) with a subject alternative name (SAN) extension.
- 2 Submit the CSR to your organization's certificate authority (CA) to obtain a signed X.509 certificate. Alternatively, use the private key and CSR to create a self-signed certificate.

- 3 Start the `admin-docker-agent` binary and use these options to pass the private key and certificate files. All the files must be in PEM format.

Option	Option Parameter
<code>--ssl-cert-file</code>	Path of certificate signed by CA or self-signed certificate file.
<code>--ssl-key-file</code>	Path of private key file.
<code>--ssl-ca-file</code>	Path of CA trust store file or self-signed certificate file.

It is recommended that you use a certificate issued by a certificate authority to configure HTTPS. The use of self-signed certificates is not recommended outside of testing environments as most web browser cannot establish the authenticity of a server that uses a self-signed certificate and consider the connection to that server not secure. Before you configure Polyspace Access for HTTPS in a production environment using a self-signed certificate, contact your network security administrator.

The configuration of HTTPS for the **Cluster Admin** enables HTTPS for the API Gateway service. This service handles all communications between the other Polyspace Access services and client machines. To configure HTTPS for communications between the different Polyspace Access services, see “Enable HTTPS Between Polyspace Access Services” on page 4-17.

1. Generate Private Key and Certificate Signing Request

These steps illustrate how to generate a private key and CSR configured with a SAN extension on a Debian Linux system by using the `openssl` utility and an `openssl` configuration file.

- 1 Copy this configuration file to a text editor and save it on your machine as `openssl.cnf`.

Configuration file

```
[ req ]
req_extensions = v3_req
distinguished_name = req_distinguished_name
prompt = no

[ x509 ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName = @alt_names

[ req_distinguished_name ]
countryName = US
stateOrProvinceName = myState
organizationName = myCompany
organizationalUnitName = myUnit
emailAddress = user@email.com
commonName = hostName

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = hostName
```

hostName is the fully qualified domain name (FQDN) of the server machine on which you run the **API Gateway** service. *hostName* must match the FQDN that you specify when you start the

`admin-docker-agent` binary. You can obtain `hostName` by running the command `hostname -f`. Editing the other fields in the `[req_distinguished_name]` section does not affect the configuration.

To configure HTTPS for multiple hostnames that resolve to the same IP address, add each hostname (DNS or IP address) on a new line in the `[alt_names]` section of the `openssl` configuration file. For instance:

```
[ alt_names ]
DNS.1 = server-machine.example.com
DNS.2 = server-machine
IP.1 = 192.168.0.1
```

- 2 Generate a CSR by using your `openssl.cnf` configuration file. In a command line terminal, enter:

```
openssl req -new -out myRequest.csr -newkey rsa:4096 -keyout myKey.pem -nodes -config openssl.cnf
```

The command outputs a private key file `myKey.pem` and the file `myRequest.csr`, which contains a public key and data that describes your server.

Do not reuse the private key file that you use in the **User Manager** configuration for the **Authentication private key file** field. See “Configure User Manager” on page 4-21.

Secure your private key by following best practices such as:

- Do not transfer the private key between machines. Instead, generate and store the private key on a local file system.
- Restrict read/write permissions. Grant access to the private key file only to the **Cluster Admin** administrators.
- Rotate your private key and certificate regularly (annually) and audit which users have access to the private key file.

2. Obtain and Verify CA Certificate or Self-Signed Certificate

Submit the CSR file `myRequest.csr` to your organization's certificate authority. The certificate authority uses the file to generate a signed server certificate. For instance, `admin_cert.cer`.

Alternatively, you can generate a self-signed certificate by running this command and using the private key, CSR, and `openssl` configuration files:

```
openssl x509 -req -days 365 -in myRequest.csr -signkey myKey.pem -out self_cert.cer -extensions x509 -extfile openssl.cnf
```

The command outputs a self-signed certificate `self_cert.cer`.

After you obtain your CA certificate or you generate a self-signed certificate, you can use the `openssl` utility to verify the content of the certificate. For instance, To check that the DNS is listed correctly in the subject alternative name section of your self-signed certificate, run this command:

```
openssl x509 -text -noout -in self_cert.cer | grep -A 1 "Subject Alternative Name"
```

The command output the line that matches "Subject Alternative Name" and the next line, which lists the DNS that is associated with the certificate.

3. Start the Admin Docker Agent

After you obtain a CA certificate or you generate a self-signed certificate, start the `admin-docker-agent` by running the corresponding command.

CA Certificate	Self-Signed Certificate
<pre>./admin-docker-agent --hostname <i>hostName</i>\ --ssl-cert-file <i>absolutePathTo/admin_cert.cer</i> \ --ssl-key-file <i>absolutePathTo/myKey.key</i> \ --ssl-ca-file /etc/ssl/certs/ca-certificates.crt</pre> <p><code>ca-certificates.crt</code> is the certificate trust store file on Debian Linux systems. When you open the Cluster Admin web interface, your browser considers the connection secure if the browser uses the certificate trust store that you specify for <code>--ssl-ca-file</code>.</p>	<pre>./admin-docker-agent --hostname <i>hostName</i>\ --ssl-cert-file <i>absolutePathTo/self_cert.cer</i> \ --ssl-key-file <i>absolutePathTo/myKey.key</i> \ --ssl-ca-file <i>absolutePathTo/self_cert.cer</i></pre> <p>The self-signed <code>certificate.pem</code> file is also used as the certificate trust store file. Because you do not specify a trust store associated with a certificate authority, when you open the Cluster Admin web interface, your browser might show a warning about the certificate being untrusted.</p>

The *hostName* that you specify in this command must match the *hostName* you specified in `opensl` configuration file. *absolutePathTo* is the absolute file path.

Enable HTTPS Between Polyspace Access Services

To enable HTTPS for communications between the Polyspace Access services, after you “Open the Cluster Admin Interface” on page 4-17, click **Configure Nodes** on the **Cluster Dashboard** and select **Enable SSL** on the **General** tab. Enabling SSL in the **Nodes** settings affects communications only between the Polyspace Access services that are installed on that node. The SSL certificate, private key, and CA files that you provide when you start the `admin-docker-agent` binary are reused in the **Nodes** settings, unless the node is already configured with a different set of files.

By default, all services are installed on the same node and the services ports are not exposed. You do not need to enable HTTPS for the **User Manager**, **Issue Tracker**, and **Polyspace Access** services unless you install these services on different nodes, or you start the `admin-docker-agent` binary with option `--force-exposing-ports`.

If you install all the services on the same node and you start the `admin-docker-agent` binary without using option `--force-exposing-ports`, and then you select **Enable SSL** in the **Nodes** setting, you see warning messages about the **Certificate File** and the **Certificate Key File** not being used because the ports are not exposed. If you do not plan on exposing the services ports, you can ignore the warnings and proceed with the next steps in your installation.

Open the Cluster Admin Interface

After you configure and start the **Cluster Admin**, open your web browser and go to URL specified in the command-line output when you started the `admin-docker-agent` binary.

Log in with the initial password that you obtained when you started the **Cluster Admin** agent. If this is your first time logging in, follow the prompts.

Cluster Admin Account ▾

Cluster Dashboard

Restart Apps Delete Apps

App	Status
User Manager Manage users	● Not installed
Issue Tracker	● Not installed
Polyspace Access Open UI	● Not installed

I want to ...

- [Configure Apps](#)
- [Configure Nodes](#)

It is best practice to change your **Cluster Admin** password after your first login. To set a new password, click **Account** in the upper right corner of the web interface and select **Change password**. Share the **Cluster Admin** password only with users who configure and manage the Polyspace Access services.

On the **Cluster Dashboard**, click **Configure Apps** to open the **Cluster Settings**. On this page, you can:

- Configure user authentications to use the user credentials from your company LDAP or by using custom credentials. See “Configure User Manager” on page 4-21.
- Integrate bug tracking tools such as Jira Software and Redmine with Polyspace Access. See “Configure Issue Tracker” on page 4-29.
- Specify the paths of the license file and of the different folders used by the Polyspace Access database, ETL, and web server . See “Configure Polyspace Access App Services” on page 4-33.

Once you fill all settings, click **Save**, return to the **Cluster Dashboard**, and click **Restart Apps** for the changes to take effect. Make sure that **Validate on Save** is enabled before you click **Save**.

To save partially filled settings, clear **Validate on Save** and click **Save**.

Cluster Admin

Account ▾

Cluster Settings

[Return to Dashboard](#)

User Manager

Use internal directory

Internal directory database volume

localP20226-BADHACCESS05jul15-2022\local_dbfs

Internal directory database username

um

Internal directory database password

.....

Administrator sign-in IDs

admin

Initial administrator password

....

Authentication token expiration (sec)

86400

Authentication private key file

localP20226-BADHACCESS05jul15-2022\local_dbfs

API keys and user IDs

5ea34345-a03b-4a20-821e-f10e45e0e863,jsmith

Polyspace Access Web Server

Upload directory

localP20226-BADHACCESS05jul15-2022\local_dbfs\polyspace

Temporary upload directory

localP20226-BADHACCESS05jul15-2022\local_dbfs\polyspace

License file

localP20226-BADHACCESS05jul15-2022\local_dbfs\polyspace\license

[Validate Now](#) Validate on Save[Save](#)[Cancel](#)

Note On Windows systems, all the file paths that you specify must point to local drives.

See Also

admin-docker-agent

More About

- “Prepare Your Installation” on page 4-9
- “Configure User Manager” on page 4-21
- “Configure Issue Tracker” on page 4-29
- “Configure Polyspace Access App Services” on page 4-33

Configure User Manager

The **User Manager** manages the authentication of the Polyspace Access user logins. The logins are authenticated by checking the usernames and passwords against the credentials of identities that you store in the **User Manager** database. To add identities to the **User Manager** database, do one of the following.

- Connect your company Lightweight Directory Access Protocol (LDAP) server to the **User Manager** and import identities from the LDAP server. To configure the connection with the LDAP server, see “Connect Your Organization LDAP Server to the User Manager” on page 4-24.
- Create custom identities. See “Manage Users and Groups” on page 4-36. Create custom identities if, for instance, you cannot or choose not to use your company LDAP.

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, click **Save**, return to the **Cluster Dashboard**, and click **Restart Apps** for the changes to take effect. Make sure that **Validate on Save** is enabled before you click **Save**.

Note On Windows systems, all the file paths that you specify must point to local drives.

General User Manager Settings

These settings are required whether you create custom user credentials or you import users from your company LDAP server.

Setting	Description
Internal directory database volume	Specify the full path to the database folder. The database stores information about users and groups that you import from the LDAP server or that you create in the User Manager interface. By default, the database is stored in the folder where you extracted the Polyspace Access installation image, under appdata/usermanager. For instance, /local/usr/Access/R2023a/appdata/usermanager/db.
Internal directory database username	Use the username that you specify in this field if you need to interact with the internal database by using PostgreSQL commands. The default username is "UM".

Setting	Description
Internal directory database password	<p>Use the password that you specify in this field if you need to interact with the internal database by using PostgreSQL commands.</p> <p>You must provide a password.</p> <p>To see the current database password, run this command.</p> <ul style="list-style-type: none"> • Windows <pre>docker inspect usermanager-db-0-main FIND "UMDB_PASSWORD"</pre> <ul style="list-style-type: none"> • Linux <pre>docker inspect usermanager-db-0-main grep UMDB_PASSWORD</pre>
Administrator sign-in IDs	<p>Enter a comma-separated list of usernames to set those users as User Manager administrators. You can specify custom usernames or usernames from your company LDAP. See “Create, Edit, or Remove Users and Groups” on page 4-37.</p> <hr/> <p>Note If you enable Connect an LDAP directory, at least one of the usernames that you specify in this field must be from the LDAP directory or must already exist in the User Manager database. Otherwise, you cannot log into the User Manager interface.</p> <hr/> <p>The users that you specify in this field are also Polyspace Access administrators. For more information about Polyspace Access administrators, see “Manage Project Permissions”.</p> <p>To remove a user as a Polyspace Access administrator:</p> <ol style="list-style-type: none"> 1 Remove the username from this field, save your changes, then restart the apps. 2 After the restart, a Polyspace Access administrator must unassign the user from all top-level folders in the Project Explorer, in the Polyspace Access web interface, by using the context menu. The administrator can also perform this task at the command line by using the <code>-unset-role</code> flag with the <code>polyspace-access</code> binary. For more information, see <code>polyspace-access</code>.
Initial administrator password	<p>Password that you use to log into the User Manager interface.</p> <p>This field is not available if you select Connect an LDAP directory in the User Manager Directory Connection settings.</p>

Setting	Description
Authentication token expiration (sec):	<p>Specify in seconds the period of validity of the signed JSON Web Tokens that the User Manager issues to authenticated users. This expiration time determines the lifetime of a session. Once you log into Polyspace Access, your license is checked out and your session refreshes periodically to keep it from expiring. The session ends once you explicitly log out or close your web browser and your license is checked back in. If your browser closes unexpectedly, your license stays checked out until the session expires.</p> <p>When you set the expiration time, consider:</p> <ul style="list-style-type: none"> • If the expiration time is too short, frequent users are prompted to log back in frequently. On large teams, the license server experiences a high volume of license checkins and checkouts. • If the expiration time is too long, the session time of less frequent users might be overestimated in the license logs. <p>Use this setting to set the licensing timeout. Polyspace Access ignores the license timeout value that you set through the license manager options file (MLM.opt) by using the <code>TIMEOUT feature seconds</code> syntax.</p>
Authentication private key file:	<p>Specify the full path to the private key PEM file that the User Manager uses to sign JSON Web Tokens. On Windows systems, the paths must point to local drives.</p> <p>The User Manager service does not support password-protected private keys. You can generate a private key by using the <code>openssl</code> utility. For example:</p> <pre>openssl genrsa -out private.pem 2048</pre> <p>Restrict access to this private key to only those administrators who manage the User Manager service.</p> <p>Do not reuse the private key that you use to generate the SSL certificates, which you provide if you enable the HTTPS protocol.</p>

Setting	Description
API keys and user IDs	<p>Enter an API key value and username pair in this field to assign an API key to a user. For example, to assign an API key to user <code>jsmith</code>, enter:</p> <pre>5ea34345-a03b-4a20-821e-f10e45e0e863,jsmith</pre> <p>To assign API keys to other users, enter additional API key and username pairs on separate lines. Each API key value must be unique.</p> <p>You can assign any combination of alphanumeric characters as an API key to a user. For example, to generate universally unique identifiers (UUID) for the API key, use these commands:</p> <ul style="list-style-type: none"> • Windows PowerShell: <code>[guid]::NewGuid()</code> • Linux: <code>uuidgen</code> <p>Use the API key with these commands that require Polyspace Access credentials:</p> <ul style="list-style-type: none"> • <code>polyspace-access</code> • <code>polyspace-results-export</code> • <code>polyspace-report-generator</code> <p>When using the API key, it is recommended that you store the API key in a text file and pass that file to the command by using <code>-credentials-file</code>. For example, to use the API key for user <code>jsmith</code>, store this line in text file <code>credentials.txt</code>:</p> <pre>-api-key 5ea34345-a03b-4a20-821e-f10e45e0e863</pre> <p>and then pass the file to the command, for instance:</p> <pre>polyspace-access -credentials-file credentials.txt</pre> <p>Alternatively, pass the API key directly at the command line by using <code>-api-key</code>.</p> <p>The commands use the API key as a login credential for the corresponding user. If a user updates his or her password, you do not have to update the API key. If you use the API key as part of an automation script, make sure that the user associated with the key has enough permissions to perform all the operations in the script. See “Manage Project Permissions”.</p>

To create or manage identities, see “Manage Users and Groups” on page 4-36.

Connect Your Organization LDAP Server to the User Manager

To use the LDAP server of your organization, in the **User Manager Directory Connection** settings, select **Connect an LDAP directory** and configure the LDAP settings. Contact your LDAP administrator to obtain the LDAP URL, LDAP user base, other LDAP settings and filters, and to determine whether your LDAP server uses the Active Directory Global Catalog feature.

Setting	Description
LDAP URL	<p>Enter the LDAP URL as:</p> <p><code>ldap://HOST:PORT</code></p> <p><i>HOST</i> is the LDAP host and <i>PORT</i> is the LDAP port number. The LDAP server uses different default port numbers if you configure it to use the global catalog. See “Configure User Manager for LDAP Server That Uses Global Catalog” on page 4-27.</p> <p>If you have configured your LDAP server over SSL, enter the URL as:</p> <p><code>ldaps://HOST:PORT</code></p> <p>For additional LDAPS configuration steps, see “Configure the User Manager for LDAP over SSL” on page 4-27.</p> <p>Because communications between the LDAP server and clients are not encrypted, the configuration and use of LDAP over SSL (LDAPS) is recommended.</p>
Synchronization interval (seconds)	<p>Specify in seconds the interval between synchronizations of the User Manager database with the LDAP server. For instance, specify 1800 to synchronize the User Manager database with the LDAP server every half hour.</p> <p>To synchronize manually, click Synch With LDAP in the User Manager interface.</p>
LDAP username	<p>username of user who has read permission to the LDAP server. Leave this field blank if your access to the LDAP server is not password-protected.</p>
LDAP password	<p>Password of user who has read permission to the LDAP server. Leave this field blank if your access to the LDAP server is not password-protected.</p> <p>The password is stored in the <code>settings.json</code> file. For added security, set restrictions on the read and write permissions for this file. By default, this file is stored in the same folder as the <code>admin-docker-agent</code> binary.</p>
Enable LDAP pagination	<p>Enable this setting to control the rate at which your LDAP server returns results. You typically use this setting if you query a large set of users and the LDAP server has limits on the number of entries that it returns, or if the client making the query has limited resources.</p> <p>Before you enable this setting, check that:</p> <ul style="list-style-type: none"> • Your LDAP server supports simple paged results control. See LDAP Control Extension for Simple Paged Results Manipulation. • If you provide an LDAP username and LDAP password, pagination is not disabled for these credentials. The User Manager log shows LDAP result code 11 when pagination is disabled for the credentials that you provide. • The LDAP server is configured to handle a reasonable number of simultaneous paged queries. The User Manager log shows LDAP result code 53 if the number of simultaneous queries exceeds the LDAP server limit.

Setting	Description
LDAP page size	Set the number of results that the LDAP server returns per page, for example 1000. This setting is available only if you select Enable LDAP pagination .
LDAP user base	You can retrieve this parameter by using an LDAP explorer tool. For instance, connect to your LDAP server through Apache Directory Studio and open the properties for your connection. In the Browser Options , click Fetch Base DNs to get the LDAP base.
LDAP user search filter	<p>Use the search filter to retrieve a subset of users from the LDAP database. The User Manager loads this subset on startup instead of loading all users in your organization. Loading a smaller number of users for authentication improves the performance of Polyspace Access.</p> <p>Specify the search filter as <i>attribute=value</i>, for instance <code>CN=test*</code> matches all users who have a common name (CN) attribute that starts with "test".</p> <p>Use parentheses to combine multiple filter expressions in an AND (&) or OR () clause. For instance, <code>((CN=jdoe) (department=foo))</code> matches all users who have CN attribute "jdoe" or department attribute "foo".</p> <p>The default search filter is <code>objectClass=organizationalPerson</code>. For more information about search filters, see the LDAP filters.</p> <p>To check whether a search filter is returning a subset of users, enter a username from that subset in the API keys and user IDs field along with a dummy key, and click Validate Now at the bottom of the page. You get a warning if the username cannot be found. For instance, to check if the filter returns <code>jsmith</code>, enter <code>1234,jsmith</code>.</p>
LDAP user ID attribute LDAP user display name attribute LDAP user email attribute LDAP user image attribute	Leave these settings unchanged unless instructed otherwise by your LDAP administrator. Polyspace Access does not use the LDAP email and image attributes.
Enable LDAP groups	Enable this setting to import user groups from the LDAP server directory.
LDAP group base	You can retrieve this parameter by using an LDAP explorer tool. For instance, connect to your LDAP server through Apache Directory Studio and search for a group that you want to import, then open the properties for that group.

Setting	Description
LDAP group search filter	<p>Use the search filter to retrieve a subset of groups from the LDAP database. The User Manager loads this subset on startup instead of loading all the groups in your organization. Loading a smaller number of groups improves the performance of Polyspace Access.</p> <p>Specify the search filter as <i>attribute=value</i>, for instance <code>CN=test*</code> matches all groups who have a common name (CN) attribute that starts with "test".</p> <p>Use parentheses to combine multiple filter expressions in an AND (&) or OR () clause. For instance, <code>((CN=Printers)(location=US))</code> matches all groups that have CN attribute "Printers" or department attribute "US".</p> <p>For more information about search filters, see the LDAP filters.</p>
LDAP group ID attribute LDAP group display name attribute LDAP group membership attribute	<p>Leave these settings unchanged unless instructed otherwise by your LDAP administrator. Polyspace Access does not use the LDAP display name attribute.</p>

Configure the User Manager for LDAP over SSL

If you use an LDAP server configured over SSL (LDAPS), add the LDAPS SSL certificate to the certificate trust store file that you specify in the **CA File** field of the **Nodes** settings. To view these settings, click **Configure Nodes** on the **Cluster Dashboard**. Depending on your trust store file, the LDAP SSL certificate might already be included in the trust store.

The certificate trust store file typically corresponds to the file that you provide with `--ssl-ca-file` when you configure the **Cluster Admin** with HTTPS. See "Choose Between HTTP and HTTPS Configuration for Polyspace Access" on page 4-14.

For instance, on a Linux Debian distribution, to add LDAP certificate `ldaps_cert.pem` to trust store file `trust_store.pem`, use this command:

```
cat trust_store.pem ldaps_cert.pem > combined_cert.pem
```

The command combines the content of the two files and outputs file `combined_cert.pem`. If you use a self-signed certificate to configure HTTPS, add the LDAP certificate to the self-signed certificate.

To complete the configuration, enter the path to `combined_cert.pem` in the **CA File** field of the **Nodes** settings, save your changes, return to the dashboard, and restart the Apps.

If you did not configure the **Cluster Admin** by using HTTPS, specify the path to the LDAP SSL certificate in the **CA File** field.

Configure User Manager for LDAP Server That Uses Global Catalog

The global catalog (GC) is a mechanism that enables you to add users from different Active Directory® (AD) servers to Polyspace Access without having to provide information about those

servers. For more information, see Global Catalog. If your LDAP server is configured to use the GC, you must specify a GC-specific port number when you provide the LDAP URL to the **User Manager** service. The default port is 3268 or, if you use secure LDAP (LDAPS), 3269. To determine whether your LDAP server is configured to use the GC, contact your LDAP administrator.

If you specify an incorrect port number, the **User Manager** service cannot communicate with the LDAP server. When you inspect the **User Manager** log, you get error messages similar to these error messages.

```
LDAP server 'ldap://server01.example.com:389' did not recognize
base DN 'DC=example,DC=com' and search base ''.
```

```
...
Unprocessed Continuation Reference(s)
```

To save the **User Manager** log to a file `out.log`, use this command.

```
docker logs -t usermanager-server-0-main >> out.log 2>&1
```

Note If you run Polyspace Access version R2021b or earlier, the docker container names might be different. To view the names of currently running containers, use command `docker ps --format '{{.Names}}'`.

The GC holds only a subset of attributes for each user from the different Active Directory (AD) servers. The LDAP ID attribute that you specify in the cluster operator settings must be available in this subset of attributes when the GC is enabled. If the LDAP ID is not available in the GC, the corresponding user is not added to Polyspace Access.

See Also

More About

- “Prepare Your Installation” on page 4-9
- “Configure and Start the Cluster Admin” on page 4-13
- “Configure Issue Tracker” on page 4-29
- “Configure Polyspace Access App Services” on page 4-33
- “Register Polyspace Desktop User Interface” on page 4-47
- “Upload Examples and Open Polyspace Access Interface” on page 4-44

Configure Issue Tracker

Configure the **Issue Tracker** if you want to enable the creation of tickets in your bug tracking tool (BTT) from the Polyspace Access interface. Polyspace Access supports integration with the Jira Software and Redmine BTT. If you do not want to integrate your BTT with Polyspace Access, select none in the **Provider** field.

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, click **Save**, return to the **Cluster Dashboard**, and click **Restart Apps** for the changes to take effect. Make sure that **Validate on Save** is enabled before you click **Save**.

Note On Windows systems, all the file paths that you specify must point to local drives.

Configure Jira Software Bug Tracking Tool

Setting	Description
Provider	Jira
Jira deployment type	Specify whether your Jira instance is hosted on a local server or on a cloud service provider. This field is available only if you select Jira as a provider.
Jira URL	Specify the URL of the Jira instance for your organization, for example, <code>https://jira.mycompany.com</code> . If your Jira instance is configured by using HTTPS, see “Add BTT Instance Configured by Using HTTPS” on page 4-31. This field is available only if you select Jira as a provider.
Authentication method	Specify the method that the Issue Tracker uses to authenticate logins to Jira from Polyspace Access users. This field is available only if you select Jira as a provider.

Setting	Description
OAuth1 callback URL	<p>If you select OAuth as an authentication method, your Jira administrator must first create an application link in Jira. The Jira administrator specifies an application URL (the URL for Polyspace Access) and generates an RSA public/private keypair. For more information, see step 1 on this page.</p> <p>The OAuth1 callback URL must match the application URL specified in Jira, for instance <code>https://access-machine.company.com:9443</code>.</p> <div data-bbox="516 541 1466 1081" style="border: 1px solid black; padding: 10px; margin: 10px 0;"> <p style="text-align: center;">Application URL: <code>https://access-machine.company.com:9443</code></p> <p>Consumer Key* <input type="text" value="OauthKey"/></p> <p>Consumer Name* <input type="text" value="Polyspace Access"/></p> <p>Public Key* <pre>-----BEGIN PUBLIC KEY----- MIGfMA0GCSqGSIb3DQEBAQUAA4 GNADCBiQKBgQCywJ7lq22Dg9vZP</pre></p> <p style="text-align: right;"><input type="button" value="Continue"/> <input type="button" value="Cancel"/></p> </div> <p>This field is available only if you select Jira as a provider.</p>
OAuth1 consumer key	<p>Specify the consumer key value that your Jira administrator entered when configuring the application link in Jira, for instance <code>OauthKey</code>.</p> <p>This field is available only if you select Jira as a provider.</p>
OAuth1 private key file	<p>Specify the path to the private key file that your Jira administrator generated when configuring the application link in Jira, for instance <code>/local/polyspace_access/jira_privatekey.pem</code>.</p> <p>Note The hostname that you specify when you create the JIRA private key must match the hostname that you use in your Polyspace Access URL. If the hostnames do not match, for instance if you use <code>localhost</code> instead of the hostname in your Polyspace Access URL, you might see a user authentication error when you attempt to create Jira tickets from Polyspace Access.</p> <p>This field is available only if you select Jira as a provider.</p>

Limitations

- Polyspace Access does not support the creation of BTT tickets that have custom fields if these fields are required fields, except if the fields are:
 - All numeric values.

- String only values.
- Single select custom fields.
- After users log into Jira from Polyspace Access and start creating Jira tickets, the users remain logged into their Jira session until the session expires.
- In Jira Software version 8.4 and later, do not enable dark feature. See Enable Dark Feature in Jira.

Configure Redmine Bug Tracking Tool

Setting	Description
Provider	Redmine
Redmine URL	<p>Specify the URL of the Redmine instance for your organization, for example, <code>https://redmine.mycompany.com</code>.</p> <p>If your Redmine instance is configured by using HTTPS, see “Add BTT Instance Configured by Using HTTPS” on page 4-31.</p> <p>This field is available only if you select Redmine as a provider.</p>
Redmine API key	<p>Specify the API access key of a Redmine administrator.</p> <p>To obtain the API key, log into your instance of Redmine as an administrator, click My account in the upper-right corner, then, in the right pane, click Show under API access key.</p> <p>The Issue Tracker does not validate the API key. Check periodically that the API key has not expired or become invalid.</p> <p>This field is available only if you select Redmine as a provider.</p>

Limitations

- Polyspace Access does not support the creation of BTT tickets that have custom fields if these fields are required fields, except if the fields are all numeric values or string only values.
- To create a redmine ticket from Polyspace Access, the username used to log into Polyspace Access must match the username of a Redmine account.
- Redmine tickets that users create from Polyspace Access can be populated only with default field values. Some of the ticket field values that a user selects in Polyspace Access might not match the field values of the Redmine ticket.

Add BTT Instance Configured by Using HTTPS

If your BTT instance is configured by using HTTPS, add the BTT SSL certificate to the certificate trust store file that you specify in the **CA File** field of the **Nodes** settings. To view these settings, click **Configure Nodes** on the **Cluster Dashboard**. Depending on your trust store file, the BTT SSL certificate might already be included in the trust store.

The certificate trust store file typically corresponds to the file that you provide with `--ssl-ca-file` when you configure the **Cluster Admin** with HTTPS. See “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14.

For instance, on a Linux Debian distribution, to add BTT certificate `bttts_cert.pem` to trust store file `trust_store.pem`, use this command:

```
cat trust_store.pem bttts_cert.pem > combined_cert.pem
```

The command combines the content of the two files and outputs file `combined_cert.pem`. If you use a self-signed certificate to configure HTTPS, add the BTT certificate to the self-signed certificate.

To complete the configuration, enter the path to `combined_cert.pem` in the **CA File** field of the **Nodes** settings, save your changes, return to the dashboard, and restart the Apps.

If you did not configure the **Cluster Admin** by using HTTPS, specify the path to the BTT SSL certificate in the **CA File** field.

See Also

More About

- “Prepare Your Installation” on page 4-9
- “Configure and Start the Cluster Admin” on page 4-13
- “Configure User Manager” on page 4-21
- “Configure Polyspace Access App Services” on page 4-33
- “Register Polyspace Desktop User Interface” on page 4-47
- “Upload Examples and Open Polyspace Access Interface” on page 4-44

Configure Polyspace Access App Services

Specify the path of your license file, and the paths of the folders that Polyspace Access uses for its database and for various working directories. Polyspace Access uses these directories to store and process results that you upload to the database.

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, click **Save**, return to the **Cluster Dashboard**, and click **Restart Apps** for the changes to take effect. Make sure that **Validate on Save** is enabled before you click **Save**.

Polyspace Access Database

Setting	Description
Data volume	<p>Specify the full path to the folder where you store the database.</p> <p>By default, the database is stored under <code>appdata/polyspace-access</code> in the folder where you extracted the Polyspace Access installation image.</p> <p>Deleting the Polyspace Access Database service and uninstalling Polyspace Access does not erase the results that you uploaded to the database from the data volume.</p> <p>To delete a data volume and its content, manually delete the folder where you store the database.</p>
Password	<p>Specify a password to authenticate connections to the database. The different Polyspace Access services use this password when they communicate with the database. Use this password if you are prompted for one while performing a database backup on page 4-51 or clean up on page 4-56.</p> <p>You can specify a password only for a new database. To change the password of an existing database, use the PostgreSQL utilities instead. If you update the password of the database by using the PostgreSQL utilities, you must update the password in this field as well.</p> <p>You must provide a password.</p> <p>To see the current database password, run this command.</p> <ul style="list-style-type: none"> • Windows <pre>docker inspect polyspace-access-db-0-main FIND "POSTGRES_PASSWORD"</pre> • Linux <pre>docker inspect polyspace-access-db-0-main grep POSTGRES_PASSWORD</pre>

Polyspace Access ETL

Setting	Description
Storage directory	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See “Storage and Port Configuration” on page 4-4.
Invalid results directory	
Working directory	By default, these folders are stored under <code>appdata/polyspace-access</code> in the folder where you extracted the Polyspace Access installation image.
Upload directory	The Upload directory path must be the same for the Polyspace Access Web Server and Polyspace Access ETL services.

Polyspace Access Web Server

Setting	Description
Upload directory:	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See “Storage and Port Configuration” on page 4-4.
Temporary upload directory:	
	By default, these folders are stored under <code>appdata/polyspace-access</code> in the folder where you extracted the Polyspace Access installation image.
	The Upload directory path must be the same for the Polyspace Access Web Server and Polyspace Access ETL services.
License file:	Specify the full path to the <code>network.lic</code> license file that you created when you configured the Polyspace Access license. See “Configure Polyspace Access License” on page 6-2. On Windows systems, the paths must point to local drives.

See Also

More About

- “Prepare Your Installation” on page 4-9
- “Configure and Start the Cluster Admin” on page 4-13
- “Configure User Manager” on page 4-21
- “Configure Issue Tracker” on page 4-29
- “Register Polyspace Desktop User Interface” on page 4-47
- “Upload Examples and Open Polyspace Access Interface” on page 4-44

Start Polyspace Access and Manage Users

After you complete the configuration of the **User Manager**, **Issue Tracker**, and **Polyspace Access** apps, save your settings, return to the **Cluster Dashboard**, and click **Restart Apps**.

Cluster Admin
Account ▾

Cluster Dashboard

Restart Apps
Delete Apps

App	Status
User Manager 🔗 Manage users	● Running
Issue Tracker	● Running
Polyspace Access 🔗 Open UI	● Running

I want to ...

- [Configure Apps](#)
- [Configure Nodes](#)

When an app starts, the indicator in the **Status** column turns green. The **Polyspace Access Web Server** service might take a few moments to start.

If one or more of the apps start and stop after a short time, click the app in the **Cluster Dashboard**, and then click **Show Logs**.

Cluster Admin
Account ▾

Polyspace Access Return to Dashboard

Container	Status	
Polyspace Access DB	● Running	Show Logs
Polyspace Access ETL	● Running	Show Logs
Polyspace Access Web Server	● Running	Show Logs

To identify the cause of the service stoppage, use the output log. If you need additional assistance, see “Contact Technical Support About Polyspace Access Issues”.

Configure Polyspace Access to Restart Automatically

Update the Docker restart policy to configure Polyspace Access for automatic restarts after an unexpected system shutdown or a reboot.

To update the Docker restart policy, in Linux, use this command:

```
docker update --restart always \  
admin \  
gateway \  
polyspace-access-web-server-0-main \  
polyspace-access-etl-0-main \  
polyspace-access-db-0-main \  
polyspace-access-download-0-main \  
issuetracker-server-0-main \  
issuetracker-ui-0-main \  
usermanager-server-0-main \  
usermanager-ui-0-main \  
usermanager-db-0-main \  
polyspace-access \  
issuetracker \  
usermanager \  

```

To use this command in Windows PowerShell, replace the backslash "\" characters with a backtick "`". The Docker daemon will always attempt to restart the specified Polyspace Access containers when they stop. If you stop a container manually, the container restarts only when the Docker daemon restarts or when you restart the container manually.

To prevent a restart loop, the restart policy applies only to containers that have started successfully. For more details, see the Docker documentation.

Note If you run Polyspace Access version R2021b or earlier, the docker container names might be different. To view the names of currently running containers, use command `docker ps --format '{{.Names}}`'.

Manage Users and Groups

Once the **User Manager** service is running, click **Manage users** to open the **User Manager** interface. The interface shows all the users and groups that are stored in the **User Manager** database.

- To create, remove, or edit users and groups, see “Create, Edit, or Remove Users and Groups” on page 4-37.
- To filter users and groups, see “Sort Users and Groups in the User Manager Interface” on page 4-41.

You might be required to log into the UI. Use the login credentials of one of the users that you specified in the **Administrator sign-in IDs** field. Only these users can open the **User Manager** interface and manage users.

User Manager
admin ▾

Dashboard

Sync With LDAP
Create ▾

Filter ▾

Showing 1-19 of 19

User
Group

ID	Type	Display Name	Source	Actions
admin ADMIN	user		internal	✕ ▾
alovela	user	Ada Lovelace	internal	✕ ▾
api_team	group	Group backend API	internal	✕ ▾
board	group		ldap	✕ ▾
dev	group		ldap	✕ ▾
dev_manager	group	Dev Managers	internal	✕ ▾
ghopper	user	Grace Hopper	internal	✕ ▾
Jack Barker	user	Action Jack	ldap	✕ ▾
jdoe	user	Jane Doe	ldap	✕ ▾
jsmith	user	John Smith	internal	✕ ▾
parent	group		ldap	✕ ▾

Create, Edit, or Remove Users and Groups

To create, edit, or remove custom users and groups and to view information about LDAP users and groups, such as group memberships, use the **User Manager** interface . When you connect the **User Manager** to an LDAP server, the **User Manager** copies to its database the user and group entries returned by the LDAP server based on your LDAP filters. The **User Manager** does not make any changes to the entries on the LDAP server.

This table describes the different operations that you can perform on users and groups (identities), depending on whether the identities are custom entries in the **User Manager** database or whether the identities are from an LDAP server.

Operation	Custom Identity	LDAP Identity
<p>Create User Manager Administrator</p>	<ul style="list-style-type: none"> • If the User Manager is not connected to an LDAP server, the custom users that you specify in the Administrator sign-in IDs field are automatically created in the User Manager database and the users are assigned the password that you specified in the Initial administrator password field. • If the User Manager is connected to an LDAP server, log into the User Manager interface with valid administrator credentials and create the custom users that you specified in the Administrator sign-in IDs field. • You see an ADMIN label next to the administrator ID in the interface. 	<ul style="list-style-type: none"> • The LDAP users that you specify in the Administrator sign-in IDs field are automatically copied to the User Manager database. • You see an ADMIN label next to the administrator ID in the interface.
<p>Remove User Manager administrator</p>	<ul style="list-style-type: none"> • To remove a User Manager administrator, remove the username from the Administrator sign-in IDs field, save your changes, then restart the apps. The ADMIN label no longer appears next to the user ID in the User Manager interface. <p>To remove the user completely from the database, after the restart, an administrator must log into the User Manager interface to delete that user.</p> <ul style="list-style-type: none"> • If the administrator is a custom user that overrides a duplicate LDAP user, click the icon in the Actions column and select Delete. The custom identity is removed from the database and the administrator role is transferred to the LDAP user. 	<p>To remove a User Manager administrator, remove the username from the Administrator sign-in IDs field, save your changes, then restart the apps. The ADMIN label no longer appears next to the user ID in the User Manager interface.</p> <p>To remove the user completely from the database, contact your LDAP administrator.</p>

Operation	Custom Identity	LDAP Identity
Create user	<ul style="list-style-type: none"> • To create a custom user, click Create > User and then specify a user ID, for instance <code>jsmith</code>, and a password. For Polyspace as You Code users, this custom user ID must match the username of the machine where Polyspace as You Code runs. <p>Optionally, you can specify a display name, for instance <code>John Smith</code>, and assign the user to groups. To assign the user to a group, click inside the Member Of field and select a group or start typing a group name. You cannot assign a custom user to an LDAP group.</p> <ul style="list-style-type: none"> • If you specify a display name, the custom user is listed by this display name in the Polyspace Access interface. Otherwise, the user is listed by the ID (username). • You cannot create duplicate identities (identities with the same ID). <p>If you import users from an LDAP server and one of the LDAP identities is a duplicate of a custom identity in the User Manager database, the custom identity overrides the LDAP identity. Polyspace Access accepts the login credentials and shows information (such as group memberships) for only this identity.</p> <ul style="list-style-type: none"> • After you create a custom user: <ul style="list-style-type: none"> • Add the user sign-in ID to the <code>MLM.opts</code> file to grant that user right-to-use privileges for Polyspace Access. See “Manage Named Users for Polyspace Access” on page 6-7. • Add the user to Polyspace Access. See “Update List of Polyspace Access Users and Groups” on page 4-42. 	<ul style="list-style-type: none"> • To create LDAP users, contact your LDAP administrator. You cannot create LDAP users from the User Manager interface. • The User Manager copies all the users that are returned by the LDAP server to the User Manager database. • If you add a user in your LDAP directory: <ul style="list-style-type: none"> • Add the users to the <code>MLM.opts</code> file to grant those users right-to-use privileges for Polyspace Access. See “Manage Named Users for Polyspace Access” on page 6-7. • The User Manager database syncs automatically with the LDAP server based on the interval you specify in the Synchronization interval (seconds) setting. <p>To update the list of LDAP identities manually, click Sync With LDAP.</p> • Add the user to Polyspace Access. See “Update List of Polyspace Access Users and Groups” on page 4-42.

Operation	Custom Identity	LDAP Identity
Create group	<ul style="list-style-type: none"> • To create a custom group, click Create > Group and then specify a group ID, for instance myGroup • Optionally, you can specify a display name, for instance My Team Members, and assign the group to other groups. To assign the group to another group, click inside the Member Of field and select a group or start typing a group name. You cannot assign a custom group to an LDAP group. <p>You can also assign other users and groups as members of this group. To assign other users and groups as member of this group, click inside the Members field or start typing a user or group name. You can assign LDAP users and groups as members of custom groups.</p> <ul style="list-style-type: none"> • If you specify a display name, the custom group is listed by this display name in the Polyspace Access interface. Otherwise, the group is listed by the ID (username). • After you create a group, add it to Polyspace Access. See “Update List of Polyspace Access Users and Groups” on page 4-42. 	<ul style="list-style-type: none"> • To create LDAP groups, contact your LDAP administrator. You cannot create LDAP groups from the User Manager interface. • The User Manager copies all the groups that are returned by the LDAP server to the User Manager database. • If you add a group in your LDAP directory: <ul style="list-style-type: none"> • The User Manager database syncs automatically with the LDAP server based on the interval you specify in the Synchronization interval (seconds) setting. <p>To update the list of LDAP identities manually, click Sync With LDAP.</p> • Add the group to Polyspace Access. See “Update List of Polyspace Access Users and Groups” on page 4-42.

Operation	Custom Identity	LDAP Identity
Edit or delete users and groups	<ul style="list-style-type: none"> To edit custom identities, click the icon in the Actions column and select View Details. In the window that opens, click Edit. For all identities, you can modify the display name and the list of groups that the identity belongs to. For groups, you can also modify the list identities that belong to this group. <p>You can add LDAP identities as group members.</p> <ul style="list-style-type: none"> You cannot edit a custom identity if that identity overrides a duplicate LDAP identity. To remove an identity, click the icon in the Actions column and select Delete. This operation cannot be undone. <p>To remove an administrator, see "Remove User Manager administrator" in this table.</p> <ul style="list-style-type: none"> If you remove a custom identity that overrides a duplicate LDAP identity, the project permissions and review information associated with that custom identity in Polyspace Access are transferred to the LDAP identity. 	<ul style="list-style-type: none"> To edit or remove LDAP identities, contact your LDAP administrator. You cannot edit or remove LDAP identities from the User Manager interface. <p>You can also remove LDAP identities by adjusting the LDAP filters in the User Manager settings or by disconnecting your LDAP server from the User Manager. See "Connect Your Organization LDAP Server to the User Manager" on page 4-24.</p>
Update user password	Click the icon in the Actions column and select Change Password	To change the password of an LDAP user, contact your LDAP administrator. After the LDAP administrator makes the changes, click Synch With LDAP to update the User Manager database. You cannot change the password of an LDAP user from the User Manager interface.

Sort Users and Groups in the User Manager Interface

To manage large sets of users and groups, sort the identities by using the filter button and the text filter.

- Click **Filter** and clear a filter to hide the corresponding users and groups. For example, if you clear **Group**, you do not see any group in the list of identities.

You can filter users by:

- Type** — Show or hide only users or groups.

- **Source** — Show or hide only custom identities (internal) or identities imported from the LDAP server.
- **Uniqueness** — Show or hide only identities that are unique or identities that override a duplicate LDAP identity.

An identity overrides a duplicate identity when you import identities from LDAP, and one of the LDAP identities has the same ID as a custom identity in the **User Manager** database. The custom identity overrides the LDAP identity and you see the **OVERRIDES** label in the **Source** column.

- Use the text field to filter identities by ID or by display name.

When you use the text filter, the text must match the ID or display name from the start of the string. For instance, if you type Smith in the search filter, this entry matches user Smith Johnson but I does not match user John Smith.

Update List of Polyspace Access Users and Groups

When the **Polyspace Access Web Server** service starts, Polyspace Access populates its list of users and groups from the **User Manager** database. You can select from only this list when you assign analysis findings to users or when you set user and group roles while managing permissions for a project or folder.

After you start the **Polyspace Access Web Server** service, if you add or remove a user or group from the **User Manager** database, Polyspace Access updates the list of users and groups whenever you log into the Polyspace Access web interface. If you are already logged in, Polyspace Access updates the list of users and groups when you refresh your web browser.

If a user or group is removed from the **User Manager** database and that user or group has an assigned Polyspace Access role, that user or group is not removed when you log into or refresh the web interface, even if you restart the **Polyspace Access Web Server** service.

To remove users or groups that have assigned roles from Polyspace Access:

- 1 Click **Restart Apps** in the **Cluster Dashboard**.
- 2 In a web browser, enter the URL that you use to “Open the Polyspace Access Web Interface” on page 4-45 and append `/identities/list/removed` to the URL, for example, `https://access-machine.company.com:9443/identities/list/removed`.

You must be logged in as a user who has **Administrator** privileges. To set a user as **Administrator**, see “General User Manager Settings” on page 4-21.

- 3 Select the usernames that you want to remove from Polyspace Access and click **Confirm clean-up**. To select multiple users, press the **CTRL** key. To return to the Polyspace Access interface, click the back button in your web browser.

See Also

Related Examples

- “Configure User Manager” on page 4-21
- “Manage Project Permissions”

- “Upload Examples and Open Polyspace Access Interface” on page 4-44

Upload Examples and Open Polyspace Access Interface

To start using Polyspace Access, upload results to the Polyspace Access database and open the web interface to view those results.

Upload Examples

Upload Results from the Command line

To upload the examples provided with your Polyspace Bug Finder Server or Polyspace Code Prover Server installation, from the command line, go to the *polyspaceroot*\polyspace and run these commands:

```
bin\polyspace-access -host hostname -port port^  
-upload examples\cxx\Bug_Finder_Example\Module_1\BF_Result
```

polyspaceroot is the path to your Polyspace installation. *hostname* is the fully qualified domain name (FQDN) of the machine that hosts Polyspace Access. *port* is the port number that you specified when starting the `admin-docker-agent` binary. For more information on uploading results from the command line, see “Upload Results at Command Line”.

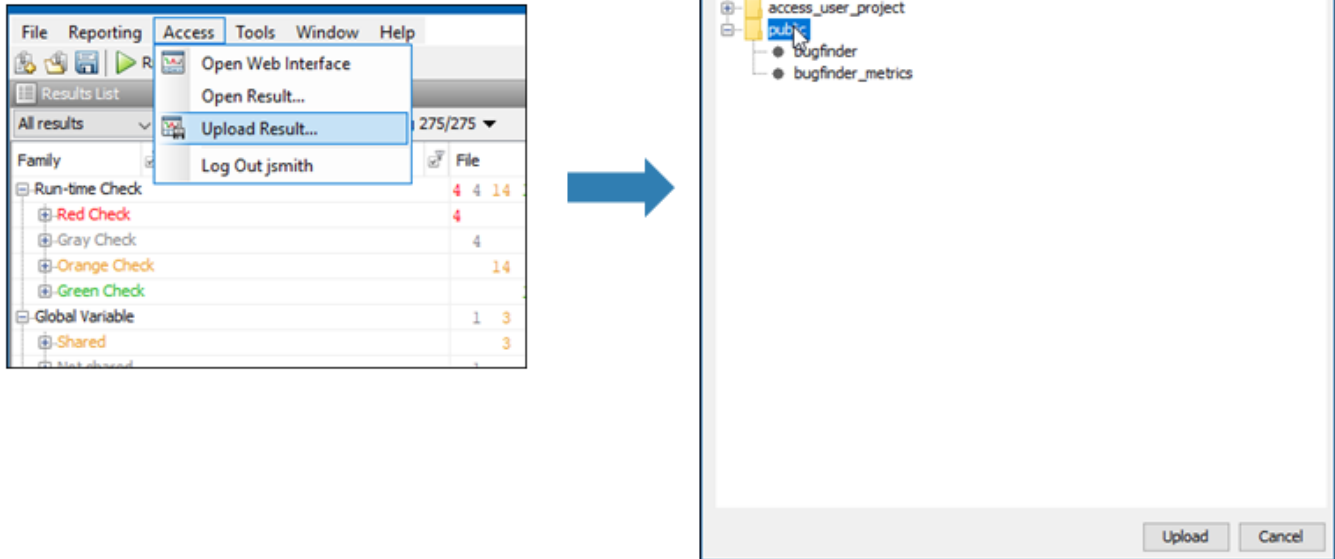
After each command, you are prompted to enter your username and password. Enter the credentials that you use to log in to Polyspace Access.

You cannot use the command line to upload results from a Polyspace Desktop product analysis to the Polyspace Access database.

Upload Results from the Desktop Interface

To upload the demo examples provided with your Polyspace Bug Finder or Polyspace Code Prover:

- 1 Open an example in the desktop interface and select the results in the **Project Browser** pane or switch to the **Results List** pane.
- 2 From the menu, click **Access > Upload Results**. If you are prompted to log in, use your Polyspace Access credentials.
- 3 In the **Upload results to Polyspace Access repository** window, click a folder to select an upload location, then click **Upload**. You can optionally rename the project.



You can also upload to the Polyspace Access database by selecting a result in the **Project Browser** pane and using the context menu.

You must configure the desktop interface to communicate with Polyspace Access. See “Register Polyspace Desktop User Interface” on page 4-47.

After you upload results to Polyspace Access:

- If you open a local copy of the results in the Desktop interface, you cannot make changes to the **Status**, **Severity**, or comment fields.
- To make changes to the **Status**, **Severity**, or comment fields, open the results from Polyspace Access by going to **Access > Open Results**.

Once you save the changes you make to these fields in the desktop interface, the changes are reflected in the Polyspace Access web interface.

Open the Polyspace Access Web Interface

TO open the Polyspace Access interface, click **Open UI** in the **Admin Dashboard**.

The screenshot shows the 'Cluster Admin' interface. At the top, there is a blue header with 'Cluster Admin' on the left and 'Account' with a dropdown arrow on the right. Below the header is the 'Cluster Dashboard' section. It contains two buttons: 'Restart Apps' (blue) and 'Delete Apps' (grey). Below these buttons is a table with columns 'App' and 'Status'. The table lists three applications: 'User Manager' (with a 'Manage users' link), 'Issue Tracker', and 'Polyspace Access' (with an 'Open UI' link). All three applications have a green dot indicating they are 'Running'. To the right of the table is a sidebar titled 'I want to ...' with two links: 'Configure Apps' and 'Configure Nodes'.

App	Status
User Manager Manage users	● Running
Issue Tracker	● Running
Polyspace Access Open UI	● Running

Copy the URL from the address bar, for instance `https://access-machine.company.com:9443/metrics/index.html` and share it with the Polyspace Access users. The URL allows users to open the Polyspace Access interface from any machine connected to the server that hosts Polyspace Access.

Once you complete the installation, close the **Cluster Admin** interface and stop the `admin-docker-agent` binary at the command line by pressing **Ctrl+C**. If you stop the binary before closing the interface, the app status is listed as Unknown state.

See Also

More About

- “Review Polyspace Bug Finder Results in Web Browser”

Register Polyspace Desktop User Interface

To enable interactions between a Polyspace desktop user interface and Polyspace Access, start the desktop interface and go to **Tools > Preferences**.

The screenshot shows the 'Polyspace Preferences' dialog box with the 'Server Configuration' tab selected. The dialog has a title bar with a green checkmark icon and a close button. Below the title bar are five tabs: 'Tools Menu', 'Review Statuses', 'Miscellaneous', 'Character Encoding', and 'Review Scope'. The 'Server Configuration' tab is active and contains the following sections:

- MATLAB Parallel Server cluster configuration:**
 - Job scheduler host name (or Cluster Profile name): localhost
 - Parallel computing username: jsmith
 - Localhost IP address: (empty field)
 - Cluster Profile Manager: Settings (button)
- Polyspace Metrics has been removed.**

To continue managing code quality metrics in a web dashboard, use Polyspace Access instead.

[See details in documentation](#)
- Polyspace Access web interface configuration:**

Review source code and monitor project metrics in an intuitive web interface that is integrated with your bug tracking tool. Log in through a web browser to begin your collaborative review process.

 - Polyspace Access URL: https://access-machine.company.com:9443
 - For https protocol
 - Client keystore path: C:\Users\jsmith\certificates\client-cert.jks (with folder icon)
 - Client keystore password: (masked with dots)
 - Check Polyspace Access connection (button)
- Enable the launching of this desktop UI from the Polyspace Access web interface:**
 - Register Polyspace UI (button)

At the bottom of the dialog are three buttons: OK, Apply, and Cancel.

In the **Server Configuration** tab, complete these fields:

Field	Description
Polyspace Access URL	Specify the URL you use to log into the Polyspace Access interface as <code>http(s)://hostName:port</code> . If you do not know the URL, contact your Polyspace Access administrator.
Client keystore path	Path to the key store file where you imported the signed certificate used to configure Polyspace Access with HTTPS. See “Generate a Client Keystore” on page 4-48. If the Polyspace Access URL does not use HTTPS, leave this field blank.
Client keystore password	The password associated with the key store file. If the Polyspace Access URL does not use HTTPS, leave this field blank.

To check the connection between the desktop interface and the Polyspace Access instance that you specify in the **Polyspace Access URL** field, click **Check Polyspace Access connection**. If required, enter your Polyspace Access credentials.

To associate your Polyspace desktop interface with Polyspace Access, click **Register Polyspace UI**, click **OK**, and then close and restart the desktop interface for the changes to take effect. From the Polyspace Access web interface, you can now start the desktop interface and view currently opened results.

Once you restart the desktop interface, select **Access** to:

- Open the Polyspace Access web interface.
- Open analysis results from the Polyspace Access database.
- Upload analysis results to the Polyspace Access database.

Note On Linux, the desktop interface must already be open before you can view results currently open in Polyspace Access.

Generate a Client Keystore

If Polyspace Access is configured to use HTTPS, generate a Java Key Store (JKS) file to enable communications between Polyspace Access and these Polyspace interfaces on client machines:

- The Polyspace desktop interface.
- The `polyspace-report-generator` binary.
- The `polyspace-results-export` binary.

Obtain the signed certificate used to configure Polyspace Access with HTTPS and import that certificate to the JKS file that you generate.

Obtain Polyspace Access Signed Certificate

If you manage the Polyspace Access software installation, copy the signed certificate that you obtained to configure Polyspace Access with HTTPS to client machines. Use a utility such as `scp` to

securely copy the certificate from the server machine where you installed Polyspace Access to client machines.

If you are a Polyspace Access end user, contact your Polyspace administrator to obtain a copy of the signed certificate. Alternatively, “Download Signed Certificate from Web Browser” on page 4-49.

Generate JKS file and Import Polyspace Access Signed Certificate

To generate the `jks` file, use the `keytool` key and certificate management utility. The `keytool` utility is available with your Polyspace desktop or server product installation in these folders:

- **Windows:** `polyspaceroot\sys\java\jre\win64\jre\bin`
- **Linux:** `polyspaceroot/sys/java/jre/glnxa64/jre/bin`

`polyspaceroot` is your product installation folder, for instance `/usr/local/Polyspace/R2023a`.

It is recommended that you use the `keytool` utility that is shipped with your Polyspace desktop or server product installation to generate the keystore file. `keytool` utilities from JDK alternatives such as OpenJDK might generate keystore files with incompatible formats and should not be used.

For example, if you obtained Polyspace Access signed certificate file `admin_cert.cer`, generate the corresponding JKS file by using this command:

```
keytool -import -trustcacerts -alias cert -file admin_cert.cer -keystore client-cert.jks -storepass password
```

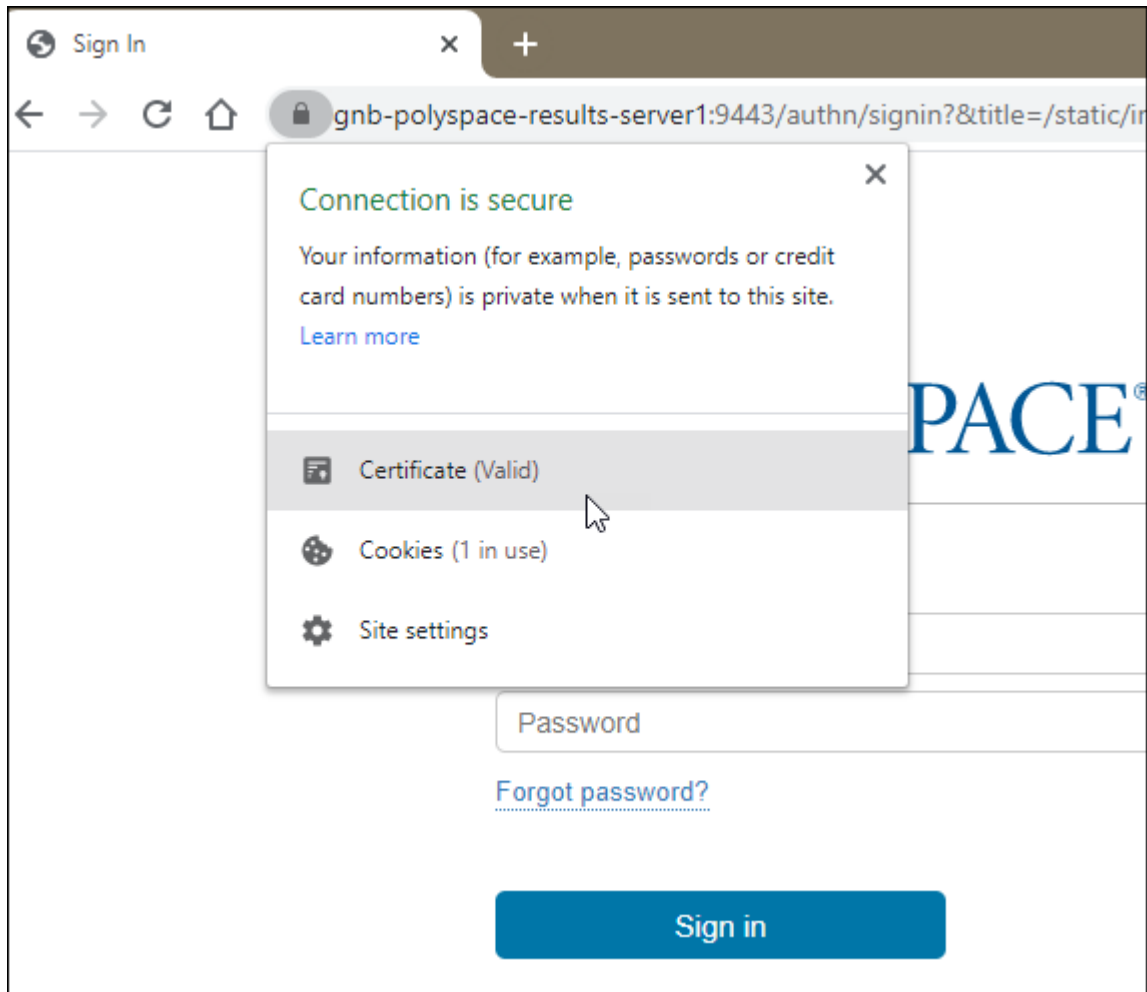
The command outputs file `client-cert.jks`. The password associated with this key store file is `password`.

Download Signed Certificate from Web Browser

To download a copy of the Polyspace Access signed certificate from your Google Chrome® or Microsoft Edge® web browser on Windows, follow these instructions. You might need to modify the instructions depending on your web browser and operating system.

Note Talk to your network security administrator before downloading a signed certificate from your web browser. If your network is not protected from unauthorized access, downloading a certificate from a browser might leave you vulnerable to a man in the middle (MITM) attack.

- 1 Go to the Polyspace Access sign in page, click the padlock icon in the address bar, and then click **Certificate (Valid)**.



- 2 In the **Certificate** window, select the **Details** tab and click **Copy to File**.
- 3 Follow the prompts in the certificate export wizard. Select **DER encoded binary X.509 (.CER)** or **Base-64 encoded X.509 (.CER)** for the **Export File Format**.

See Also

More About

- “Upload Results from the Desktop Interface” on page 4-44
- “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14

Database Backup

To create a backup of your Polyspace Access database, use the `pg_dumpall` PostgreSQL utility. The utility creates a dump of your database. You can then restore the state of the database from when the dump was created. The `pg_dumpall` utility is available in the `polyspace-access-db-0-main` container of Polyspace Access.

Note If you run Polyspace Access version R2021b or earlier, the docker container names might be different. To view the names of currently running containers, use command `docker ps --format '{{.Names}}'`.

Based on your database size and frequency of use, establish a policy for how often you create a backup. Users cannot interact with Polyspace Access while you perform a database backup or restore. Inform users before you start a backup or restore operation.

Create Database Backup

When you create a database backup, the `pg_dumpall` utility generates a list of SQL commands that you use to reconstruct your database. The backup operation requires superuser privileges. The privileges are set through PostgreSQL and are separate from the user privileges on your system. For example, to generate a database dump and save it as `backup_db.sql`, open a terminal on the machine that hosts the **Polyspace Access Database** service and follow these steps. This workflow assumes that all the Polyspace Access services are running.

- 1 To ensure that your backup does not contain partial or corrupted data, stop the **Polyspace Access ETL** and **Polyspace Access Web Server** services before starting the backup operation. In the terminal, enter this command:

```
docker stop polyspace-access-etl-0-main polyspace-access-web-server-0-main
```

- 2 Generate the database backup and save it to `backup_db.sql`.

```
docker exec polyspace-access-db-0-main pg_dumpall -U postgres > backup_db.sql
```

The `docker exec` command runs the `pg_dumpall` utility inside the `polyspace-access-db-0-main` container. The `-U` specifies superuser `postgres`. The output of `pg_dumpall` is then saved as `backup_db.sql`.

You can also compress the generated backup file. For instance, to create a backup and compress it using `gzip`, enter this command:

```
docker exec polyspace-access-db-0-main pg_dumpall -U postgres | gzip > backup_db.gz
```

Note Using `pg_dumpall` on large databases might generate files that exceed the maximum file size limit on some operating systems and can be time consuming.

- 3 Once you complete your backup, use this command to restart the **Polyspace Access ETL** and **Polyspace Access Web Server** services.

```
docker start polyspace-access-etl-0-main polyspace-access-web-server-0-main
```

Restore Database from Backup

To recover your data from a database backup, use the `psql` utility. This utility is available in the `polyspace-access-db-main` container. The operation restores the data and the user permissions

for the Polyspace Access projects. For example, you can restore your database from a backup stored in file `backup_db.sql`. You complete some steps in the **Cluster Admin** interface. Other steps require a terminal on the server that hosts the **Polyspace Access Database** service. On Linux, you might need superuser privileges to complete this operation.

- 1 Stop the **Polyspace Access ETL** and **Polyspace Access Web Server** services. In the terminal, enter this command:

```
docker stop polyspace-access-etl-0-main polyspace-access-web-server-0-main
```

- 2 Delete the folder that stores the Polyspace Access database, and then restart the **Polyspace Access Database** service.

```
sudo rm -rf databaseFolderPath
docker restart polyspace-access-db-0-main
```

databaseFolderPath is the folder path that you specify in the **Data volume** field of the **Polyspace Access Database** service in the Admin **Cluster Settings**, for example:

```
/local/Polyspace/R2020b/appdata/polyspace-access/db
```

If you specify a volume name instead of a folder path in the **Data volume** field, for example `polyspace-data`, use these commands to stop the database service, delete the volume, create a new volume, and restart the database service:

```
docker stop polyspace-access-db-0-main
docker volume rm polyspace-data
docker volume create polyspace-data
docker restart polyspace-access-db-0-main
```

- 3 Restore the database from `backup_db.sql`. In the terminal, enter this command:

```
docker exec -i polyspace-access-db-0-main psql -U postgres postgres <backup_db.sql
```

If you stored your backup in a compressed file, decompress the file, and then pipe its content to the `docker exec` command. For instance, if you use `gzip`, use this command to restore the database from file `backup_db.gz`.

```
gzip -cd backup_db.gz | docker exec -i polyspace-access-db-0-main psql -U postgres postgres
```

- 4 In the **Cluster Admin** interface, click **Restart Apps** to start all the services.

After the services start, open the Polyspace Access interface in your web browser to view the projects that were stored in the database when you created the backup.

Alternatively, you can rely on write ahead log (WAL) files to perform incremental backups and recoveries of your database. The WAL records all changes made to the database. The system stores only a few WAL files and recycles older files.

By creating a base backup and storing all subsequent WAL files, you can restore your database by replaying the WAL sequence up to any point between when you made your base backup and the present. For an example of how to configure an incremental backup, see [Continuous Archiving and Point-in-Time Recovery \(PITR\)](#).

See Also

More About

- “Storage and Port Configuration” on page 4-4

- “Install Polyspace Access for Web Reviews”

Database Backup for Polyspace Access Versions R2020a and Earlier

Note This topic applies to Polyspace Access versions R2020a and earlier. For versions R2020b and later, see “Database Backup” on page 4-51.

To create a backup of your Polyspace Access database, use the `pg_dumpall` PostgreSQL utility. The utility creates a dump of your database. You can then restore the state of the database from when the dump was created. The `pg_dumpall` utility is available in the `polyspace-db` container of Polyspace Access.

Based on your database size and frequency of use, establish a policy for how often you create a backup. Users cannot interact with Polyspace Access while you perform a database backup or restore. Before you start a backup or restore operation, inform your users.

Create Database Backup

When you create a database backup, the `pg_dumpall` utility generates a list of SQL commands that you use to reconstruct your database. The backup operation requires superuser privileges. The privileges are set through PostgreSQL and are separate from the user privileges on your system. For example, to generate a database dump and save it as `backup_db.sql`, open a terminal on the machine that hosts the **Database** service and follow these steps.

- 1 To ensure that your backup does not contain partial or corrupted data, stop the **ETL** and **Web Server** services before starting the backup operation. In the terminal, enter this command:

```
docker stop polyspace-etl polyspace-web-server
```

Tip Stopping the Polyspace Access services by using the Cluster Operator (COP) interface on Windows Server 2019 might result in a slow response time. Instead, use these commands to stop the services:

```
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
```

- 2 Generate the database backup and save it to `backup_db.sql`.

```
docker exec polyspace-db pg_dumpall -U postgres > backup_db.sql
```

The `docker exec` command runs the `pg_dumpall` utility inside the `polyspace-db` container. The `-U` specifies superuser `postgres`. The output of `pg_dumpall` is then saved as `backup_db.sql`. Be aware that using `pg_dumpall` on large databases might generate files that exceed the maximum file size limit on some operating systems and can be time consuming.

- 3 To restart the **ETL** and **Web Server** services.

```
docker start polyspace-etl polyspace-web-server
```

Restore Database from Backup

To recover your data from a database backup, use the `psql` utility. This utility is available in the `polyspace-db` container. The operation restores the data and the user permissions for the Polyspace

Access projects. For example, you can restore your database from a backup stored in file `backup_db.sql`. You complete some steps in the COP interface. Other steps require a terminal on the server that hosts the **Database** service.

- 1 Create a database. In the COP interface, delete all the services, then click **Settings** in the left pane. Under the **Database** settings, click **create volume** next to the **Data volume** field to create a volume, then select this volume from the **Data volume** drop-down list. If you do not see a **create volume** link, specify a new folder path in the **Data volume** field. If the folder path does not exist, the COP creates it. Save your changes.
- 2 Return to the **Services** tab, click **PROVISION**, then start only the **Database** service.
- 3 Restore the database from `backup_db.sql`. In the terminal, enter this command:

```
docker exec -i polyspace-db psql -U postgres postgres <backup_db.sql
```

If you stored your backup in a compressed file, decompress the file, and then pipe its content to the `docker exec` command. For instance, if you use `gzip`, to restore the database from file `backup_db.gz`, enter:

```
gzip -cd backup_db.gz | docker exec -i polyspace-db psql -U postgres postgres
```

- 4 In the COP interface, click **START ALL** to start all the services.

After the services start, open the Polyspace Access interface in your web browser to view the projects that were stored in the database up to when you created the backup.

Alternatively, you can rely on write ahead log (WAL) files to perform incremental backups and recoveries of your database. The WAL records all changes made to the database. The system stores only a few WAL files and recycles older files.

By creating a base backup and storing all subsequent WAL files, you can restore your database by replaying the WAL sequence up to any point between when you made your base backup and the present. For an example of how to configure an incremental backup, see Continuous Archiving and Point-in-Time Recovery (PITR).

See Also

More About

- “Storage and Port Configuration” on page 4-4
- “Install Polyspace Access for Web Reviews”

Database Clean Up

To optimize the performance of the Polyspace Access database, perform regular database clean up operations such as vacuuming and the deletion of old or obsolete projects. It is recommended that you back up your database before you perform a clean up operation. See “Create Database Backup” on page 4-51.

Perform Database Vacuuming

When a row is updated or deleted in a database table, it is not physically removed from the table because other database transactions might still use the old version of the row. To reclaim the disk space of old rows that are no longer used by any database transaction, use the PostgreSQL `vacuumdb` command. Vacuuming the database regularly prevents your database disk space from growing too large or fragmented.

Before you perform a vacuum operation, ensure that no users are connected to Polyspace Access then stop the **Polyspace Access Web Server** and **Polyspace Access ETL** services. To stop the services, from a terminal on the server hosting these services, use this command and entering:

```
docker stop polyspace-access-etl-0-main polyspace-access-web-server-0-main
```

Note If you run Polyspace Access version R2021b or earlier, the docker container names might be different. To view the names of currently running containers, use command `docker ps --format '{{.Names}}'`.

To vacuum your Polyspace Access database, open a terminal on the server hosting your database and enter:

```
docker exec polyspace-access-db-0-main vacuumdb -U postgres prs_data
```

You can also run the `vacuumdb` command and use the `--analyze` option to update the PostgreSQL server statistics. Open a terminal on the server hosting your database and enter:

```
docker exec polyspace-access-db-0-main vacuumdb -U postgres --analyze prs_data
```

Accurate server statistics help prevent degradations in the performance of the database.

To minimize the size of your database tables and return unused space to the operating system, run `vacuumdb` by using the `--full` option. Open a terminal on the server hosting your database and enter:

```
docker exec polyspace-access-db-0-main vacuumdb -U postgres --full prs_data
```

This operation can take a long time and writes a new version of the table that does not have any empty spaces. When you perform a full vacuum, no other database process can run in parallel. The database is not accessible during a full vacuum.

Establish a policy for how often you want to perform a regular and a full vacuum. For instance perform a regular vacuum weekly.

After you complete the vacuum operation, restart the **Polyspace Access Web Server** and **Polyspace Access ETL** services. Use this command:

```
docker start polyspace-access-etl-0-main polyspace-access-web-server-0-main
```

After you restart the **Polyspace Access Web Server** service, it might take a few moments before you can open Polyspace Access in your web browser.

Delete Project Runs or Entire Projects

When users delete projects from the **Project Explorer** of the Polyspace Access web interface, the projects move to the **ProjectsWaitingForDeletion** folder. The projects, including all the runs that you uploaded to the projects, remain in the database until you explicitly delete them.

The **ProjectsWaitingForDeletion** folder is visible only to Polyspace Access users who have the role of **Administrator**. Even users who have the **Administrator** role cannot delete projects *from the Polyspace Access interface*.

Define a policy for how often you delete older projects or project runs from the database. Automate this operation by using a script. You can delete older results even if these are not in the **ProjectsWaitingForDeletion** folder.

To remove old project runs or entire projects from your database, write a command in a text file that you save as a `.pscauto` file. Run the command by copying the `.pscauto` file to the **Storage directory** of the **Polyspace Access ETL** service. Only a user who has write privileges on the **Storage directory** can perform this operation.

- To delete older Polyspace Access project runs from a project but not the project itself, use the `clean_project` command. Specify the project path and the time range for which you want to delete project runs using one of these command parameters.
 - `clean_project projectPath DATE YYYY-MM-DD`
The command deletes project runs that were uploaded before YYYY-MM-DD.
 - `clean_project projectPath MAXRUNS NNN`
NNN is an integer. The command keeps the NNN most recent runs. To delete all the project runs, use MAXRUNS 0.
 - `clean_project projectPath AGE DDD`
DDD is the number of days. To keep only recently uploaded results, use this option. The command deletes project runs that are older than DDD days.

You cannot delete project runs by specifying the run ID with the `clean_project` command.

- To completely delete a project from the Polyspace Access database, use the `delete_project` command and specify the project path:

```
delete_project projectPath
```

projectPath is the full project path in the Polyspace Access **Project Explorer**. To get a project path, use the context menu in the **Project Explorer** or, at the command line, use the `polyspace-access` binary with the `-list-project` flag. For more information, see `polyspace-access -h -list-project`.

If the path contains whitespace characters, enclose the project path in double quotes. If you use `echo` to write the commands to a file, you must also use a `"\"` character to escape whitespace characters in the project path.

For example, to perform a one-time cleanup of project `public/Bug_Finder_Example` (Bug Finder) and remove all results uploaded before a specific date:

- 1 Open a text editor, paste this command, then save the file as a `.pscauto` file, for instance `cleanup.pscauto`.

```
clean_project "public/Bug_Finder_Example (Bug Finder)" DATE 2019-09-01
```

- 2 Copy the file to the **Storage directory** of the **Polyspace Access ETL** service, for instance:

```
cp cleanup.pscauto /local/ACCESS/install_dir/appdata/polyspace-access/storage
```

The path of the **Storage directory** is available in the **Cluster Settings** of the **Cluster Admin** interface. Alternatively, you can find this path in the `settings.json` file by searching for `etlStorageDir`.

All analysis runs uploaded to project `public/Bug_Finder_Example (Bug Finder)` prior to September 1, 2019 are deleted from the database.

You can also perform an automatic cleanup on a specific project every time you upload a run to that project. To automate the cleanup of a project, add this line above the cleanup command:

```
assign_to_project projectPath AFTER_STATISTICS scriptName
```

where *projectPath* is the full project path in the Polyspace Access **Project Explorer** and *scriptName* is the name of the script that you assign to that project. For example, to keep only the 10 most recent runs every time you upload a result to `public/Bug_Finder_Example (Bug Finder)`, save these commands to your `.pscauto` file.

```
assign_to_project "public/Bug_Finder_Example (Bug Finder)" AFTER_STATISTICS myScript  
clean_project "public/Bug_Finder_Example (Bug Finder)" MAXRUNS 10
```

The cleanup commands that you enter after the `assign_to_project` line are stored internally in a script `myScript` that is assigned to the project `public/Bug_Finder_Example (Bug Finder)`. Use distinct names for the internal script that you assign to different projects. You specify the internal script name with the last parameter of the `assign_to_project` command. After you copy the file to the **Storage directory** of the **Polyspace Access ETL** service, the automatic cleanup starts.

To turn off the automatic cleanup, save this command to a `.pscauto` file and copy that file to the **Storage directory**:

```
unassign_to_project "public/Bug_Finder_Example (Bug Finder)" myScript
```

You must provide the name of the internal script that you assigned to the project by using the `assign_to_project` command. Make sure that you keep a record of the internal script names and the projects to which they are assigned.

Caution You cannot recover the data that you delete by using the `.pscauto` script unless you have a backup copy of the data.

See Also

More About

- “Storage and Port Configuration” on page 4-4
- “Database Backup” on page 4-51
- “Configure Polyspace Access App Services” on page 4-33

Update or Uninstall Polyspace Access

You can update or uninstall Polyspace Access. Before you begin, inform Polyspace Access users of the upcoming update or uninstallation.

Update Polyspace Access

To update Polyspace Access, download and unzip the new installation image. See Download instructions on page 4-13.

When you perform an update, you can reuse your current settings and database or you can import a snapshot of the current database to a new database. If you do not reuse the settings and database, follow the standard installation instruction. See “Install Polyspace Access for Web Reviews”.

Compatibility Considerations

- The Polyspace Access docker container names might change between releases. If you run scripts that use the container names, check those names in the new release and update your scripts accordingly.

To view the names of currently running containers, use command `docker ps --format '{{.Names}}'`.

- If you update Polyspace Access version R2021a or earlier to version R2021b or later, and both of these are true:
 - You use your company LDAP for user authentication.
 - You use the option `--force-exposing-ports` when you start the `admin-docker-agent` binary for the new installation.

Check that all the services have a valid port number assigned to them before you click **Restart Apps** to start the services.

To view the currently assigned ports for the services, in the **Cluster Admin** dashboard, click **Configure Nodes** and then go to the **Services** tab of the **Nodes** settings.

- To download results stored in Polyspace Access version R2022a Update 1 or later by using the `polyspace-access --download` command, update the Polyspace product that you run the command from to version R2022a Update 1 or later.
- If you update Polyspace Access version R2020a or earlier to version R2020b or later and you use the **embedded LDAP** in the older version, use the **User Manager** internal directory in the newer version. Enter the user names and passwords from the LDIF file into the **User Manager** interface. See “Create, Edit, or Remove Users and Groups” on page 4-37.

Create Polyspace Access Snapshot

To reuse your database and settings, create a snapshot of your current instance of Polyspace Access. In your snapshot, include a backup of the:

- Polyspace Access database. See “Database Backup” on page 4-51. To back up a database for Polyspace Access version R2020a or earlier, see “Database Backup for Polyspace Access Versions R2020a and Earlier” on page 4-54.
- Current settings. Make a copy of the `settings.json` file. This file is typically in the same folder as the `admin-docker-agent` binary.

- User profiles that are not stored in your company LDAP.

To back up user profiles that are stored in the **User Manager** internal directory (Polyspace Access R2020b and later), save a copy of the internal directory folder. You specify the path of this folder in the **Internal directory database volume** field of the Admin **Cluster Settings**, for example, `/local/Polyspace/R2020b/appdata/usermanager/db`.

To back up user profiles that are stored in the **embedded LDAP** (Polyspace Access R2020a and earlier), save a copy of the LDIF file. You specify the path of this file in the **LDIF file** field of the Cluster Operator settings.

Uninstall Polyspace Access and Reuse Settings and Database with New Installation

- 1 To remove your current instance, see “Uninstall Polyspace Access” on page 4-62 but omit step 2.a.
- 2 In the new installation folder, start the `admin-docker-agent` binary and use the `--data-dir` flag to point to the folder that contains the backup `settings.json` file. For example, if `settings.json` is in folder `/local/Polyspace/R2020a_backup`, enter:

```
./admin-docker-agent --data-dir /local/Polyspace/R2020a_backup
```

- 3 Open the **Cluster Admin** web interface and click **Restart Apps**.

If you install your new Polyspace Access instance on a different machine, you cannot reuse your SSL certificates. See “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14.

Keep Current Polyspace Access and Reuse Settings and Database with New Installation

- 1 In the new installation folder, start the `admin-docker-agent` binary and use the `--data-dir` flag to point to the folder that contains the backup `settings.json` file.
- 2 Open the **Cluster Admin** web interface, click **Configure Apps**, and then enter a new folder path in the **Data volume** field to create a new database. You cannot use the same database in two different instances of Polyspace Access at the same time.
- 3 Start the **Polyspace Access Database** service and import the database snapshot into the new database. For example, if you stored the database snapshot in `db_backup.sql`, at the command line, enter:

```
docker restart polyspace-access-db-0-main
docker exec -i polyspace-access-db-0-main psql -U postgres postgres <backup_db.sql
```

Ensure that the **Polyspace Access Web Server** and **Polyspace Access ETL** services are stopped before you import the database.

- 4 Return to the **Cluster Dashboard** and click **Restart Apps**.

If you install your new Polyspace Access instance on a different machine, you cannot reuse your SSL certificates. See “Choose Between HTTP and HTTPS Configuration for Polyspace Access” on page 4-14.

Reuse Database with New Polyspace Access Installation

Start the `admin-docker-agent` in the new installation folder, and then, for instructions, see “Restore Database from Backup” on page 4-51.

Check License Manager Version

To avoid any potential issues with license file operation, make sure that you run the latest license manager software version. To view the latest license manager software version available, see the FlexNet Version on this page.

To check your current license manager software version, at the command line, depending on your operating system, enter the commands listed in this table.

Windows	<code>cd LM_Folder\etc\win64 lmgrd.exe -v</code>
Linux	<code>cd LM_Folder/etc/glnx64 ./lmgrd -v</code>

LM_Folder is the folder where you installed the license manager. See also “Update Network License Manager Software”.

Check for Docker Network Conflicts

The Polyspace Access services run inside a docker network. See Networking overview. When you update Polyspace Access, the services reuse the existing docker network unless you explicitly specify a different docker network.

To avoid network conflicts, check with your network administrator whether the IP range of the docker network used by your current installation is not also used by other services. If you already performed this check during a previous installation and you have not experienced networking issues, you may skip this section. Otherwise, follow these steps. Run the commands on the machine where you install Polyspace Access.

- 1 To view which docker network your current installation uses, while the Polyspace Access services are running, use this command:

```
docker container inspect polyspace-access-etl-0-main
```

and then inspect the `NetworkSettings.Networks` node in the output.

If more than one network is listed in the output, perform step 2 for each network.

- 2 To view the IP range that is used by the docker network from step 1, run this command:

```
docker network inspect networkName
```

and inspect the `IPAM.config` node. Contact your network administrator to determine if the IP range is used by other services. *networkName* is the name of the docker network, for example `mathworks`.

- If your docker network conflicts with an existing network, create a docker network and specify the subnet and gateway to avoid conflicts with existing networks. See `docker network create`. To use the new network, specify the `--network-name newNetworkName` option when you start the `admin-docker-agent` binary.

Use the command `docker network rm networkName` to remove Docker networks that you no longer use.

- If the docker network does not conflict with an existing network, no action is required. The network is reused with the new installation.

Uninstall Polyspace Access

- 1 Verify that the **Cluster Admin** agent is running. Use the command:

```
docker stats --no-stream
```

If **admin** is not listed under the **NAME** column in the command output, start the **admin-docker-agent** binary.

- 2 Open the **Cluster Admin** web interface and click **Delete Apps**. After you delete an app, the status indicator turns gray and you see the text **Not installed** next to the indicator.

Deleting the **Polyspace Access Database** service and uninstalling Polyspace Access does not erase the results that you uploaded to the database from the data volume.

- a To delete a data volume and its content, manually delete the folder where you store the database.

```
sudo rm -rf databaseFolderPath
```

databaseFolderPath is the folder path that you specify in the **Data volume** field of the **Polyspace Access Database** service in the Admin **Cluster Settings**, for example, `/local/Polyspace/R2023a/appdata/polyspace-access/db`.

If you specify a volume name instead of a folder path in the **Data volume** field, for instance `polyspace-data`, use this command to delete the volume:

```
docker volume rm polyspace-data
```

- 3 Stop the **admin-docker-agent** binary from the command line window by pressing **CTRL+C**, and then stop the remaining services:

```
docker stop gateway \  
polyspace-access \  
issuetracker \  
usermanager
```

To use this command in Windows PowerShell, replace the backslash "`\`" characters with a backtick "```".

- 4 Delete the Polyspace Access installation folder.

See Also

`admin-docker-agent`

More About

- “Configure and Start the Cluster Admin” on page 4-13
- “Configure User Manager” on page 4-21
- “Configure Issue Tracker” on page 4-29
- “Configure Polyspace Access App Services” on page 4-33
- “Database Backup” on page 4-51

Install Polyspace as You Code

Install Polyspace as You Code Using Installer

Polyspace as You Code checks your code for bugs and coding standard violations while you work in a code editor or in these supported IDEs:

- Visual Studio® 2017 (versions 15.9.* and later) and 2019 (versions 16.7.* and later)
- Visual Studio Code (versions 1.74 and later)^{1, 2}
- Eclipse (versions 2019-19 to 2022-12).³and Eclipse based IDEs such as Code Composer Studio, HighTech, or Windriver Workbench.

Polyspace as You Code uses the Polyspace Access Network Named Users (NNU) license and requires a license manager to manage license checkouts.

Prepare Polyspace as You Code Installation

The Polyspace as You Code installer is available in the Polyspace Access installation image.

In a typical installation:

- 1 An administrator downloads Polyspace Access, installs the license manager on a server machine, and configures the Polyspace Access license.

- a To download the Polyspace Access installation image, go to the MathWorks download page, expand the **Get Polyspace Access** section and click **Download**.

You might have to log into your MathWorks Account to complete this step.

- b The license manager is available with your Polyspace Access installation. You can also download the license manager from License Manager Download.

To configure the Polyspace Access license, see “Configure Polyspace Access License” on page 6-2.

Provide end users with the path to the client license file that you configure in “Step 3: Configure Client License” on page 6-4. The license is typically named `network.lic` and contains these lines:

```
SERVER lmHostname HostID 27000
USE_SERVER
```

where *lmHostname* is the hostname of the machine where you install the license manager. *HostID* is the MAC address that you provided to activate the Polyspace Access license.



- 2 An administrator or an end user installs Polyspace as You Code on client machines that are on the same network as the license manager server machine.

Tip Polyspace as You Code uses a local server for communications between the analysis engine and the IDE extensions. If you use a proxy server, configure your system to exclude the

- 1 It is likely that Polyspace as You Code will run successfully with later version, however, there might be certain issues.
- 2 If your version of Visual Studio Code uses the Workspace Trust feature, the Polyspace as You Code extension is disabled when you open a folder in **Restricted Mode**. To enable the extension, mark the folder as trusted. See Workspace Trust.
- 3 If you install the Polyspace as You Code Eclipse plugin on Linux, see “Configure Eclipse for Supported Java Version on Linux” on page 5-16.

localhost from using the proxy. For instance, on Linux, set environment variable `export no_proxy=localhost`.

This table describes additional required steps before you begin your installation, depending on your role and the type of installation.

Type of Installation	Role	Steps
Download installer from Polyspace Access interface and install interactively	Administrator	<p>a Install and start Polyspace Access. See “Install Polyspace Access for Web Reviews”.</p> <p>b Provide end users with the Polyspace Access URL and path to license file.</p>
	End User	<p>a Obtain the path to the license file and Polyspace Access URL from your administrator.</p> <p>b Download and unzip the Polyspace as You Code installation folder. Log into the Polyspace Access interface and click  > Download Polyspace as You Code.</p> <p>c For installation, see “Install Polyspace as You Code Interactively” on page 5-4. If you install Visual Studio Code or Eclipse IDE extensions, you must provide the installation path of the IDEs.</p>
Start installer from shared location and install interactively	Administrator	<p>a Install and start Polyspace Access. See “Install Polyspace Access for Web Reviews”.</p> <p>b Download and unzip the Polyspace as You Code installation folder. Log into the Polyspace Access interface and click  > Download Polyspace as You Code.</p> <p>c Provide end users with the path to the installation folder that you unzipped in step a, and the path to the license file.</p>
	End User	<p>a Obtain the path to the license file and location of the installer from your administrator.</p> <p>b For installation, see “Install Polyspace as You Code Interactively” on page 5-4. If you install Visual Studio Code or Eclipse IDE extensions, you must provide the installation path of the IDEs.</p>

Type of Installation	Role	Steps
Install noninteractively	Administrator or End User	<p>a Obtain the path to the license file and location of the installer.</p> <p>b Modify the installer properties file:</p> <ul style="list-style-type: none"> • Provide the path for the Polyspace as You Code installation folder. • Determine whether you want to install IDE extensions. For Visual Studio Code and Eclipse IDEs, you must provide the installation path of the IDEs. <p>See “Install Polyspace as You Code Noninteractively” on page 5-6.</p>

Note To obtain a Polyspace as You Code installer without installing and starting Polyspace Access , contact your MathWorks sales representative.

The Polyspace as You Code plugin for Eclipse-based IDEs is not compatible with the Polyspace desktop plugin for Eclipse. If you use the installer to install the Polyspace as You Code Eclipse plugin, Polyspace uninstalls the desktop plugin.

Install Polyspace as You Code Interactively

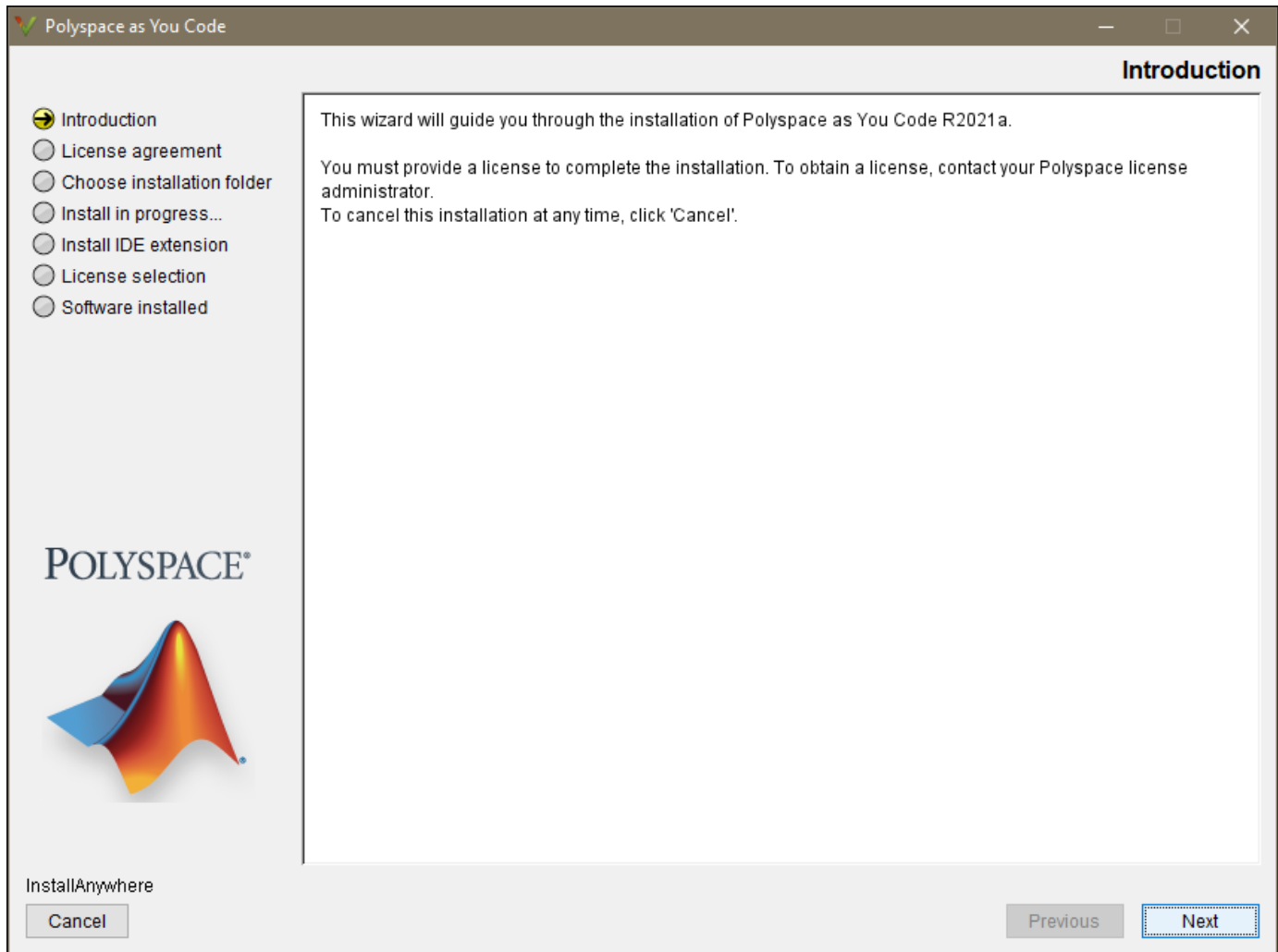
Prerequisites

— See “Prepare Polyspace as You Code Installation” on page 5-2.

To install Polyspace as You Code interactively, at the command line, navigate to the folder that contains the Polyspace as You Code installer and enter the commands listed in this table, depending on your operating system.

Windows	<p>setup.exe</p> <p>You can also start the installer by double-clicking the setup.exe binary.</p>
Linux	./install.sh

Note If you install the Polyspace as You Code Eclipse plugin on Linux, see “Configure Eclipse for Supported Java Version on Linux” on page 5-16.



In the Polyspace as You Code installation wizard, click **Next** and follow the prompts to complete the installation.

- If you do not have a license file available, you may skip the **License selection** step and provide a license at a later time.

To complete the installation, after you obtain the `network.lic` license file from your Polyspace administrator, copy that file to the `licenses` folder located in the Polyspace as You Code installation folder. For instance `/usr/local/PolyspaceAsYouCode/R2023a/licenses`.

- If you install the IDE extensions:
 - The installer might take some time before you can proceed to the next installation step.
 - Before you start using Polyspace as You Code, you must complete the configuration of the extensions. See “Polyspace as You Code IDE Extension Settings”.

Uninstall Polyspace as You Code Interactively

- 1 Go to the `Uninstall` folder located in your Polyspace as You Code installation folder, for instance `C:\Program Files\Polyspace as You Code\R2023a\Uninstall`.

- 2 Start the `Uninstall.exe` binary (Windows) or the `Uninstall` script (Linux).

The operation uninstalls Polyspace as You Code and the IDE extensions.

Install Polyspace as You Code Noninteractively

Prerequisites

— See “Prepare Polyspace as You Code Installation” on page 5-2.

When installing noninteractively, you can either:

- Start the installer by using a properties file where you specify your installation configuration.
- Start the installer by using the `-silent` option to install only the Polyspace as You Code analysis engine.

Install Polyspace as You Code by Using Installer Properties File

To specify your installation configuration in a properties file and install Polyspace as You Code noninteractively:

- 1 Edit the installer properties file. A template installer properties file is located in `installerRoot/Docs/installer.properties`, where `installerRoot` is the folder that contains the Polyspace as You Code installer.
- 2 At the command line, navigate to the folder that contains the Polyspace as You Code installer. Depending on your operating system, enter one of the commands listed in this table.

Windows	<code>setup.exe -f installerPropertiesFile</code>
Linux	<code>./install.sh -f installerPropertiesFile</code>

`installerPropertiesFile` is the full file path to the installer properties file.

For example, to install Polyspace as You Code in folder `C:\Program Files\Polyspace as You Code\R2023a` with license file `C:\Polyspace\licenses\network.lic`, specify the following installer properties file. The line `INSTALL_VS_PLUGIN=true` enables the installation of the Visual Studio extension.

```
# Installer properties file
# Uncomment the line below to launch the installer in silent mode.
INSTALLER_UI=silent

# Uncomment the line below and specify an installation folder path or
# leave line commented to install in default installation folder.
# Enter '\' characters in the folder path as '\\', for instance:
# C:\Program Files\Polyspace as You Code\R2023a
USER_INSTALL_DIR=C:\Program Files\Polyspace as You Code\R2023a

##### Begin - Windows only options #####
# Set to true to install the Visual Studio extension.
INSTALL_VS_PLUGIN=true
##### End - Windows only options #####

# Set to true to install the Visual Studio Code extension.
INSTALL_VSCODE_PLUGIN=false
# If INSTALL_VSCODE_PLUGIN is set to "true", provide the path to the
# Visual Studio Code installation folder, for instance /usr/share/code
VSCODE_INSTALL_FOLDER=

# Set to true to install the Eclipse extension.
INSTALL_ECLIPSE_PLUGIN=false
# If INSTALL_ECLIPSE_PLUGIN is set to "true", provide the path to the
# Eclipse executable
```



```
ECLIPSE_EXECUTABLE_PATH=
# Provide the path to the license file. The file will be copied to <install_root>/licenses.
LICENSE_FILE=C:\\Polyspace\\licenses\\network.lic

# Keep true to improve Polyspace by sending user experience information to MathWorks.
# Set to false to not send information to MathWorks.
# See details here: https://www.mathworks.com/support/faq/user_experience_information_faq.html
SEND_UX_INFO_TO_MW=true
```

Note If you install the Polyspace as You Code Eclipse plugin on Linux, see “Configure Eclipse for Supported Java Version on Linux” on page 5-16.

Install Only Polyspace as You Code Analysis Engine

To install only the Polyspace as You Code analysis engine without opening the graphical interface, start the installer binary by using the `-silent` flag. The installer installs Polyspace as You Code to these default locations, based on your operating system:

Windows	C:\Program Files\Polyspace as You Code\R2023a
Linux	/usr/local/PolyspaceAsYouCode/R2023a
macOS	/Applications/PolyspaceAsYouCodeR2023a

To specify a different installation path, use the `-install-dir` flag, for instance `./install.sh -silent -install-dir /local/myFolder`.

To complete the installation, copy your Polyspace Access `network.lic` license file to the `licenses` folder located in the Polyspace as You Code installation folder. For instance `/usr/local/PolyspaceAsYouCode/R2023a/licenses`.

Uninstall Polyspace as You Code Noninteractively

To uninstall Polyspace as You Code without opening the graphical interface:

- 1 Go to the `Uninstall` folder located in your Polyspace as You Code installation folder, for instance `C:\Program Files\Polyspace as You Code\R2023a\Uninstall`.
- 2 Start the `Uninstall.exe` binary (Windows) or the `Uninstall` script (Linux or Mac OS) by using the `-i silent` flag.

The uninstall operation removes Polyspace as You Code and the IDE extensions.

See Also

Related Examples

- “Configure Polyspace Access License” on page 6-2
- “Install Polyspace as You Code Extension in Visual Studio” on page 5-8
- “Install Polyspace as You Code Extension in Visual Studio Code” on page 5-10
- “Install Polyspace as You Code Plugin in Eclipse” on page 5-13

Install Polyspace as You Code Extension in Visual Studio

The Polyspace as You Code extension in the Visual Studio IDE allows you to run Polyspace on the file that you are currently viewing and see analysis results such as bugs and coding standard violations. You must install the Polyspace as You Code analysis engine to run the analysis. The extension allows you to point to this analysis engine from Visual Studio and show results produced by the Polyspace analysis.

You can install the extension in one of two ways:

- While running the Polyspace as You Code installer, select the option to install the extension. The installer installs the Polyspace as You Code analysis engine and the extension.

See “Install Polyspace as You Code Using Installer” on page 5-2.

- If you install only the analysis engine while running the Polyspace as You Code installer, use the Visual Studio Code extension installer (VSIX) file at a later time. The VSIX file is available only after you run the installer.

The rest of this topic describes the second approach where you skip the installation of the extension while running the installer and use the VSIX file to install the extension later.

You can install the extension in Visual Studio 2017 and 2019.

Interactive Installation

To install the extension interactively:

- 1 Double-click the VSIX file in the folder *polyspaceroot*\polyspace\plugin\visual_studio.

Here, *polyspaceroot* is the Polyspace as You Code installation folder, for instance, C:\Program Files\Polyspace as You Code\R2023a.

- 2 Follow the prompts on the screen.

If you see a message that indicates that your Visual Studio IDE does not satisfy the prerequisites for the extension and you are using Visual Studio 2017 or Visual Studio 2019, install the latest updates to these versions and try re-installing the extension.

After installation, open the Visual Studio IDE and check that the extension has been installed. For instance, in Visual Studio 2019, select **Extensions > Manage Extensions**. You should see Polyspace in the list of extensions installed. You can also disable or uninstall the extension right from this list.

Command-Line Installation

If you want to install the extension without opening a graphical user interface, you can run the VSIX installer at the command line with the /q flag.

- To install the extension in all Visual Studio versions on the machine, enter:

```
cd VSIXInstallerpath
VSIXInstaller.exe /q polyspaceroot\polyspace\plugin\visual_studio\polyspace.vsix
```

Here, *VSIXInstallerpath* is the path to the VSIX installer file. For instance, in a Visual Studio 2019 installation, the path can be C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional\Common7\IDE .

- To install the extension on a specific Visual Studio versions installed on the machine, enter:

```
cd VSIXInstallerpath  
VSIXInstaller.exe /q /s:name /v:version ^  
polyspaceroot\polyspace\plugin\visual_studio\polyspace.vsix
```

Here, *name* is name of the Visual Studio application, for instance, **Pro** for the Visual Studio Professional edition, and *version* is the version number in the form *major.minor*, for instance, **16.0** for the major version of Visual Studio 2019.

To uninstall the extension silently, use the `/u` flag.

For more information on the flags, search for the online VSIXInstaller documentation or simply enter:

```
cd VSIXInstallerpath  
VSIXInstaller.exe
```

See Also

More About

- “Install Polyspace as You Code Using Installer” on page 5-2

Install Polyspace as You Code Extension in Visual Studio Code

The Polyspace as You Code extension in the Visual Studio Code IDE allows you to run Polyspace on the file that you are currently viewing and see analysis results such as bugs and coding standard violations. You must install the Polyspace as You Code analysis engine to run the analysis. The extension allows you to point to this analysis engine from Visual Studio Code and show results produced by the Polyspace analysis.

You can install the extension in one of two ways:

- While running the Polyspace as You Code installer, select the option to install the extension. The installer installs the Polyspace as You Code analysis engine and the extension.

See “Install Polyspace as You Code Using Installer” on page 5-2.

- If you install only the analysis engine while running the Polyspace as You Code installer, use the Visual Studio Code extension installer (VSIX) file at a later time. The VSIX file is available only after you run the installer.

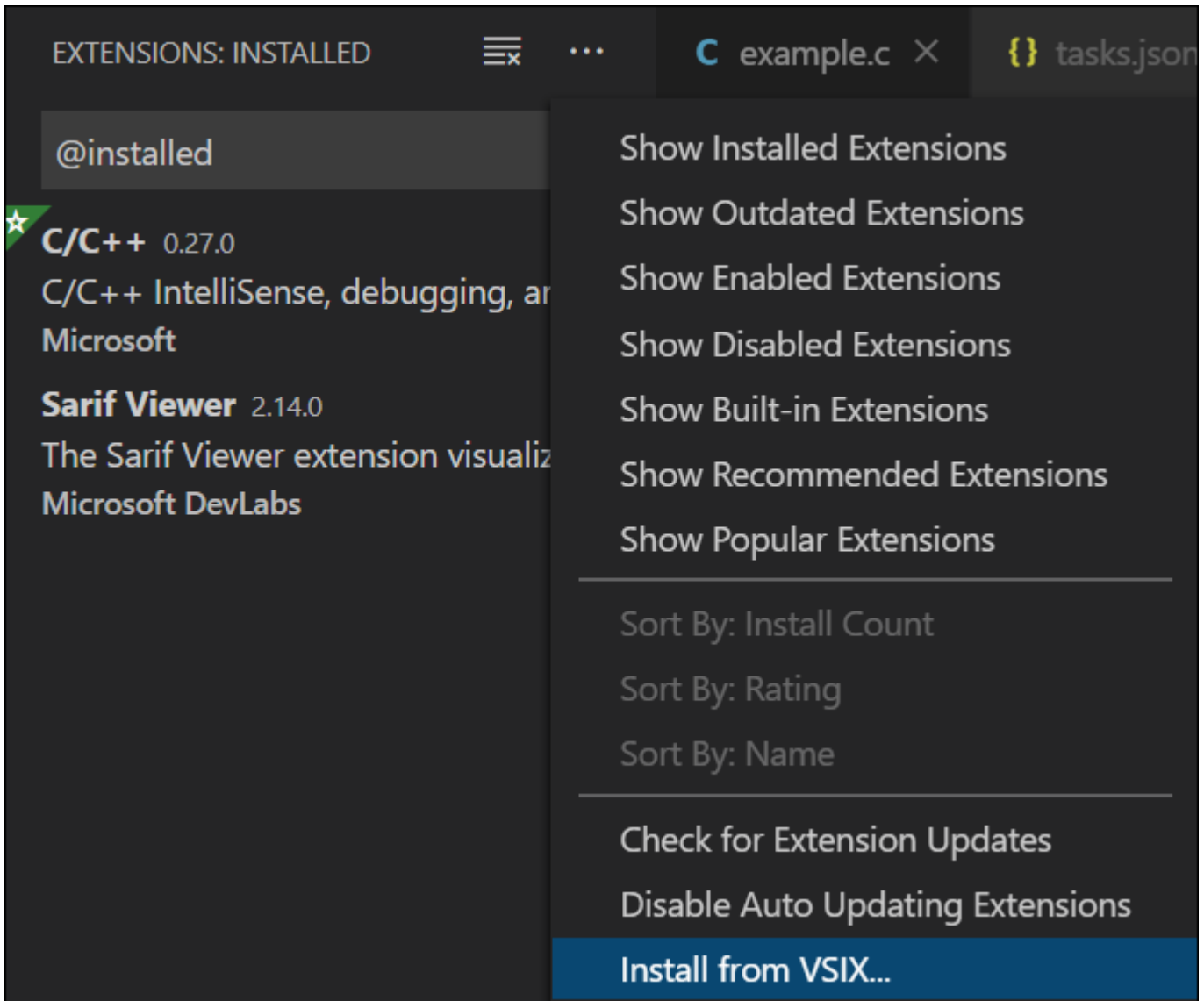
The rest of this topic describes the second approach where you skip the installation of the extension while running the installer and use the VSIX file to install the extension later.

Note If you use Visual Studio Code version 1.57.1 or later, the Polyspace as You Code extension is disabled when you open a folder in **Restricted Mode**. To enable the extension, mark the folder as trusted. See [Workspace Trust](#).

Interactive Installation

To install the extension interactively:

- 1 In the Visual Studio Code IDE, select **View > Extensions**.
- 2 On the **EXTENSIONS** pane, click the ellipsis on the upper right and select **Install from VSIX**.



- 3 Navigate to the VSIX file in the folder `polyspaceroot\polyspace\plugin\visual_studio_code`.

Here, *polyspaceroot* is the Polyspace as You Code installation folder, for instance, `C:\Program Files\Polyspace as You Code\R2023a`.

After installation, you can see the extension in the **EXTENSIONS** pane.

To uninstall the extension, on the **EXTENSIONS** pane, click the  icon and select **Uninstall**.

Command-Line Installation

You can also install the extension using the VSIX file at the command line.

- To install the extension, in a command window, enter:

```
code --install-extension polyspaceroot\polyspace\plugin\visual_studio_code\polyspace.vsix
```

- To uninstall the extension, enter:

```
code --uninstall-extension polyspace\plugin\visual_studio_code\polyspace.vsix
```

See Also

More About

- “Install Polyspace as You Code Using Installer” on page 5-2

Install Polyspace as You Code Plugin in Eclipse

This topic describes how to install the Polyspace as You Code plugin in Eclipse. For Polyspace desktop products such as Polyspace Bug Finder, see “Bug Finder Analysis Based on Eclipse Projects”.

The Polyspace as You Code plugin in the Eclipse based IDEs allows you to run Polyspace on the file that you are currently viewing and see analysis results such as bugs and coding standard violations. You must install the Polyspace as You Code analysis engine to run the analysis. The plugin allows you to point to this analysis engine from Eclipse and show results produced by the Polyspace analysis.

Note If you install the Polyspace as You Code Eclipse plugin on Linux, see “Configure Eclipse for Supported Java Version on Linux” on page 5-16.

You can install the plugin in one of two ways:

- While running the Polyspace as You Code installer, select the option to install the plugin. The installer installs the Polyspace as You Code analysis engine and the plugin.

See “Install Polyspace as You Code Using Installer” on page 5-2.

- If you install only the analysis engine while running the Polyspace as You Code installer, use the contents of the `polyspaceroot\polyspace\plugin\ eclipse` folder to install the plugin at a later time. This folder is available only after you run the installer. `polyspaceroot` is the Polyspace as You Code installation folder, for instance, `C:\Program Files\Polyspace as You Code\R2023a`.

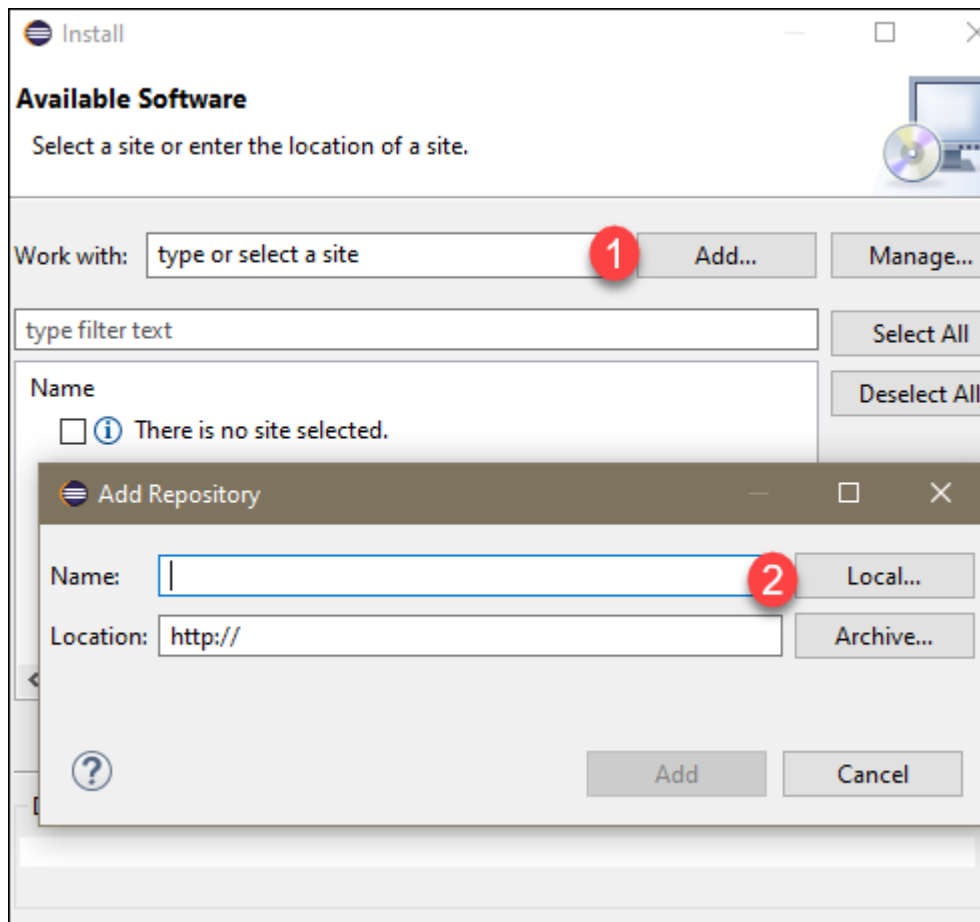
The rest of this topic describes the second approach where you skip the installation of the plugin while running the installer and use the contents of the `polyspaceroot\polyspace\plugin\ eclipse` folder to install the plugin later.

The Polyspace as You Code Eclipse plugin is not compatible with the Polyspace desktop plugin for Eclipse. Uninstall the desktop plugin before installing the Polyspace as You Code Eclipse plugin.

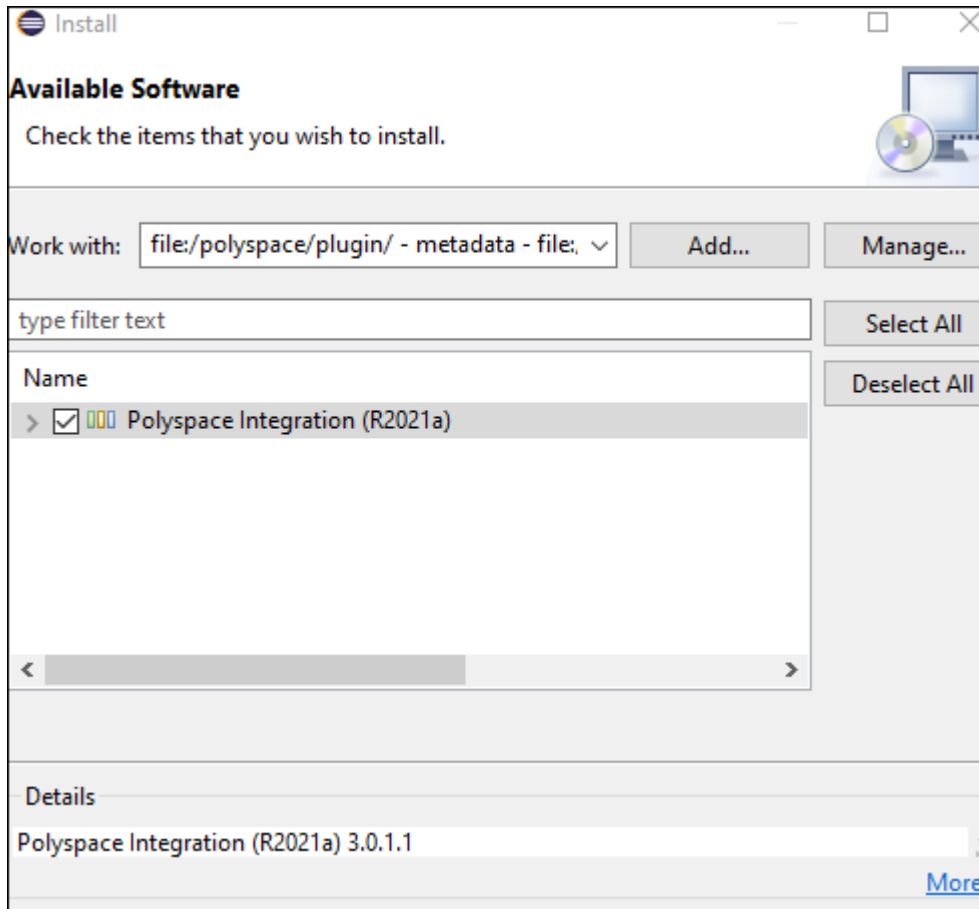
Interactive Installation

To install the plugin interactively:

- 1 In the Eclipse IDE, select **Help > Install New Software**.
- 2 In the **Install** window, click **Add**, and then click **Local** in the **Add Repository** popup.



- 3 Navigate to the *polyspaceroot\polyspace\plugin\eclipse* folder and click **Select Folder**, then add the folder to the repository.
- 4 Make sure that the Polyspace plugin is selected, then click **Next** and follow the prompts to complete the installation.



To uninstall the plugin, go to **Help > About Eclipse IDE**. In the **About Eclipse IDE** window, click **Installation Details**, select the Polyspace plugin on the **Installed Software** tab, and then click **Uninstall** and follow the prompts.

Command-line Installation

To install the plugin, open a terminal and navigate to your Eclipse installation folder; for instance `C:\Program Files\eclipse`, and enter:

```
eclipse.exe -application org.eclipse.equinox.p2.director ^
-repository file:"/polyspaceroor/polyspace/plugin/eclipse/" ^
-installIU com.polyspace.eclipse.feature -nosplash
```

where *polyspaceroor* is the Polyspace as You Code installation folder; for instance, `C:/Program Files/Polyspace as You Code`. The `-repository` path must be specified with forward slashes `"/`.

To uninstall the plugin, enter:

```
eclipse.exe -application org.eclipse.equinox.p2.director ^
-uninstallIU com.polyspace.eclipse.feature -nosplash
```

Configure Eclipse for Supported Java Version on Linux

The Polyspace as You Code Eclipse plugin does not support Java versions 13 and later. If your Eclipse IDE uses an unsupported Java version, you see an error message when you start the IDE.

To configure your IDE to use a supported Java version:

- 1 Go to <https://jdk.java.net/archive/>, download the OpenJDK version **12 GA (build 12+33)** for Linux, and then right-click the downloaded TAR file to extract its content.

Alternatively, at the command line, enter these commands:

```
wget https://download.java.net/java/GA/jdk12/33/GPL/openjdk-12_linux-x64_bin.tar.gz
tar xzvf openjdk-12_linux-x64_bin.tar.gz
```

- 2 Open file `eclipseRoot/eclipse.ini` in a text editor and replace the path below the `-vm` line with the path to the `bin` folder for the openJDK installation that you downloaded in step 1. The `eclipseRoot` folder is your Eclipse IDE installation folder.

For example, if you extracted the download from step 1 into folder `/local/tools`, the section of your `eclipse.ini` file around the `-vm` line looks similar to this section:

```
--launcher.defaultAction
openFile
--launcher.appendVmargs
-vm
/local/tools/jdk-12/bin
```

- 3 Restart your IDE to apply the changes.

See Also

More About

- “Install Polyspace as You Code Using Installer” on page 5-2

Manage Polyspace Licenses

Configure Polyspace Access License

The Polyspace Access license is a Network Named User (NNU) license that requires a license manager to manage license checkouts and an options file to specify the Named Users to whom you grant right-to-use privileges.

Warning For Enterprise license customers, the Polyspace Access user login must match the user login of the computer where Polyspace as You Code is used. If these two logins do not match, the same user checks out two different licenses when using Polyspace Access and Polyspace as You Code. For more information, contact your MathWorks sales representative.

Configure NNU License

Prerequisites

- Install the license manager. See “Install License Manager” on page 6-5.

Follow these steps to configure the Polyspace Access license. To add or remove users, see “Manage Named Users for Polyspace Access” on page 6-7.

Each licensed Polyspace Access user can log in to up to five concurrent sessions.

Note These instructions do not apply to Enterprise license customers. Contact your license administrator to configure the Polyspace Access Enterprise license.

Step 1: Configure License Manager Options File

Copy this template file to a text editor and save it as MLM.opt on the machine where you installed the license manager.

Template

```
# Options file used by MATLAB vendor daemon (MLM).
# This file contains the INCLUDE lines necessary
# for a User Based license.
# If you change the user names listed here,
# you must restart the license manager
# for the changes to take effect.
# The frequency of user name changes may be limited
# by your software license agreement.
# If you have combined multiple license files into a
# single license file, you will need to change
# the INCLUDE lines to specify a particular INCREMENT
# line. You can do this using the "featurename Key=value"
# syntax in the INCLUDE line.
# See the FLEXnet Licensing End Users Guide for
# details on how to use options files.

# Make user names and host names case insensitive when
# listed in a GROUP or HOST_GROUP. This is not
# required but it is here to prevent some common errors.
GROUPCASEINSENSITIVE ON
```

```
# Define a group of users
GROUP ACCESS_users user1 user2 user3

# Grant right-to-use privileges to individual users
INCLUDE Polyspace_BF_Access USER user1
INCLUDE Polyspace_BF_Access USER user2
INCLUDE Polyspace_BF_Access USER admin
# Grant right-to-use privileges to group of users
INCLUDE Polyspace_BF_Access GROUP ACCESS_users
```

- Use this file to identify the users to whom you grant right-to-use privileges for Polyspace Access (Polyspace_BF_Access). Users with right-to-use privileges for Polyspace Access can review or generate reports for results that were generated with a Bug Finder, Code Prover, and Ada analysis.

A user with right-to-use privileges for Polyspace Access has right-to-use privileges for Polyspace as You Code.

- For each user, enter the username that the user specifies to log into Polyspace Access. The usernames correspond to the username entries in your company LDAP server or the **User Manager** internal directory. See “Configure User Manager” on page 4-21.
- For Polyspace as You Code users, the username must also match the username used to log into the machine where the user installs and runs Polyspace as You Code.

Polyspace Access ignores any license timeout value you set in the license options file (MLM.opt) by using the syntax `TIMEOUT feature seconds`. To set the licensing timeout, use the **Authentication token expiration** setting of the **User Manager**. See “Configure User Manager” on page 4-21.

Step 2: Configure Server License

Copy your Polyspace Access license to the server machine where you installed the license manager and save it as `license.dat`. Open the file in a text editor and insert these lines at the top of the file.

```
SERVER lmHostname HostID 27000
DAEMON MLM pathTo_MLM_bin options=pathTo_MLM.opt port=27100
```

Parameter	Description
<i>lmHostname</i>	Fully qualified domain name (FQDN) of the machine where you installed the license manager. To get the FQDN, open a command-prompt window and enter: <ul style="list-style-type: none"> • Windows <pre>net config workstation findstr /C:"Full Computer name"</pre> • Linux <pre>hostname --fqdn</pre>
<i>HostID</i>	MAC address that you provided to activate the Polyspace Access license. This MAC address must match the host ID listed for Polyspace Access in the license file. <i>HostID</i> must also match a MAC address on the machine where you run the license manager.

Parameter	Description
<i>pathTo_MLM_bin</i>	Path to the MLM binary (vendor daemon). You can find this binary in <i>LM_Folder\etc\win64</i> (Windows) or <i>LM_Folder/etc/glnx64</i> (Linux), where <i>LM_Folder</i> is the folder where you installed the license manager.
<i>pathTo_MLM.opt</i>	Path to the options file that you created in step 1.

By default:

- The license manager daemon starts on port 27000. To use a different port, specify a different port number at the end of the `SERVER` line.
- If you do not specify a port on the `DAEMON MLM` line, the vendor daemon (MLM) starts on a random port chosen by your system. To specify a different port, add `port=portNumber` at the end of the `DAEMON MLM` line. For example, to start the vendor daemon on port 27100, add `port=27100`.

Specify custom ports for the license manager and vendor daemons if, for instance, you run the license manager through a firewall and you want to use ports that are open in the firewall.

If you used the MATLAB installer to install the license manager, the file `license.dat` already exists in the folder `matlabroot/etc` and the file already includes the `SERVER` and `DAEMON` lines. You might have to add the `options=pathTo_MLM.opt` instruction on the `DAEMON` line of `license.dat`. `matlabroot` is your MATLAB installation folder. Append the content of your Polyspace Access license to the `license.dat` file and go to step 3.

Step 3: Configure Client License

Copy the `SERVER` line from the `license.dat` file and paste it in a new file in a text editor. Add `USE_SERVER` below the `SERVER` line.

```
SERVER lmHostname HostID 27000
USE_SERVER
```

Save this file as `network.lic` in a location that is accessible from the machine where you installed Polyspace Access or Polyspace as You Code. This location can be on a different machine from the one where you installed the license manager.

- For the Polyspace Access web server, specify the path to this file for the **License file:** field of the **Polyspace Access Web Server** settings in the Cluster Admin web interface. See “Configure Polyspace Access App Services” on page 4-33.

Check that the docker engine can resolve the hostname `lmHostname`. In a command-prompt window, enter:

```
docker run --rm -it alpine ping lmHostname
```

If the docker engine cannot resolve this hostname, in `network.lic`, replace `lmHostname` with the IP address of the machine where you installed the license manager.

- For Polyspace as You Code, specify the path to `network.lic` when the installer prompts you to provide a license file path. See “Install Polyspace as You Code Using Installer” on page 5-2.

Step 4: Start License Manager

In a command-prompt window, navigate to the folder where you installed the license manager and start the license manager.

Windows	<pre>cd LM_Folder\etc\win64 lmgrd.exe -c pathToLicense -l lm_log.log</pre> <p>On Windows, you can also use lmtool.exe and go to the Start/Stop/Reread tab to start the license manager.</p>
Linux	<pre>cd LM_Folder/etc/glnx64 ./lmgrd -c pathToLicense -l lm_log.log</pre>

LM_Folder is the folder where you installed the license manager.

pathToLicense is the path to the `license.dat` file that you saved on the machine where you installed the license manager. The command starts the license manager and outputs a log file `lm_log.log`. Refer to this log file for debugging purposes.

Note The license file path listed in the log and error messages of the license manager might not correspond to *pathToLicense*. The **Polyspace Access Web Server** service remaps *pathToLicense* to an internal path inside the docker container.

Step 5: Configure License Manager to Start at Boot Time

After you start the license manager, check that the license manager is configured to automatically start at boot time.

Windows	Use lmtool.exe and go to the Config Services tab, then check that Start Server at Power Up and Use Services are selected.
Linux	<p>Refer to the documentation for your Linux distribution to configure the license manager to start automatically at boot time, for instance by adding a script to the <code>/etc/init.d</code> folder.</p> <p>Configure the license manager to start at the end of the boot sequence.</p>

Install License Manager

The license manager is shipped with the Polyspace Access software. The license manager binaries and utilities are located in `accessRoot/lm`. `accessRoot` is the folder where you extracted your Polyspace Access installation image.

To run the license manager on a separate server from the server where you run Polyspace Access, copy the folder that corresponds to your platform from `accessRoot/lm`, for instance `accessRoot/lm/glnxa64`, to that server.

You can also download the license manager from License Manager Download.

The license manager folder includes these binaries:

- `lmgrd`: Core license manager binary. Use this binary to start the license manager from the command line. For a list of useful commands, enter `lmgrd -h`.

- `mlm`: The MATLAB vendor daemon.
- `lmutil`: a suite of tools for administering the license manager at the command line. For a list of useful commands, enter `lmutil -h`.
- `lmtools.exe` (Windows only): Graphical user interface for administering the license manager.
- For Linux systems, the license manager folder also includes command-line utilities. See “Using Command-Line Utilities”.

To avoid any potential issues with license file operation, make sure that you run the latest license manager software version. To view the latest license manager software version available, see the FlexNet Version on this page.

To check your current license manager software version, at the command line, depending on your operating system, enter the commands listed in this table.

Windows	<code>cd LM_Folder\etc\win64</code> <code>lmgrd.exe -v</code>
Linux	<code>cd LM_Folder/etc/glnx64</code> <code>./lmgrd -v</code>

LM_Folder is the folder where you installed the license manager. See also “Update Network License Manager Software”.

See Also

Related Examples

- “Install Polyspace as You Code Using Installer” on page 5-2
- “Configure Polyspace Access App Services” on page 4-33

Manage Named Users for Polyspace Access

The Polyspace Access license is a Network Named User (NNU) license. You specify the users to whom you grant right-to-use privileges in a text file typically named `MLM.opt`. See “Configure Polyspace Access License” on page 6-2. To add or remove users from the list of Named Users associated with the Polyspace Access license, edit the list of Named Users in the `GROUP` and `INCLUDE` entries of the `MLM.opt` file.

```
Define a group of users
GROUP ACCESS_users user1 user2 user3

# Grant right-to-use privileges to individual users
INCLUDE Polyspace_BF_Access USER user1
INCLUDE Polyspace_BF_Access USER user2
INCLUDE Polyspace_BF_Access USER admin
# Grant right-to-use privileges to group of users
INCLUDE Polyspace_BF_Access GROUP ACCESS_users
```

For each user, enter the username that the user specifies to log into Polyspace Access. The usernames correspond to the username entries in your company LDAP server or the **User Manager** internal directory. See “Configure User Manager” on page 4-21.

For Polyspace as You Code users, the username must also match the username used to log into the machine where the user installs and runs Polyspace as You Code.

Then, in a command-prompt window, navigate to the folder where you installed the license manager. Stop and restart the license manager.

Note These instructions do not apply to Enterprise license customers.

Windows	<pre>cd LM_Folder\etc\win64 lmutil lmdown lmgrd.exe -c pathToLicense -l lm_log.log</pre> <p>On Windows, you can also use <code>lmtool.exe</code> and go to the Start/Stop/Reread tab to stop and restart the license manager.</p>
Linux	<pre>cd LM_Folder/etc/glnx64 ./lmutil lmdown ./lmgrd -c pathToLicense -l lm_log.log</pre>

LM_Folder is the folder where you installed the license manager.

pathToLicense is the path to the Polyspace Access license file.

Changes to the options file might be delayed by 15 minutes before they take effect.

To view the status of the license manager and see how many licenses are currently checked out, use the `lmstat` command.

Windows	<pre>cd LM_Folder\etc\win64 lmutil.exe lmstat -c pathToLicense -a</pre> <p>On Windows, you can also use <code>lmtool.exe</code> and go to the Server Status tab to stop and restart the license manager.</p>
Linux	<pre>cd LM_Folder/etc/glnx64 ./lmutil lmdown ./lmutil lmstat -c pathToLicense -a</pre>

LM_Folder is the folder where you installed the license manager. *pathToLicense* is the path to the Polyspace Access license file.

See Also

Related Examples

- “Install Polyspace as You Code Using Installer” on page 5-2
- “Configure Polyspace Access App Services” on page 4-33

Configure License Borrowing

If your Polyspace product uses a license that requires a license manager, for example a Commercial Concurrent license, you can enable license borrowing and borrow a license.

Borrow a license for up to 720 hours (30 days) to continue using your Polyspace product even after you disconnect your machine from the license manager server.

Licenses that support borrowing include the keyword `BORROW` on the `INCREMENT` lines for products, for example:

```
INCREMENT Polyspace_BF MLM 40 01-jan-0000 5 BORROW=720 \
  AB1CD234567EF6A3C24A
```

You must be an administrator to enable license borrowing. See “Enable License Borrowing”.

Borrow a Polyspace License

Note This workflow applies to R2019a and later releases. For R2018b and earlier releases, see “Borrow Licenses”.

To borrow a license, use the `lmutil` binary that is available with the license manager installation.

The binary is in the same folder as the `lmgrd` and `MLM` binaries, for instance `installFolder\win64\etc`, where `installFolder` is the license manager installation folder. If you do not know the location of this folder contact your Polyspace license administrator.

To borrow a license:

- 1 Copy the `lmutil` binary from the license server machine to the machine from which you want to borrow a license.
- 2 Use the following format to run the command that specifies the borrow period end date, and optionally, the end time:

```
lmutil lmborrow MLM dd-Mmm-YYYY hh:mm
```

For instance, to borrow a license until January 20, 2021 a 1pm, enter:

```
lmutil lmborrow MLM 20-Jan-2021 13:00
```

If you do not specify an end time, the licenses expires at midnight on the day that you specify.

- 3 Run the product whose license you want to borrow. For instance, use the Polyspace Bug Finder desktop product to analyze a small source file.

Run this command to check the borrow status of your license:

```
lmutil lmborrow -status
```

- 4 Disconnect your machine from the network and continue using your Polyspace product.

So that you do not borrow licenses for other products, after step 3, run this command:

```
lmutil lmborrow -clear
```

Return Polyspace License

To return your borrowed license before the end of the borrow period:

- 1 Connect your machine to the network of the license manager.
- 2 Use the following format to run the command that specifies the license server port and host name, and the name of the feature whose license you are returning:

```
lmutil lmborrow -return -c serverPort@serverHostname featureName
```

For instance, to return a Polyspace Bug Finder desktop license (Polyspace_BF), enter:

```
lmutil lmborrow -return -c 27000@licenseMgrHostName Polyspace_BF
```

Contact your Polyspace license administrator for the license server port and host name.

Update Polyspace

Update Polyspace Products

To install the most recent update for Polyspace desktop or server products, close all running sessions of Polyspace, then run the update installer binary. The binary is part of your original Polyspace installation. Each update includes all the fixes from previous updates.

To update the Polyspace desktop and server products, run the update installer twice, once for each Polyspace base product.

If you offload your Polyspace analysis from a client machine to a server machine, update Polyspace on both the client and server machines. The versions of Polyspace on the client and server machines must match. See also “Send Polyspace Analysis from Desktop to Remote Servers”.

Note

- This update procedure is for R2019a and later releases only.
 - To download results stored in Polyspace Access version R2022a Update 1 or later by using the `polyspace-access -download` command, update the Polyspace product that you run the command from to version R2022a Update 1 or later.
-

Install Update on Machine Connected to Internet

- 1 Open a command-line window and navigate to a folder that corresponds to your platform.

Windows	<code>polyspaceroot\bin\win64</code>
Linux	<code>polyspaceroot/bin/glnxa64</code>
macOS	<code>polyspaceroot/bin/maci64</code>

`polyspaceroot` is the Polyspace installation folder, for instance `C:\Program Files\Polyspace\R2019a`.

- 2 At the command line, enter the command that corresponds to your platform.

Windows	<code>update_installer.exe</code>
Linux/macOS	<code>./update_installer</code>

Alternatively, you can open the update installer folder in a file explorer then double-click the `update_installer` binary.

To complete the update, follow the prompts in the UI.

Install Update on Machine Without Internet

To install a Polyspace update on a machine that is not connected to Internet, download the update package to an online computer. Then, run the update installer binary on the machine that is not connected to the Internet and use the `-updatepackage` option. You can use this workflow to run the update installer without a UI, for example to install an update on a server machine without a display.

- 1 Go to https://www.mathworks.com/downloads/web_downloads/download_update_installers/R20XXy and download the update package that corresponds to your platform to an online computer.

R20XXy is the release version of the update, for instance R2023a.

- 2 Unzip the update package to a location that is accessible from the machine where you want to install the update, for example, a network drive.
- 3 On the machine where you want to install the update, open a command-line window and navigate to the folder that corresponds to your platform.

Windows	polyspaceroot\bin\win64
Linux	polyspaceroot/bin/glnxa64
macOS	polyspaceroot/bin/maci64

polyspaceroot is the Polyspace installation folder, for instance C:\Program Files\Polyspace\R2023a.

- 4 At the command line, enter the command that corresponds to your platform.

Windows	update_installer.exe -updatepackage <i>package_folder</i>
Linux/macOS	./update_installer -updatepackage <i>package_folder</i>

package_folder is the full path to the folder where you unzipped the update package. The update installer runs silently and does not output any messages.

Check Update Installer Log and Update Version

To see the installation status if you run the update installer silently, or to check for errors in the installation update process, open the installer log file that is stored in your platform temporary folder.

Windows	\%TEMP%\mathworks_ <i>yourusername</i> .log
Linux/macOS	/tmp/mathworks_ <i>yourusername</i> .log

To check the version of the Polyspace update that is installed, go to the Polyspace installation folder and open *VersionInfo.xml*. The update version is listed inside the `<description/>` tag, for example:

```
...
<release>R2023a</release>
<description>Update 4</description>
...
```

If no update is installed, the `<description/>` tag is empty.

See Also

Related Examples

- “Migrate Polyspace Projects After Product Upgrade” on page 7-4

Migrate Polyspace Projects After Product Upgrade

This topic describes how to migrate existing Polyspace projects or configurations to a new release of Polyspace. For migrating Polyspace Access results databases to a new release of Polyspace Access, see “Update or Uninstall Polyspace Access” on page 4-59.

When you upgrade your Polyspace products to a newer release, you have to migrate existing projects to this release. In general, migrating older projects to a newer release can be as simple as opening them in the user interface of the newer release. For instance, if a Polyspace analysis option has changed name between releases, the change is automatically handled in the user interface of the Polyspace desktop products.

However, each newer release of Polyspace provides new options, improved results and better user workflows. When you upgrade to a newer release, you might have to account for these changes.

Account for Changes in Options

The user interface of the Polyspace desktop products handles most changes in analysis options. However, there are certain situations where you might have to explicitly update options. For instance, if you enter a command-line-only option in the **Other** field of the analysis configuration and the option is deprecated in a later release, you have to explicitly remove the option.

If you run an analysis at the command line, you have to explicitly update options as needed.

Step 1: Rerun Analysis

To see which options have been removed in the current release (and not automatically handled in the user interface):

- 1 Open your project in the user interface of the current release.
- 2 Rerun the analysis and check the messages on the **Output Summary** pane:
 - Options that will be removed in a near release trigger a warning.
 - Options that are removed trigger an error and stop the analysis.

If you are running an analysis at the command line, the same errors and warnings appear in the analysis log. See also “View Error Information When Analysis Stops”.

Step 2: Update Options

An option might be replaced by an alternative or removed altogether because it is no longer required. An option typically changes the default behavior of the analysis. So, if the default behavior itself changes in a new release, the option might no longer be required.

Messages on the **Output Summary** pane only state if an option will be removed in a near release or has already been removed. To understand why option has been removed and what the option has been replaced with, check the product release notes:

- *Release Notes for Polyspace Bug Finder*
- *Release Notes for Polyspace Code Prover* (Polyspace Code Prover)

All changes in options appear each release in a release note entry titled **Changes in analysis options and binaries**. Check this entry to see if the option is replaced by an alternative or no longer required.

- If the option is replaced by an alternative option, change the option to the new one.
- If the option is no longer required, remove the option from your analysis configuration.

Account for Improvements in Results

Newer releases of Polyspace are, in general, more precise than previous releases. This means that for the same source files and analysis options, you might see a difference in results. For instance, a Polyspace Bug Finder analysis might no longer show a certain kind of false positive from previous releases.

Before migrating several projects, sometimes, you might want to perform a test migration of one or a few projects to gauge what result changes to expect after migration. This test migration can help you anticipate changes to expect after full migration.

Step 1: Compare Results

You can follow this process to compare the results of a project in two different releases:

- 1 Run analysis on the same project in both releases.
 - If you use the Polyspace desktop products, run analysis on the same project (.psprj file) in the user interface of the two Polyspace releases.
 - If you use the Polyspace server products, run analysis using the same scripts changing only the paths to the Polyspace binaries in the two runs.

After the analysis completes, check the log for any unexpected warnings. For instance, Polyspace might warn you that the new release contains checkers that are not available in the older release.

Note that you can compare two runs and attribute all changes to the product upgrade only if the source code and analysis options are the same. Make sure that your sources or options have not changed between the two runs. Some option changes might be necessary in the newer release but these changes can only remap older options to new ones (or remove redundant options). See “Account for Changes in Options” on page 7-4.

- 2 Compare the two sets of results and find which results have changed. To compare two sets of results in the Polyspace desktop user interface:
 - a Open the newer set of results.
 - b Select **Tools > Import Comments** and import from the previous set of results.
 - c After the import, click the **New** button.

Results that remain are the new ones in the current analysis. For details on comment import, see “Import Review Information from Previous Polyspace Analysis”.

To compare two sets of results in the Polyspace Access web interface:

- a Upload both sets of results to the same Polyspace Access project. You can upload using the `polyspace-access` command or from the Polyspace desktop user interface.

- b Use the run comparison feature in Polyspace Access to compare the two uploaded results. See “Compare Results in Polyspace Access Project to Previous Runs and View Trends”.

Step 2: Interpret Changes in Results

For the same source code and analysis configuration, you might see a change in results because of improvements to the analysis engine. See “Understanding Changes in Polyspace Results After Product Upgrade”.

To account for changes in results, you might want to do one or more of the following:

- Remove redundant code annotations.

If you had entered code annotations justifying results that no longer show up, you might want to remove the unnecessary code annotations. Code annotations that no longer apply show up as warnings in the analysis log. See “Code Annotation Warnings”.

- Update quality gates or objectives.

Sometimes, a change in checker specifications or analysis precision might affect several results of a certain type. To adapt to this change, you might have to update your quality gates or objectives. For instance, if you had earlier required that only red **Illegally dereferenced pointer** Code Prover checks must be fixed or justified, you might now want to include orange checks in your requirements.

If you are running Bug Finder, you might also want to enable new checkers after a product upgrade. These checkers might be newly available in the current release or have fewer false positives compared to previous releases. Check the **Analysis results** section of the *Release Notes for Polyspace Bug Finder* for new and updated checkers.

See Also

More About

- “Understanding Changes in Polyspace Results After Product Upgrade”
- “Import Review Information from Previous Polyspace Analysis”