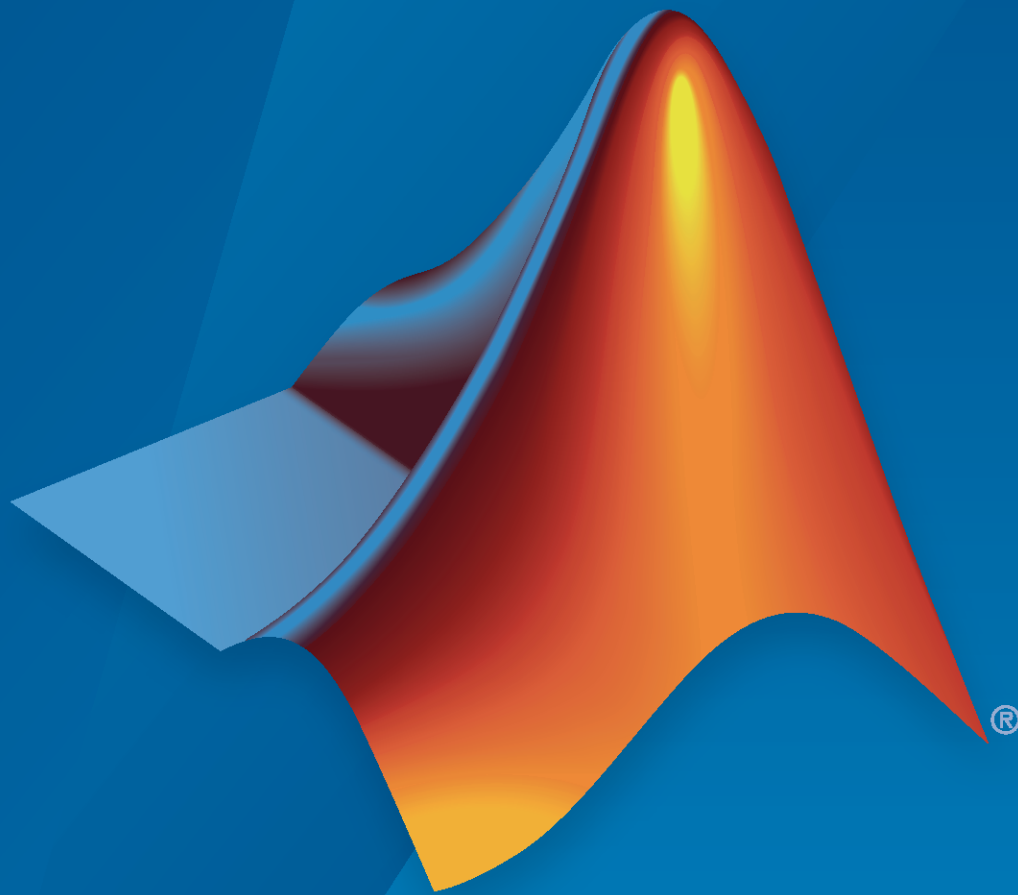


Polyspace® Code Prover™ Access™

Getting Started Guide



R2020a



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[®] Code Prover[™] Access[™] Getting Started Guide

© COPYRIGHT 2019–2020 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

March 2019	Online Only	New for Version 2.0 (Release 2019a)
September 2019	Online Only	Revised for Version 2.1 (Release 2019b)
March 2020	Online Only	Revised for Version 2.2 (Release 2020a)

1

Install Polyspace Code Prover Access

Polyspace Code Prover Access Product Description	1-2
System Requirements for Polyspace Access	1-3
Required Software	1-3
Windows Requirements	1-3
Hardware and Other Requirements	1-4
Storage Configuration	1-5
Network Port Configuration	1-6
Create a Linux Virtual Machine by Using Hyper-V	1-7
Prerequisites	1-7
Create a Virtual Machine	1-7
Start and Configure the Virtual Machine	1-8
Configure Windows Server 2019 for Polyspace Access	1-10
Prerequisites	1-10
Configure Linux Container Support	1-10
Configure and Start the Cluster Operator	1-12
Prerequisites	1-12
Unzip Installation Image and Start COP	1-13
Configure COP for HTTPS	1-14
Configure Polyspace Access Services	1-16
Prerequisites	1-16
Default Installation	1-17
Custom Installation	1-18
Configure Polyspace Access Services with HTTPS	1-19
Configure Lightweight Directory Access Protocol	1-20
Configure the User Manager and Issue Tracker	1-25
Add BTT Instance Configured with HTTPS	1-28
Configure the Database and ETL Services	1-29
Configure the Web Server and Gateway	1-30
Register Polyspace Desktop User Interface	1-33
Generate a Client Keystore	1-34

Start Polyspace Access and Upload Examples	1-35
Configure Polyspace Access to Restart Automatically	1-35
Upload Examples	1-36
Open the Polyspace Access Web Interface	1-37
Database Backup	1-39
Database Backup	1-39
Database Clean Up	1-41
Perform Database Vacuuming	1-41
Delete Outdated Projects	1-42
Update or Uninstall Polyspace Access	1-44

Manage Polyspace Access License

2

Configure Polyspace Access License	2-2
Install License Manager	2-4
Manage Named Users for Polyspace Access	2-6

Get Started with Polyspace Code Prover Access

3

Upload Results to Polyspace Access	3-2
Upload Results from Polyspace Desktop Client	3-2
Upload Results at Command Line	3-3
Open or Export Results from Polyspace Access	3-4
Open Polyspace Access Results in a Desktop Interface	3-4
Export Polyspace Access Results to a TSV File	3-4
Dashboard	3-6
Review	3-13
Manage Permissions and View Project Trends	3-16
Create a Project Folder	3-16
Manage Project Permissions	3-17
View Project Trends	3-19
Manage Results	3-21
Migrate Results from Polyspace Metrics to Polyspace Access	3-23
Requirements for Migration	3-24
Migration of Results	3-25
Differences in SQO Between Polyspace Metrics and Polyspace Access ..	3-26

Quick Start Guide for Polyspace Server and Access Products	3-28
Installation	3-29
Setting Up Polyspace Analysis	3-29

Install Polyspace Code Prover Access

- “Polyspace Code Prover Access Product Description” on page 1-2
- “System Requirements for Polyspace Access” on page 1-3
- “Storage Configuration” on page 1-5
- “Network Port Configuration” on page 1-6
- “Create a Linux Virtual Machine by Using Hyper-V” on page 1-7
- “Configure Windows Server 2019 for Polyspace Access” on page 1-10
- “Configure and Start the Cluster Operator” on page 1-12
- “Configure Polyspace Access Services” on page 1-16
- “Configure the User Manager and Issue Tracker” on page 1-25
- “Configure the Database and ETL Services” on page 1-29
- “Configure the Web Server and Gateway” on page 1-30
- “Register Polyspace Desktop User Interface” on page 1-33
- “Start Polyspace Access and Upload Examples” on page 1-35
- “Database Backup” on page 1-39
- “Database Clean Up” on page 1-41
- “Update or Uninstall Polyspace Access” on page 1-44

Polyspace Code Prover Access Product Description

Review code proving results and monitor software quality metrics

Polyspace Code Prover Access provides a web browser interface to Polyspace code verification results proving the absence of critical run-time errors in source code. It includes a central repository for analysis results that enables team-based collaboration. Results from Polyspace Code Prover Server™ can be published to Polyspace Code Prover Access for triage and resolution. With Polyspace Code Prover Access you can create and assign tickets in defect-tracking systems such as Jira.

Polyspace Code Prover Access dashboards display information that you can use to monitor software quality. The dashboards help you graphically track overall project status in terms of run-time errors and measure progress against Software Quality Objectives (SQO) thresholds.

System Requirements for Polyspace Access

Required Software

- The installation of Polyspace Access components requires Docker version 1.10 or later.
- Polyspace Access is compatible with Docker Community Edition (CE) and with Docker Enterprise Edition (EE). On Windows® systems, Polyspace Access is compatible with Docker EE only on Windows Server 2019.

To install Docker CE on your machine, click your platform and follow the installation instructions. The installation of Polyspace Access on a Linux platform is recommended.

- Windows 10
- Ubuntu
- Debian
- CentOS
- Fedora

To install Docker EE on Windows Server® 2019, see [Install Docker Engine - Enterprise on Windows Servers](#). Docker EE is also available on these platforms.

- The configuration of some Polyspace Access services requires `openssl` or a similar 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. See “Install License Manager” on page 2-4.

Windows Requirements

- To install Polyspace Access on Windows 10, or 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.

- During the installation of Docker on Windows 10, in the **Installing Docker for Windows** window, make sure that you clear the **Use Windows containers instead of Linux containers** box.
- You cannot run Docker CE as a service on Windows 10. Logging off your machine shuts down Polyspace Access. The creation of a dedicated user account to start Docker CE is recommended.
- Docker version 2.1.0.0 and later does not support Windows build 14393.

For additional information on supported Windows 10 versions and builds, see [Install Docker Desktop on Windows](#) and [Docker Release Notes](#).

Hardware and Other Requirements

- The minimum hardware configuration that is recommended for up to 100 users of Polyspace Access is:
 - 4 cores
 - 32 GB of RAM
 - 500 GB of disk space

Typically, only a reasonable fraction of users are concurrently interacting actively with the Polyspace Access web interface. To configure Polyspace Access for more than 100 users, 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 Operator does not support the Internet Explorer web browser.
- It is recommended to install Polyspace Access on physical machines. 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”

Storage Configuration

- To ensure optimal data storage performance, use physical drives instead of networked storage solutions.
- The database is stored under *polyspaceAccessRoot/polyspace/data*. *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. On Windows 10, the database folder is on the hard disk of the virtual machine running the Docker services. The default location for this virtual hard disk is C:\Users\Public\Documents\Hyper-V\Virtual Hard Disks\MobyLinuxVM.vhdx.
- 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:** 3 GB
 Uploaded files are stored in this directory while they are transferred to the web server mount point.
 - **Upload directory:** 3 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:** 15 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:** 3 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 **ETL** and the **Temporary upload directory** in the COP settings.

See Also

More About

- “Install Polyspace Access”
- “Database Backup” on page 1-39

Network Port Configuration

When you configure the Polyspace Access services, you specify port numbers for some of the services. To avoid installation errors, and to ensure that the services are accessible, make sure that the ports that you specify are open. To check whether a port *portNumber* is open, use these commands:

- On Windows: `netstat -na | find "portNumber"`
- On Linux: `netstat -na | grep portNumber`

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

This table lists the port numbers of the different services when you use the default configuration for the Polyspace Access installation.

Service	Default Port Number
Cluster Operator	8080 for HTTP configuration 8443 for HTTPS configuration
Database	5432
Gateway	9443
Issue Tracker	5002
User Manager	5001
Web Server	9444

See Also

More About

- “Install Polyspace Access”
- “Configure Polyspace Access Services” on page 1-16

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 Windows 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 on Docker community edition (CE) or Enterprise Edition (EE), see supported platforms.
- Download and install the network license manager. See “Install License Manager” on page 2-4.

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 (Polyspace Bug Finder Access) 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 (Polyspace Bug Finder Access) 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 host name for the Linux machine and a user name 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 user name `accessUser` to log into the Linux VM with host name `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” and “Manage Polyspace Access License”.

Configure Windows Server 2019 for Polyspace Access

You can install Polyspace Access on Windows Server 2019 by using Docker Enterprise Edition (EE) and by enabling Linux Containers on Windows (LCOW). You do not need to create a virtual machine in Windows Hyper-V to run LCOW.

Note The use of LCOW requires enabling experimental features in Docker. For more information, see the LCOW GitHub.

Prerequisites

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

Open Windows 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.

- Make sure that your machine meets the Docker EE system requirements. See system requirements for Windows Servers.
- Make sure that Docker EE is enabled and running as a service on your machine. To install Docker EE, in a PowerShell command prompt, enter:

```
Install-Module DockerMsftProvider -Force  
Install-Package Docker -ProviderName DockerMsftProvider -Force
```

You might need to run these commands as an administrator. For more information on the installation of docker EE on Windows Servers, see [Install Docker Engine - Enterprise on Windows Servers](#).

Configure Linux Container Support

To use LCOW with Docker EE, in a PowerShell command prompt:

- Set the LCOW_SUPPORTED environment variable to enable Linux containers.

```
[Environment]::SetEnvironmentVariable("LCOW_SUPPORTED", "1", "Machine")
```

- Enable experimental features in the Dockerd daemon configuration file.

```
$configfile = @"  
{  
  "experimental": true  
}  
"@  
$configfile|Out-File -FilePath C:\ProgramData\docker\config\daemon.json -Encoding ascii -Force
```

- Download and install the latest LCOW release. To find the version of the latest LCOW release, see [LCOW releases](#).

```
# The first command removes any existing LCOW installation  
Remove-Item "$env:ProgramFiles\Linux Containers" -Force -Recurse
```



```
Invoke-WebRequest -Uri "https://github.com/linuxkit/lcow/releases/download/v4.14.35-v0.3.9/release.zip" `
-UseBasicParsing -OutFile release.zip
Expand-Archive release.zip -DestinationPath "$Env:ProgramFiles\Linux Containers\"
rm release.zip
```

- To complete the configuration, restart your machine. Then, in a PowerShell command prompt, start the Docker service.

```
Start-Service docker
```

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

```
docker exec -it polyspace-usermanager kill 1
docker exec -it polyspace-db kill 1
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
docker exec -it polyspace-gateway kill 1
```

To install Polyspace Access, see “Install Polyspace Access” and “Manage Polyspace Access License”.

See Also

More About

- “Create a Linux Virtual Machine by Using Hyper-V” on page 1-7
- “Configure and Start the Cluster Operator” on page 1-12

Configure and Start the Cluster Operator

The Cluster Operator (COP) 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 COP, make sure that:

- You have downloaded the Polyspace Access installation image ZIP file. To download the ZIP file, go to the MathWorks® download page, click the **Download Rxxxxxy** button. You may be required to log in to your MathWorks account to complete this step. On the following page, select the Polyspace Access link under the Related Links. **Rxxxxxy** corresponds to a release number, for instance R2019b.
- You have installed the required software and that your system meets the minimum hardware requirements. See “System Requirements for Polyspace Access” on page 1-3.
- You have enough data storage available. See “Storage Configuration” on page 1-5.
- You have created and configured 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 1-7.
- You have configure Docker with Linux Containers on Windows if you are installing Polyspace Access on Windows Server 2019. See “Configure Windows Server 2019 for Polyspace Access” on page 1-10.
- Docker is running on your machine. At the command line, type:

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

If you get an error message, follow the instructions in this table.

Windows	<ul style="list-style-type: none">• Windows 10: Go to the Start menu, type <code>docker</code>, and press ENTER. If you get a sign-in request, close the sign-in window. You do not have to sign in to start Docker. Once Docker has started, from the Windows taskbar, right-click the Docker icon in the system tray and open the settings. On the Shared Drives tab, select the drives you want to share with Docker. See the Docker documentation.• Windows Server 2019: In a Powershell command prompt, enter: <code>Start-Service docker</code>
---------	--

Linux®	<p>To start Docker, run the command <code>sudo systemctl start docker</code>. If <code>systemctl</code> is not available, use <code>service</code> instead.</p> <p>After you start Docker, you must be logged in as a member of the <code>docker</code> group to run Docker commands. To see a list of current members of this group, use the command</p> <pre>grep 'docker' /etc/group</pre> <p>.</p> <p>To add the current user to the <code>docker</code> group, use the command</p> <pre>sudo usermod -aG docker \$USER</pre> <p>.</p>
--------	--

Unzip Installation Image and Start COP

The COP binary is included with the `polyspace-access-VERSION.zip` installation image for Polyspace Access. *VERSION* is the release version, for instance R2019a. After you download the installation image, unzip it to extract these files:

- `cop-docker-agent` and `cop-docker-agent.exe`.
- `polyspace-cop.tar`
- `polyspace-db.tar`
- `polyspace-etl.tar`
- `polyspace-gateway.tar`
- `polyspace-issuetracker.tar`
- `polyspace-usermanager.tar`
- `polyspace-web-server.tar`

To start the COP 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:

```
cop-docker-agent
```

If you configure the COP or Polyspace services with HTTPS, specify *hostName*, the fully qualified domain name (FQDN) of the machine where you run the COP:

```
cop-docker-agent --hostname hostName
```

For more information, see “Configure COP for HTTPS” on page 1-14 and “Configure Polyspace Access Services with HTTPS” on page 1-19. 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:

```
Cluster Operator started. You can now connect to the Cluster Operator through
your web browser at http://localhost:8080/ 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 COP. To reset the COP password, press **CTRL+C** to stop the COP binary if it is running and run this command:

```
cop-docker-agent --reset-password
```

To view the new password, restart the COP binary.

By default, the COP starts on port 8080. If this port is already in use, you get a `Permission denied` error message. To start the COP on a different port, use the flag `--port` and specify a different port number, for instance:

```
cop-docker-agent --port 9999
```

In Windows 10, if you get an error message about shared drives, make sure that you shared your drives with Docker. To see which drives are shared with Docker, right-click the Docker icon in the system tray, select settings, then click the **Shared Drives** tab. See the Docker documentation.

Once you start the COP, the agent creates a `settings.json` file and stores it in the same folder as the COP binary by default. Ensure that only the user who starts the COP has read/write permissions on this file.

Configure COP for HTTPS

To encrypt the data between the COP server and client machines, configure the COP with the HTTPS protocol. To complete the configuration, you must provide a certificate and the private key you used to generate the certificate as PEM files.

Do not reuse the private key file you use for the **Authentication private key file**: in the **User Manager** configuration.

It is recommended to use a certificate issued by a certificate authority to configure HTTPS. If you do not want to use a certificate authority, you can configure HTTPS by using self-signed certificates. By default, the COP starts on port 8443 when you enable HTTPS.

Secure your COP 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 COP administrators.
- Rotate your private key and certificate regularly (annually) and audit which users have access to the private key file.

The configuration of HTTPS for the COP does not enable HTTPS for the Polyspace Access web interface.

Use Certificates Signed by a Certificate Authority

These steps illustrate how to configure SSL encryption on a Debian Linux system by using your organization's certificate authority and the `openssl` utility.

- 1 Create a certificate signing request. In the CN field (common name), specify *hostName*, the fully qualified domain name (FQDN) of the machine where you run the COP.

```
openssl req -new -newkey rsa:4096 -nodes -out myRequest.csr -keyout myKey.key \  
-subj "/C=US/ST=/L=/O=/CN=hostName"
```

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

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

- 3 Start the COP by using the generated private key and signed certificate:

```
./cop-docker-agent --hostname hostName\
--https-certificate-file fullPathTo/cop_cert.cer \
--https-private-key-file fullPathTo/myKey.key \
--https-trusted-certificates-file /etc/ssl/certs/ca-certificates.crt
```

The *hostName* you specify in this command must match the *hostName* you specified in step 1. *fullPathTo* is the full file path. When you open the COP web interface, your browser considers the connection secure if the browser uses the certificate store that you specify for **Trusted certificates file**.

Use Self-Signed Certificates

To configure HTTPS on a Debian Linux system by using a self-signed certificate that you generate with `openssl`, follow these steps:

- 1 Generate a certificate and private key as PEM files. In the CN field (common name), specify *hostName*, the fully qualified domain name (FQDN) of the machine where you run the COP.

```
openssl req -newkey rsa:2048 -new -nodes -x509 -days 365 -keyout private_key.pem \
-out certificate.pem -subj "/C=US/ST=/L=/O=/CN=hostName"
```

- 2 Start the COP by using the generated `certificate.pem` and `private_key.pem` files.

```
./cop-docker-agent --hostname hostName\
--https-certificate-file fullPathTo/certificate.pem \
--https-private-key-file fullPathTo/private_key.pem \
--https-trusted-certificates-file fullPathTo/certificate.pem
```

The *hostName* you specify in this command must match the *hostName* you specified in step 1. *fullPathTo* is the full file path. If you use relative paths, you get an error message.

See Also

`cop-docker-agent`

More About

- “System Requirements for Polyspace Access” on page 1-3
- “Storage Configuration” on page 1-5
- “Configure Polyspace Access Services” on page 1-16

Configure Polyspace Access Services

To set up the Polyspace Access centralized database, web server, issue tracker, and user authentication, you install these services:

- **User Manager:** To authenticate user logins and issue signed JSON Web Tokens to authenticated users.
- **Issue Tracker:** To manage the communication between Polyspace Access and your bug tracking tool software.
- **Database and extract-transform-load (ETL):** To upload and manage results on the database.
- **Web Server:** To provide a user interface that you can open in a web browser.
- **Gateway:** To handle all communications between clients and the other Polyspace Access services.

Prerequisites

Before you begin configuring the Polyspace Access services, make sure that you have started the cluster operator. See “Configure and Start the Cluster Operator” on page 1-12 (COP).

After you configure and start COP, open your web browser and go to URL specified in the command-line output when you started COP.

Log in with the initial password that you obtained when you started the COP agent. If this time is your first time logging in, follow the prompts. Then, on the installation wizard page, select an installation method.

Welcome to the Polyspace Access Installation Wizard

Choose one of the following installation methods:

- Default single-machine setup**
Use this method if you want to install all Polyspace Access services on the machine you started this agent using default settings. If necessary, you can change the settings later.
- Custom and/or multi-machine setup (Advanced)**
Use this method if you want to install Polyspace Access services across multiple machines and/or want to use custom settings. This method will redirect you to the Settings page where you can set up your Polyspace Access cluster.

Apply

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

Note

- 1 Whenever you change the settings, click **Save**, then in the **Services** tab click **PROVISION** for the changes to take effect.
 - 2 On Windows systems, all the file paths in the **Settings** must point to local drives.
-

Default Installation

The default installation for Polyspace Access uses a configuration that enables you to install and run all the services on the machine where COP is running. The default settings use the HTTPS protocol and the embedded Lightweight Directory Access Protocol (LDAP) for user authentication. See “Configure Polyspace Access Services with HTTPS” on page 1-19 and “Use the Polyspace Access Embedded LDAP” on page 1-22. Complete the configuration of the **User Manager** on page 1-25, **Issue Tracker** on page 1-27, **Web Server** on page 1-30, and **Gateway** on page 1-31 services before you start Polyspace Access.

Polyspace Access Cluster Operator

Settings

Services

Nodes

Settings

Database

Node: master ▼

Data volume: polyspace-data ▼ [Create volume](#)

Port number: 5432

ETL

Node: master ▼

Storage directory: /tmp/polyspace/storage

Invalid results directory: /tmp/polyspace/corrupted

Working directory: /tmp/polyspace/working

Upload directory: /tmp/polyspace/upload

User Manager

Node: master ▼

Gateway

Node: master ▼

Port number: 9443

Use HTTPS protocol:

Certificate file: /local/ACCESS/certificates/base_64_web_server.cer

Certificate private key file: /local/ACCESS/certificates/myKey.key

Trusted certificates file: /etc/ssl/certs/ca-certificates.crt

[Save](#)

Custom Installation

To specify different data volume, port numbers, and folder paths for the different services, select a custom installation on the installation wizard page.

The paths of the **Upload directory** setting for the **ETL** and **Web Server** services must point to the same hard drive.

Complete the configuration of the **User Manager** on page 1-25, **Issue Tracker** on page 1-27, **Web Server** on page 1-30, and **Gateway** on page 1-31 services before you start Polyspace Access.

Configure Polyspace Access Services with HTTPS

The HTTPS protocol is enabled by default to allow the encryption of data transfers between Polyspace Access and client machines, and between the different Polyspace Access services.

When you configure the Polyspace Access services for HTTPS, you provide a certificate and the private key that you used to generate that certificate. It is recommended that you secure your Polyspace Access services private keys 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 to only those Polyspace Access administrators who manage the services.
- Establish a policy to periodically check which users have access to the private key file.

To configure the HTTPS protocol for the **User Manager** on page 1-25, **Issue Tracker** on page 1-27, **Web Server** on page 1-30, and **Gateway** on page 1-31 services, complete these fields.

- **Certificate file:** Full path to the signed server certificate PEM file. On Windows systems, the path must point to a local drive.
- **Certificate private key file:** Full path to the private key PEM file you used to generate the certificate file. On Windows systems, the path must point to a local drive.

Do not reuse the private key file you use for the **Authentication private key file** in the **User Manager** configuration.

- **Trusted certificates file:** Full path to the certificate store where you store trusted certificate authorities. For instance, on a Linux Debian distribution, `/etc/ssl/certs/ca-certificates.crt`. If you use self-signed certificates, use the same path you specify for **Certificate file**.

If you do not want to use HTTPS, clear **Use HTTPS protocol** for the **User Manager**, **Issue Tracker**, **Web Server**, and **Gateway** services.

Use Certificates Signed by a Certificate Authority

These steps illustrate how to configure SSL encryption on a Debian Linux system by using your organization's certificate authority and the `openssl` utility.

- 1 Create a certificate signing request. In the CN field (common name), specify *hostName*, the fully qualified domain name (FQDN) of the machine where you run the Polyspace Access service you are configuring. This must match the host name you specified when you started the `cop-docker-agent` binary.

```
openssl req -new -newkey rsa:4096 -nodes -out myRequest.csr -keyout myKey.key \
-subj "/C=US/ST=/L=/O=/CN=hostName"
```

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

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

- 3 Complete the configuration using the private key file, the signed server certificate, and the certificate store where you store the certificate that identifies your certificate authority.

Certificate file:	Full path to the server certificate you obtained in step 2.
Certificate private key file:	Full path to the private key file you generated in step 1.
Trusted certificates file:	/etc/ssl/certs/ca-certificates.crt

When you open Polyspace Access in a web browser, it considers the connection secure if the browser uses the certificate store that you specify for **Trusted certificates file:**.

Use Self-Signed Certificates

Alternatively, use self-signed certificates. Because the identity of your server is not certified by a certificate authority, your browser might consider the connection to your server as untrusted. To generate a self-signed certificate with `openssl`, use this command:

```
openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:4096 -keyout self-key.pem \
-out self-cert.pem -subj "/C=US/ST=/L=/O=/CN=hostName"
```

In the CN field (common name), specify *hostName*, the fully qualified domain name (FQDN) of the machine where you run the Polyspace Access service you are configuring. This must match the host name you specified when you launched the `cop-docker-agent` binary.

The command outputs files `self-key.pem` and `self-cert.pem`. Use these files to complete the configuration.

Certificate file:	<i>fullPathTo/self-cert.pem</i>
Certificate private key file:	<i>fullPathTo/self-key.pem</i>
Trusted certificates file:	<i>fullPathTo/self-cert.pem</i>

fullPathTo is the full file path.

Configure Lightweight Directory Access Protocol

Use Your Organization LDAP

Polyspace Access authenticates users by checking for valid user names and passwords against information from the LDAP server. To use the LDAP server of your organization, clear **Use embedded LDAP** in the **COP User Manager** settings. Contact your network administrator for the LDAP URL, base, and any other setting.

LDAP URL	<p>You must enter the LDAP URL as <code>ldap://HOST:PORT</code>, where <i>HOST</i> is the LDAP host. If you have configured your LDAP server over SSL, enter the URL as <code>ldaps://HOST:PORT</code>. For additional LDAPS configuration steps, see “Configure the User Manager for LDAP over SLL” (Polyspace Bug Finder Access).</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>
LDAP username:	User name of user with read permission to the LDAP server. Leave this field blank if your access to the LDAP server is not password-protected.
LDAP password:	<p>Password of user with 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 COP binary.</p>
LDAP base	You can retrieve this parameter by using an LDAP explorer tool. For instance, connect to your LDAP server with Apache Directory Studio and open the properties for your connection. In the Browser Options , click Fetch Base DNs to get the LDAP base.
LDAP search filter:	<p>Use the search filter to retrieve a subset of users from the LDAP database. Polyspace Access loads this subset on start up instead of 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 <code>attribute=value</code>, for instance <code>CN=test*</code> matches all users that have a common name (CN) attribute that starts with "test".</p> <p>Use parentheses to combine multiple filter expression in an AND (&) or OR() clause. For instance <code>((CN=jdoe)(department=foo))</code> matches all users that have CN attribute "jdoe" or department attribute "foo".</p> <p>The default search filter is <code>objectClass=organizationalPerson</code>. For more information on search filters, see the LDAP filters.</p>
Other LDAP settings	Leave these settings unchanged unless instructed otherwise by your LDAP administrator.

Configure the User Manager for LDAP over SLL

If you use a LDAP configured over SSL (LDAPS), add the LDAPS certificate to the **Trusted certificates file**: you specify in COP settings for the **User Manager** service.

For instance, on a Linux Debian distribution:

- If you configure the **User Manager** service for HTTPS with a certificate authority:
 - If the LDAPS certificate is already part of the certificate trust store file, and you point to `/etc/ssl/certs/ca-certificates.crt` in the COP settings, no action is required.
 - If the LDAPS certificate is not part of the certificate trust store file, concatenate `/etc/ssl/certs/ca-certificates.crt` and the LDAPS certificate and point to the resulting file in

Trusted certificates file: For example, for a LDAPS certificate file `ldaps_cert.pem`, use this command.

```
cat /etc/ssl/certs/ca-certificates.crt \  
/local/ldaps/ssl/ldaps_cert.pem > \  
/local/access/ssl/trusted-certificates.pem
```

Point to `/local/access/ssl/trusted-certificates.pem` for the **Trusted certificates file:** setting.

- If you configure the **User Manager** service using self-signed certificates:
 - Concatenate the certificate you use to configure HTTPS for the **User Manager** service with the LDAPS certificate, or with `/etc/ssl/certs/ca-certificates.crt` if the LDAPS certificate is in the certificates trust store. For example, for a LDAPS certificate file `ldaps_cert.pem`, and a self-signed certificate `self-cert.pem`, use this command.

```
cat /local/access/self_cert.pem \  
/local/ldaps/ssl/ldaps_cert.pem > \  
/local/access/ssl/trusted-certificates.pem
```

Point to `/local/access/ssl/trusted-certificates.pem` for the **Trusted certificates file:** setting.

- If you do not configure the **User Manager** service for HTTPS, you must still point to the LDAPS certificate in the **Trusted certificates file:** setting, or to `/etc/ssl/certs/ca-certificates.crt` if the LDAPS certificate is in the certificates trust store .

Use the Polyspace Access Embedded LDAP

If you cannot or choose not to use the LDAP server of your organization, you can authenticate user logins by using the Polyspace Access embedded LDAP. Select **Use embedded LDAP** in the COP settings. The embedded LDAP checks user logins against credentials that are stored in an LDIF file.

This example shows how to create an LDIF file and add new users.

- 1 Copy the template file to a text editor and save it on your system as `embedded_ldap.ldif`

Template file

```
dn: dc=customer,dc=mathworks,dc=com  
objectclass: top  
objectclass: domain  
objectclass: extensibleObject  
dc: customer  
  
## BEGIN entry  
dn: uid=admin,dc=customer,dc=mathworks,dc=com  
objectclass: inetOrgPerson  
# 'common' name  
cn: admin  
# user id  
uid: admin  
mail: admin@invalid.com  
userPassword: pass  
displayName: admin  
# surname  
sn: admin  
##END entry, leave one empty line before next entry
```

The template defines a single user with user name `admin` and password `pass`.

- 2 To create a user, copy all the lines from `##BEGIN` entry to `##END` entry... and paste them below the last entry in the file.
- 3 Enter a new user name in the fields for `dn: uid=`, `cn:` , and `uid:` , and provide a new password. For instance, the entry for user `jsmith` with password `new_pass` is:

```
## BEGIN entry
dn: uid=jsmith,dc=customer,dc=mathworks,dc=com
objectclass: inetOrgPerson
# 'common' name
cn: jsmith
# user id
uid: jsmith
mail: jsmith@invalid.com
userPassword: new_pass
displayName: John
# surname
sn: Smith
##END entry, leave one empty line before next entry
```

You can optionally provide values for the `mail`, `displayName`, and `sn` fields.

- 4 Complete the configuration in COP.

LDIF file:	<code>fullPathTo/embedded_ldap.ldif</code> <i>fullPathTo</i> is the full file path. On Windows systems, the path must point to a local drive.
LDAP base	<code>dc=customer,dc=mathworks,dc=com</code>
Other LDAP settings	Leave these settings unchanged unless you are familiar with LDIF file templates.

The passwords you store in the LDIF file are not encrypted. Restrict read/write permissions for this file to only the administrator who edits the file.

Manage LDAP Users in Polyspace Access

When the **User Manager** service starts, Polyspace Access loads a list of authorized users to its database from your organization's LDAP database or, if you use the embedded LDAP, from the LDIF file that you specify. You can select users only from the list of authorized users when assigning analysis findings or managing permission for a project.

Polyspace Access does not update its database of authorized users if you remove a user from the LDAP database or from the LDIF file, even if you restart the **User Manager** service.

To update the list of authorized users in Polyspace Access, follow these steps. You must be logged in as a user with **Administrator** privileges. To set a user as **Administrator**, see “Configure the Web Server and Gateway” on page 1-30.

- 1 Restart the **User Manager** service to load the most recent list of authorized users. At the command line, enter:


```
docker restart polyspace-usermanager
```
- 2 In a web browser, enter the URL that you use to “Open the Polyspace Access Web Interface” on page 1-37 and append `/users/list/removed` to the URL, for example `https://access-machine.company.com:9443/users/list/removed`.
- 3 Select the user names that you want to remove from the Polyspace Access database of authorized users and click **Confirm clean-up**. To select multiple users, hold down the **CTRL** key while you click.

See Also

More About

- “Start Polyspace Access and Upload Examples” on page 1-35

Configure the User Manager and Issue Tracker

User Manager

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Port number:	See “Network Port Configuration” on page 1-6.
Use HTTPS protocol:	This option is enabled by default. See “Configure Polyspace Access Services with HTTPS” on page 1-19. Use the same HTTPS protocol settings for the User Manager , Issue Tracker , Web Server , and Gateway services.
Certificate file:	If you select Use HTTPS protocol , these fields are required. See “Configure Polyspace Access Services with HTTPS” on page 1-19.
Certificate private key file:	
Trusted certificates file:	
Use embedded LDAP:	This option is enabled by default. See “Use the Polyspace Access Embedded LDAP” on page 1-22. If you disable this option, see “Use Your Organization LDAP” on page 1-20
LDIF file: or LDAP URL:	To configure these settings, see “Use the Polyspace Access Embedded LDAP” on page 1-22 if Use embedded LDAP is enabled or “Use Your Organization LDAP” on page 1-20 otherwise.
LDAP base:	
LDAP search filter:	
Other LDAP settings	
Authentication token expiration (sec):	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 in to 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 crashes, your license stays checked out until the session expires. When you set the expiration time, consider: <ul style="list-style-type: none"> • If the expiration time is too short, frequent users will be 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.
Authentication issuer:	Service that issues the signed JSON Web Tokens.

Setting	Description
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 you use to generate the SSL certificates you provide if you enable the HTTPS protocol.</p>
Authentication public key file:	<p>Specify the full path to the public key PEM file that the User Manager uses to sign JSON Web Tokens. You generate this file from the private key file that you specify for Authentication private key file:. On Windows systems, the paths must point to local drives.</p> <p>You can generate a public key pair from private key <code>private.pem</code> by using the <code>openssl</code> utility. For example:</p> <pre>openssl rsa -in private.pem -outform PEM -pubout -out public.pem</pre> <p>Do not reuse the private key you use to generate the SSL certificates you provide if you enable the HTTPS protocol.</p>

Configure the **Issue Tracker** if you want to enable the creation of tickets in your bug tracking tool (BTT) from the Polyspace Access interface.

Issue Tracker

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Port number:	See “Network Port Configuration” on page 1-6.
Use HTTPS protocol:	This option is enabled by default. See “Configure Polyspace Access Services with HTTPS” on page 1-19. Use the same HTTPS protocol settings for the User Manager , Issue Tracker , Web Server , and Gateway services.
Certificate file:	If you select Use HTTPS protocol , these fields are required. See “Configure Polyspace Access Services with HTTPS” on page 1-19.
Certificate private key file:	
Trusted certificates file:	
Provider	Specify your bug tracking tool (BTT) provider. Polyspace Access supports integration with the Jira Software and Redmine BTT.
URL:	Specify the URL of the BTT instance for your organization, for example <code>https://jira.mycompany.com</code> . If your BTT instance is configured with HTTPS, see “Add BTT Instance Configured with HTTPS” on page 1-28. Note Polyspace Access does not support integration with bug tracking tools that are hosted in the cloud.
API key:(Redmine only)	Specify the API access key of a redmine administrator. 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 . COP does not validate the API key. Check periodically that the API key has not expired or become invalid.

Limitations

- Polyspace Access does not support the creation of BTT tickets with custom fields if these fields are required fields.
- To create a redmine ticket from Polyspace Access, the user name 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.
- 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, dark feature should not be enabled. See Enable Dark Feature in Jira.

Add BTT Instance Configured with HTTPS

If your BTT instance is configured with HTTPS, add the BTT certificate to the **Trusted certificates file**: you specify in COP settings for the **Issue Tracker** service.

Note Polyspace Access does not support integration with bug tracking tools that are hosted in the cloud.

For instance, on a Linux Debian distribution:

- If you configure the **Issue Tracker** service for HTTPS with a certificate authority:
 - If the BTT certificate is already part of the certificate trust store file, and you point to `/etc/ssl/certs/ca-certificates.crt` in the COP settings, no action is required.
 - If the BTT certificate is not part of the certificate trust store file, concatenate `/etc/ssl/certs/ca-certificates.crt` and the BTT certificate and point to the resulting file in **Trusted certificates file**:. For example, for a JIRA certificate file `jira_cert.pem`, use this command.

```
cat /etc/ssl/certs/ca-certificates.crt \  
    /local/jira/ssl/jira_cert.pem > \  
    /local/access/ssl/trusted-certificates.pem
```

Point to `/local/access/ssl/trusted-certificates.pem` for the **Trusted certificates file**: setting.

- If you configure the **Issue Tracker** service using self-signed certificates:
 - Concatenate the certificate you use to configure HTTPS for the **Issue Tracker** service with the BTT certificate, or with `/etc/ssl/certs/ca-certificates.crt` if the BTT certificate is in the certificates trust store. For example, for a JIRA certificate file `jira_cert.pem`, and a self-signed certificate `self-cert.pem`, use this command.

```
cat /local/access/self_cert.pem \  
    /local/jira/ssl/jira_cert.pem > \  
    /local/access/ssl/trusted-certificates.pem
```

Point to `/local/access/ssl/trusted-certificates.pem` for the **Trusted certificates file**: setting..

- If you do not configure the **Issue Tracker** service for HTTPS, you must still point to the BTT certificate in the **Trusted certificates file**: setting, or to `/etc/ssl/certs/ca-certificates.crt` if the BTT certificate is in the certificates trust store .

See Also

More About

- “Configure the Database and ETL Services” on page 1-29
- “Configure the Web Server and Gateway” on page 1-30
- “Register Polyspace Desktop User Interface” on page 1-33
- “Start Polyspace Access and Upload Examples” on page 1-35

Configure the Database and ETL Services

Database

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Data volume:	<ul style="list-style-type: none"> • Linux: Specify the full path to the folder where you store the database or specify a volume name. • Windows 10: Select an existing data volume from the dropdown or create a volume for the database mount point. • Windows Server 2019: Specify the full path to the folder where you store the database. <p>Deleting the 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, click Nodes in the left pane, click the appropriate node ID, select the volume, and then click DELETE. You can also manually delete the folder where you store the database.</p>
Port number:	See “Network Port Configuration” on page 1-6.

ETL

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Storage directory:	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See “Storage Configuration” on page 1-5.
Invalid results directory:	
Working directory:	
Upload directory:	The Upload directory: path must be the same for the Web Server and ETL services if they are running on the same machine. If the services are running on different machines, the paths must point to the same hard drive.

See Also

More About

- “Configure the User Manager and Issue Tracker” on page 1-25
- “Configure the Web Server and Gateway” on page 1-30
- “Register Polyspace Desktop User Interface” on page 1-33
- “Start Polyspace Access and Upload Examples” on page 1-35

Configure the Web Server and Gateway

Web Server

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Port number:	See “Network Port Configuration” on page 1-6.
Use HTTPS protocol:	This option is enabled by default. See “Configure Polyspace Access Services with HTTPS” on page 1-19. Use the same HTTPS protocol settings for the User Manager , Issue Tracker , Web Server , and Gateway services.
Certificate file:	If you select Use HTTPS protocol , these fields are required. See “Configure Polyspace Access Services with HTTPS” on page 1-19.
Certificate private key file:	
Trusted certificates file:	
Upload directory:	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See “Storage Configuration” on page 1-5. The Upload directory : path must be the same for the Web Server and ETL services if they are running on the same machine. If the services are running on different machines, the paths must point to the same hard drive.
Temporary upload directory:	
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 2-2. On Windows systems, the paths must point to local drives.

Setting	Description
Authorization administrators:	<p>Enter a comma-separated list of user names to set those users as administrators. A user with administrator privileges can:</p> <ul style="list-style-type: none"> • Assign or unassign other users as administrators and manage permissions for all projects from the Polyspace Access web interface. See “Manage Permissions in Polyspace Access Web Interface” (Polyspace Bug Finder Access). • Remove or add project owners. • “Manage LDAP Users in Polyspace Access” on page 1-23 <p>To remove an administrator:</p> <ol style="list-style-type: none"> 1 Remove the user name from the list, save your changes, then provision and restart Polyspace Access. 2 After the restart, a Polyspace Access administrator must unassign the user from all top-level folders in the PROJECT EXPLORER in the 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 -unset-role -h</code>.

Gateway

Setting	Description
Node:	Use the drop-down list to select the node on which you run this service. To create nodes and run the service on a different machine, see “Custom Installation” on page 1-18.
Port number:	See “Network Port Configuration” on page 1-6. This number is the port number that you specify in the URL you use to open the Polyspace Access web interface.
Use HTTPS protocol:	This option is enabled by default. See “Configure Polyspace Access Services with HTTPS” on page 1-19. Use the same HTTPS protocol settings for the User Manager , Issue Tracker , Web Server , and Gateway services.
Certificate file:	If you select Use HTTPS protocol , these fields are required. See “Configure Polyspace Access Services with HTTPS” on page 1-19.
Certificate private key file:	
Trusted certificates file:	

See Also

More About

- “Configure the User Manager and Issue Tracker” on page 1-25
- “Configure the Database and ETL Services” on page 1-29

- “Register Polyspace Desktop User Interface” on page 1-33
- “Start Polyspace Access and Upload Examples” on page 1-35

Register Polyspace Desktop User Interface

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

Polyspace Preferences

Tools Menu | Review Statuses | Miscellaneous | Character Encoding | Review Scope

Server Configuration | Project and Results Folder | Editors

MATLAB Parallel Server cluster configuration

Job scheduler host name: localhost | Parallel computing username: jsmith

Localhost IP address:

Use Polyspace Metrics

Use Polyspace Access

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 web interface configuration

Polyspace Access URL: https://access-machine-deb9-64:9443/

For https protocol

Client keystore path: C:\users\jsmith\certificates\client-cert.jks

Client keystore password: ●●●●●●

Enable the launching of this desktop UI from the Polyspace Access web interface:

Register Polyspace UI

OK | Apply | Cancel

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

- **Polyspace Access URL:** `https://hostName:port`, where *hostName* is the fully qualified domain name (FQDN) of the machine on which the **Gateway** service is running. *port* is the **Port number** value specified in the **Gateway** COP settings.

If you did not select **Use HTTPS protocol** in the **Gateway COP** settings, replace `https` with `http`.

- **Client keystore path:** path to the key store where you imported the **Gateway** signed certificate. See “Generate a Client Keystore” on page 1-34.

If you did not select **Use HTTPS protocol** in the **Gateway COP** settings, leave this field blank.

- **Client keystore password:** The password associated with the key store file.

If you did not select **Use HTTPS protocol** in the **Gateway COP** settings, leave this field blank.

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

Generate a Client Keystore

If you configure the Polyspace Access **Gateway** service to use the HTTPS protocol, you must generate a Java® Key Store (JKS) file to enable communications between Polyspace Access and the desktop interface or the `polyspace-report-generator` binary. You import the signed certificate that you used to configure HTTPS for the **Gateway** on page 1-30 service to the JKS file you generate.

To generate the `jks` file, use the `keytool` key and certificate management utility. To use `keytool` you must have Java Platform, Standard Edition Development Kit (JDK) installed on your machine. `keytool` is available from the Java installation folder, for instance:

- **Windows:** `C:\Program Files\Java\jdk1.8.0_181\bin\keytool.exe`
- **Linux:** `/usr/bin/keytool` or `%JAVA_HOME%/bin/keytool`

For example, if you used signed certificate `gateway-cert.cer` to configure HTTPS for the **Gateway** service, generate the corresponding JKS file by using this command:

```
keytool -import -trustcacerts -alias cert -file gateway-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`.

See Also

More About

- “Upload Results from the Desktop Interface” on page 1-36

Start Polyspace Access and Upload Examples

To complete the installation, start the Polyspace Access services and upload some examples to the Polyspace Access database. Before you begin, make sure that you have configured the Polyspace Access services. See “Configure Polyspace Access Services” on page 1-16.

After you enter the settings parameters, click **Save**. From the left pane, go to the **Services** view and click **PROVISION**. COP loads and installs Docker images for the **User Manager**, **Issue Tracker**, **Database**, **ETL**, **Web Server**, and **Gateway** services. If the installation is successful, all the services are stopped and show a red indicator.

Polyspace Access Cluster Operator			
Services			
🔧 PROVISION ▶ START ALL ■ STOP ALL 🗑 DELETE ALL			
Database	●	Stopped	Start
ETL	●	Stopped	Start
User Manager	●	Stopped	Start
Web Server	●	Stopped	Start
Gateway	●	Stopped	Start

To start the services, click **START ALL**. The indicator turns green when a service starts. The **Web Server** might take a few moments to start even after the indicator turns green.

Once you complete the installation, you can stop the COP agent at the command line by pressing **Ctrl +C**.

Note If one of the services starts and stops after a short time, try restarting the service and, at the command line, enter:

```
docker logs -f polyspace-service
```

service corresponds to db (database), etl, usermanager, issuetracker, web-server, or gateway. Use the output log to try to identify the cause of the stopped service.

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, at the command line, enter:

```
docker update --restart always polyspace-gateway polyspace-web-server polyspace-etl
docker update --restart always polyspace-db polyspace-usermanager polyspace-issuetracker
```

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.

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

```
docker exec -it polyspace-usermanager kill 1
docker exec -it polyspace-db kill 1
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
docker exec -it polyspace-gateway kill 1
```

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 hosting the **Gateway** service. *port* is the **Port number** value specified for the **Gateway** setting in COP. For more information on uploading results from the command line, see the documentation for Polyspace Bug Finder Server or Polyspace Code Prover Server.

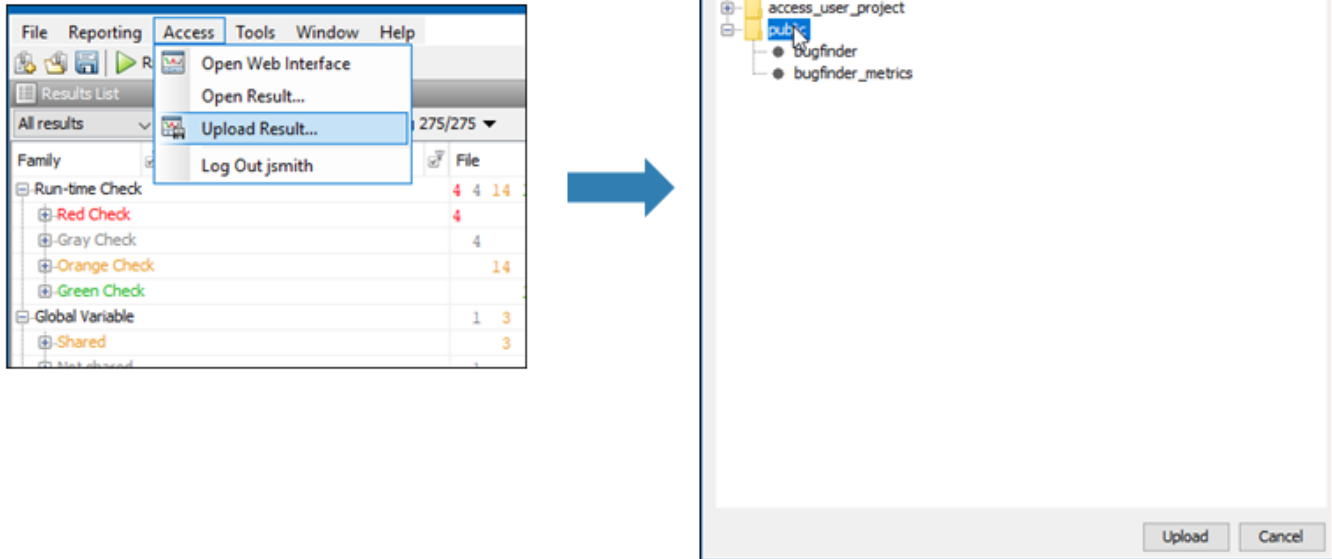
After each command, you are prompted to enter your user name and password. Enter the credentials that you use to log in to Polyspace Access.

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 1-33.

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

You can now open Polyspace Access from any machine connected to the server hosting the Polyspace **Gateway** service. If you have enabled the HTTPS protocol, from your web browser, go to `https://hostName:port`

hostName is the fully qualified domain name (FQDN) of the web server host. *port* is the **Port number** value specified in the COP settings for the **Gateway** service. For instance, `https://access-machine.company.com:9443`.

If you did not select **Use HTTPS protocol**, replace `https` with `http` in the address bar of your browser.

See Also

More About

- “Review Results in Polyspace Code Prover Access”

Database Backup

There are two recommended methods to create a backup of your Polyspace Access database. Both methods rely on PostgreSQL utilities. The first method creates a dump of your database, or database cluster, and enables you to recover the database from when the dump was created. The second method uses write ahead logs (WAL) generate by PostgreSQL. It allows incremental backups and recovery.

Based on your database size and frequency of use, establish a policy for how often you perform a backup and which backup method is the most adequate.

Database Backup

To back up a particular database, use the `pg_dump` utility. The utility generates a list of SQL commands that allow you to reconstruct your database. To back up all the databases in your database cluster, use `pg_dumpall`. The backup operations require superuser privileges. The privileges are set through `postgre` and are separate from the user privileges on your system. To ensure that your backup does not contain partial or corrupted data, stop the **ETL** and **Web Server** services before starting the backup operation.

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

```
docker exec -it polyspace-usermanager kill 1
docker exec -it polyspace-db kill 1
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
docker exec -it polyspace-gateway kill 1
```

For example, to back up database `my_db` from the **Database** service running on machine `host1` on port 1234, use this command:

```
pg_dump -U postgres -h host1 -p 1234 my_db > db_backup
```

The `-U` specifies superuser `postgres`. The command outputs a text file `db_backup`.

You can then create a database `new_db` by using template `template0` and restore from `db_backup`.

```
createdb -U postgres -h host1 -p 1234 -T template0 new_db
psql -U postgres -h host1 -p 1234 new_db < db_backup
```

To back up all the databases of the **Database** service, use `pg_dumpall`. Use pipes to load your backup directly to a different server:

```
pg_dumpall -U postgres -h host1 -p 1234 | \
psql -U postgres -h host2 -p 5678 postgres
```

Before you execute this command, the **Database** service on `host2` at port 5678 must exist and the service must be running. For more examples on creating database dumps, see [SQL Dump](#).

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

Alternatively, you can rely on WAL files to perform incremental backups and recoveries of your database. The WAL records all changes made to the database. The system usually 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 Configuration” on page 1-5
- “Manage Polyspace Code Prover Access Software”

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 on page 1-39 your database before you perform a clean up operation.

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 **Web Server** and **ETL** services. To stop the services, go to the **Services** tab on the cluster operator (COP) web UI and click **Stop** next to the service you want to stop. You can also stop the services by opening a terminal on the server hosting these services and entering:

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

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

```
docker exec -it polyspace-usermanager kill 1
docker exec -it polyspace-db kill 1
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
docker exec -it polyspace-gateway kill 1
```

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

```
docker exec polyspace-db vacuumdb -U postgres --no-password --host hostName --port port prs_data
```

hostName is the host name of the machine hosting the **Database** service. *port* is the **Port number** value specified for the **Database** setting in the cluster operator.

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-db vacuumdb -U postgres --no-password --host hostName --port port --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-db vacuumdb -U postgres --no-password --host hostName --port port --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 **Web Server** and **ETL** services from the **Services** tab of the COP web UI, or in a terminal on the server hosting the services by entering:

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

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

Delete Outdated Projects

When users delete a project from the **PROJECT EXPLORER** of the Polyspace Access web interface, the project moves to the **ProjectsWaitingForDeletion** folder. The project, including all the runs that you uploaded to the project, remains stored in the database until you explicitly delete it.

The **ProjectsWaitingForDeletion** folder is visible only to Polyspace Access users that have the role of **Administrator**. However, users with this role cannot delete the projects *from the Polyspace Access interface*.

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

To remove old project runs or entire projects from your database, use a script that you save as a `.pscauto` file. You run the script by moving the `.pscauto` file to the **Storage directory** of the **ETL** service. Only a user with write privileges on the **Storage directory** can perform this operation.

- To delete project runs from a project but not the project itself, use the `clean_project` command. Specify the project path and one of these options.

- `clean_project projectPath DATE YYYY-MM-DD`

Results uploaded before the given date are deleted.

- `clean_project projectPath MAXRUNS NNN.`

NNN is an integer. If you upload more than NNN runs to a project, only the NNN most recent runs are not deleted. To delete all the project runs, use `MAXRUNS 0`.

- `clean_project projectPath AGE D`

D is the number of days. To remove recently uploaded results, use this option. All results runs that are older than D days are deleted.

- 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 `-list-project` flag with the `polyspace-access` binary. For more information, see `polyspace-access -h -list-project`. If the project path contains spaces or special characters, enclose the project path in double quotes.

For example, to perform a one-time cleanup of project `public/foo` and remove all results uploaded before a specific date, save this command to a `.pscauto` file, for instance `cleanup.pscauto`:

```
clean_project public/foo DATE 2018-09-01
```

Move the file to the **Storage directory** of the **ETL** service:

```
mv cleanup.pscauto /tmp/polyspace/storage
```

All analysis runs uploaded to project `foo` prior to September 1, 2018 are deleted from the database.

You can also perform a cleanup on a specific project every time you upload a run to that project. To keep only the 20 most recent runs every time you upload a result to `myProject/example (Bug Finder)`, save these commands to your `.pscauto` file:

```
assign_to_project "myProject/example (Bug Finder)"  
clean_project $currentProjectName MAXRUNS 20
```

To delete `myProject/example (Bug Finder)` entirely from the database, save this command to the `.pscauto` file.

```
delete_project "myProject/example (Bug Finder)"
```

Note 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 Configuration” on page 1-5
- “Manage Polyspace Code Prover Access Software”

Update or Uninstall Polyspace Access

If you want to remove or update Polyspace Access, follow these steps to uninstall Polyspace Access. Before you apply any updates, uninstall your current version of Polyspace Access. Before you uninstall the software, back up the Polyspace Access database. See “Database Backup” on page 1-39.

- 1 Inform Polyspace Access users of the upcoming update or uninstallation.
- 2 Verify that the COP agent is running with the command

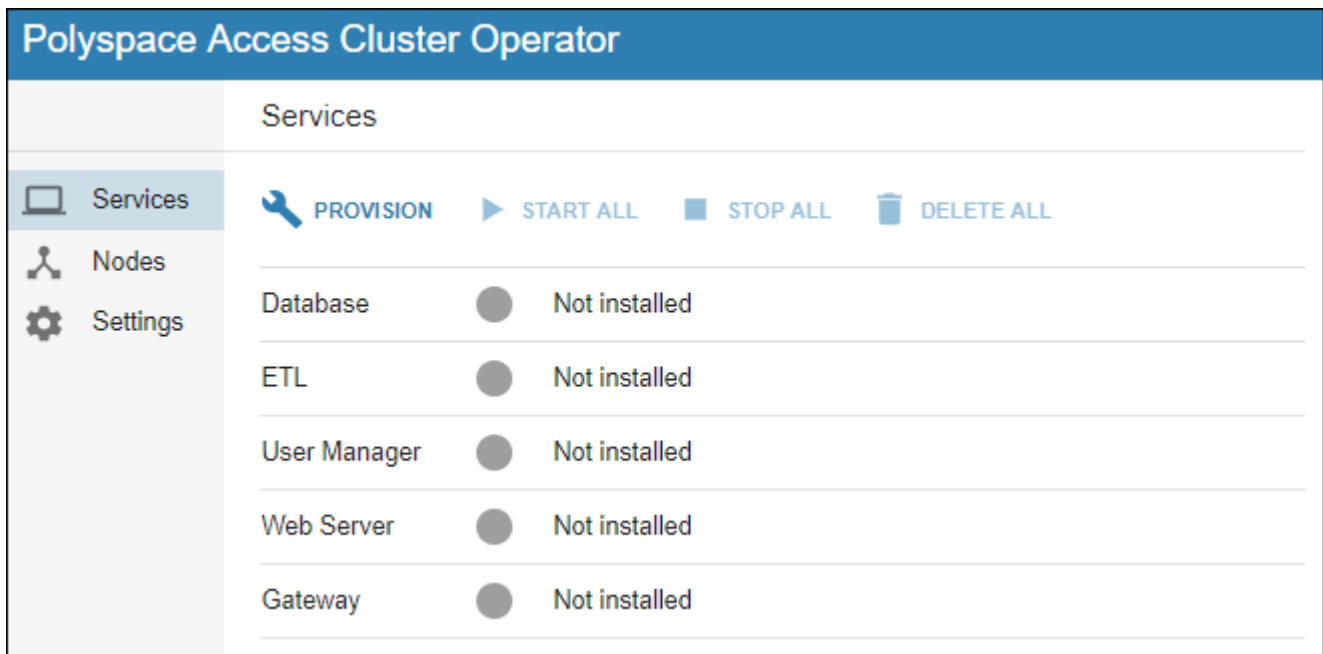
```
docker stats --no-stream
```

If `polyspace-cop` is not listed under the `NAME` column in the command output, start the COP binary `cop-docker-agent`.

- 3 Open the COP web interface, go to the **Services** tab, and click **DELETE ALL**. After you delete a service, the service indicator turns gray and you see the text **Not installed** next to the indicator.

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

```
docker exec -it polyspace-usermanager kill 1
docker exec -it polyspace-db kill 1
docker exec -it polyspace-etl kill 1
docker exec -it polyspace-web-server kill 1
docker exec -it polyspace-gateway kill 1
```



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

To delete a data volume and its content, click **Nodes** in the left pane, click the appropriate node ID, select the volume, and then click **DELETE**. You can also manually delete the folder where you store the database.

- 4 Stop the COP agent from the command line window by pressing **CTRL+C**.
- 5 Go to the folder where you unzipped the installation image for Polyspace Access and delete the `cop-docker-agent` binaries and TAR files. If you are updating Polyspace Access, do not delete these files until you complete the update.

To reuse your current Polyspace Access services configuration after you update, make a backup copy of the `settings.json` file.

To complete an update, download and unzip the new installation image, then go to the folder where you unzipped the new image and “Configure and Start the Cluster Operator” on page 1-12.

- If you reuse an existing configuration for the Polyspace Access services, copy your backup `settings.json` file into the same folder as the COP binary before you start the COP.
- To apply new configuration settings for the Polyspace Access services, see “Configure Polyspace Access Services” on page 1-16.

See Also

`cop-docker-agent`

More About

- “Configure and Start the Cluster Operator” on page 1-12
- “Configure Polyspace Access Services” on page 1-16

Manage Polyspace Access License

Configure Polyspace Access License

If this is your first time configuring the Polyspace Access license, follow these steps. Otherwise, see “Manage Named Users for Polyspace Access” on page 2-6 to add or remove users. Make sure that you install the license manager on page 2-4 before you continue.

Note These instructions do not apply to Enterprise license customers. Contact your license administrator to configure the Polyspace Access Enterprise license.

- 1 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_CP_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_CP_Access USER admin
# Grant right-to-use privileges to group of users
INCLUDE Polyspace_CP_Access GROUP ACCESS_CP_users
```

You use this file to identify the users to whom you grant right-to-use privileges for Polyspace Bug Finder Access (`Polyspace_BF_Access`) and Polyspace Code Prover Access (`Polyspace_CP_Access`). For each user, enter the user name that the user specifies to log into Polyspace Access. The user names correspond to the user name entries in your company LDAP or the LDIF file if you use embedded LDAP. See “Configure Lightweight Directory Access Protocol” on page 1-20.

- 2 Copy your Polyspace Access license to the machine where you installed the license manager and save it as `license.dat`. Then, 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
```

- *lmHostname* is the 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:

Windows	net config workstation findstr /C:"Full Computer name"
Linux	hostname -A

- *HostID* is the 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. *HostID* must also match a MAC address on the machine where you run the license manager.
- *pathTo_MLM_bin* is the path to the MLM binary. You can find this binary in *LM_Folder\etc\win64* (Windows) or *LM_Folder/etc/glnx64* (Linux), where *LM_Folder* is the folder where you installed the license manager.
- *pathTo_MLM.opt* is the path to the options file that you created in step 1.
- By default, the license manager starts on port 27000. To use a different port, specify a different port number at the end of the **SERVER** line.

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 may 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.

- 3 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` on the machine where you installed Polyspace Access. This machine can be a different machine from the one where you installed the license manager. Specify the path to this file for the **License file:** field of the **Web Server** settings in the Cluster Operator web interface.

Make sure that the docker engine can resolve the host name *lmHostname*. In a command-prompt window, enter:

```
docker run --rm -it alpine ping lmHostname
```

If the docker engine cannot resolve this host name, in `network.lic`, replace *lmHostname* with the IP address of the machine where you installed the license manager.

- 4 In a command-prompt window, navigate to the folder where you installed the license manager, and then 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 <code>lmtool.exe</code> 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.

- 5 After you start the license manager, ensure that the license manager is configured to automatically start at boot time.

Windows	Use <code>lmtool.exe</code> and go to the Config Services tab, then check that Start Server at Power Up and Use Services are selected.
Linux	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. Configure the license manager to start at the end of the boot sequence.

Each licensed Polyspace Access user can log in to up to five concurrent sessions. These actions define a session:

- Log into the Polyspace Access web interface.
- View results stored on Polyspace Access from the desktop interface.
- Upload results to Polyspace Access from the desktop UI or command-line interface.
- Perform an operation from the command-line interface that requires a Polyspace Access login.
- Generate a report for results stored on Polyspace Access.

To review or generate reports for results that were generated with Polyspace Code Prover or Polyspace Ada products and that are stored on Polyspace Access, you need a Polyspace Code Prover Access license.

Install License Manager

Install License Manager Manually

You can install the License Manager by downloading the binaries directly from License Manager Download. The download 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.

Install License Manager by Using the MATLAB Installer

You can use the MATLAB installer to download the FlexNet Publisher license manager only if you have MathWorks products other than Polyspace Access with licenses that require a license manager.

When you run the MATLAB installer to install the other MathWorks products, the license manager is available in the list of available products.

For additional configuration instructions, see “Set Up Named User Licensing” (Installation and Licensing).

Manage Named Users for Polyspace Access

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_CP_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_CP_Access USER admin
# Grant right-to-use privileges to group of users
INCLUDE Polyspace_CP_Access GROUP ACCESS_CP_users
```

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 lmtool.exe 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 lmtool.exe 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.

Get Started with Polyspace Code Prover Access

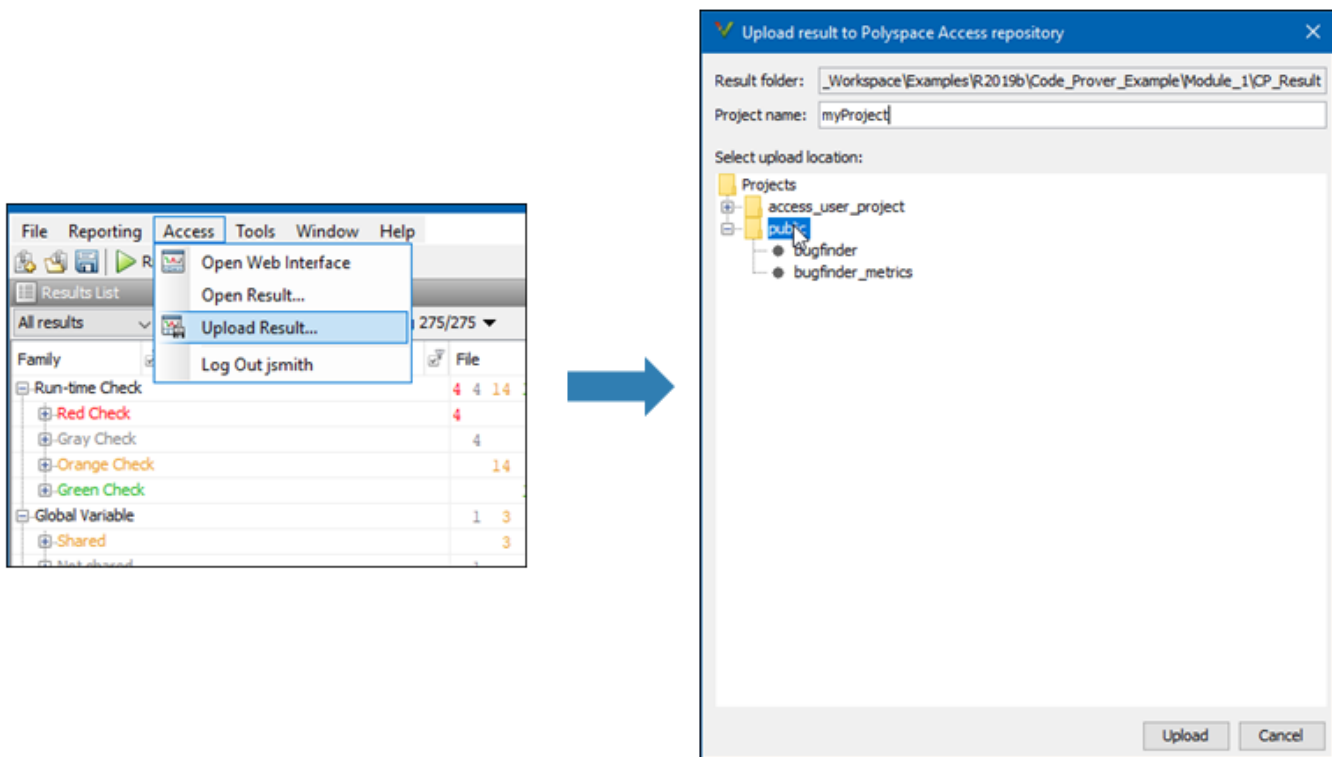
Upload Results to Polyspace Access

Polyspace Access offers a centralized database where you can store Polyspace analysis results for sharing and collaborative reviews. After you upload results, open the Polyspace Access web interface to view statistics about the quality of your code and to triage and review individual results.

Upload Results from Polyspace Desktop Client

Before you upload results, you must configure the Polyspace desktop client to communicate with Polyspace Access. See “Register Polyspace Desktop User Interface” on page 1-33.

To upload analysis results to the Polyspace Access database from the Polyspace desktop client, select a set of results in the **Project Browser** pane or open the results in the **Results List** pane. Go to **Access > Upload Results** and follow the prompts. If you get a login request, use your Polyspace Access login credentials.



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

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.

Upload Results at Command Line

You can upload results from the command line only if they are generated with Polyspace Bug Finder Server or Polyspace Code Prover Server.

To upload analysis results to Polyspace Access from the DOS or UNIX command line, use the `polyspace-access` binary. In the command, specify the path of the folder under which the `.psbf`, `.pscp`, or `.rte` results file is stored. For instance, to upload Polyspace Bug Finder results stored in the file `BF_results\ps_results.psbf`, use this command:

```
polyspace-access -host hostName -port port -upload BF_results
```

The command prompts you for your Polyspace Access login credentials, then uploads the results to the public folder of the Polyspace Access database. *hostName* is the fully qualified host name of the machine where you run the **Gateway** service. *port* is the **Port number** value specified for the **Gateway** service in the settings of the cluster operator. Depending on your configuration, you might also have to specify the `-protocol` option in the command. See “Configure Polyspace Access Services” on page 1-16.

For additional information on `polyspace-access`, see the documentation for Polyspace Code Prover or Polyspace Code Prover Server .

See Also

More About

- “Register Polyspace Desktop User Interface” on page 1-33
- “Interpret Results”
- “Manage Results”

Open or Export Results from Polyspace Access

Polyspace Access offers a centralized database where you can store Polyspace analysis results for sharing with your team and collaborative reviews. After you upload analysis results to the database, you can view the results in your web browser or you can open the results from any Polyspace desktop interface that is configured for Polyspace Access. You can also export a list of results to a tab-separated value (TSV) file for further processing, such as applying custom filters and pass/fail criteria.

Open Polyspace Access Results in a Desktop Interface

Before you open Polyspace Access results in a desktop interface, you must configure the Polyspace desktop interface to communicate with Polyspace Access. See “Register Polyspace Desktop User Interface” on page 1-33.

To open results stored in the Polyspace Access database, go to **Access > Open Result** in the desktop interface, and follow the prompts. If you get a login request, use your Polyspace Access login credentials.

You can also open the desktop interface from the Polyspace Access web interface. On the toolstrip, click **Open in Desktop**. The desktop interface opens and shows the analysis results currently displayed in the Polyspace Access web interface.

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

Once you open results in the Polyspace desktop interface, changes you make to the **Status**, **Severity**, or comments fields are reflected in Polyspace Access after you save those changes.

After you upload analysis results to Polyspace Access, if you open a local copy of these results in the desktop interface, you cannot edit the **Status**, **Severity**, or comments fields.

Export Polyspace Access Results to a TSV File

You can export Polyspace Access results to a tab-separated values (TSV) file only from the command line by using the `polyspace-access` binary. When you export results, you generate a TSV file that lists results with most of the same results attributes as the “Results List”. Each listed result also includes a URL through which you can open the result in Polyspace Access. To filter the list of results you export, see the `polyspace-access` export options.

For example, to export all coding rules with status Unreviewed from project **myProject** stored in the **public** folder on Polyspace Access, open a command prompt terminal and enter:

```
polyspace-access -host hostName -port port ^  
-export public/myProject -coding-rules -review-status Unreviewed ^  
-output coding_rules.tsv
```

The command prompts you for your Polyspace Access login credentials, and then outputs file `coding_rules.tsv`.

hostName is the fully qualified host name of the machine where you run the **Gateway** service. *port* is the **Port number** value specified for the **Gateway** service in the settings of the cluster operator.

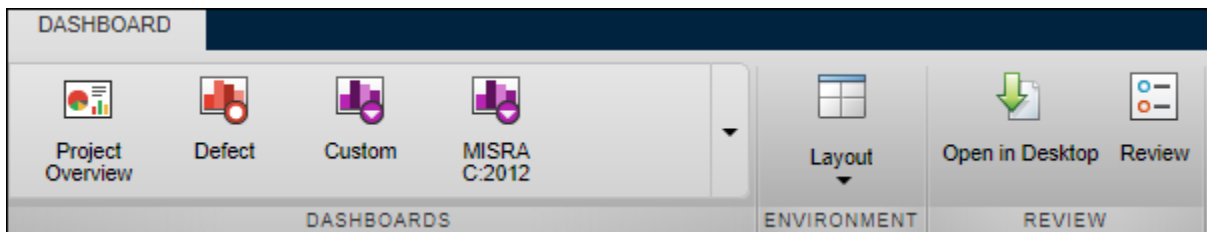
Depending on your configuration, you might also have to specify the `-protocol` option in the command. See “Configure Polyspace Access Services” on page 1-16.

For additional information on `polyspace-access`, see the documentation for Polyspace Code Prover or Polyspace Code Prover Server.

Dashboard

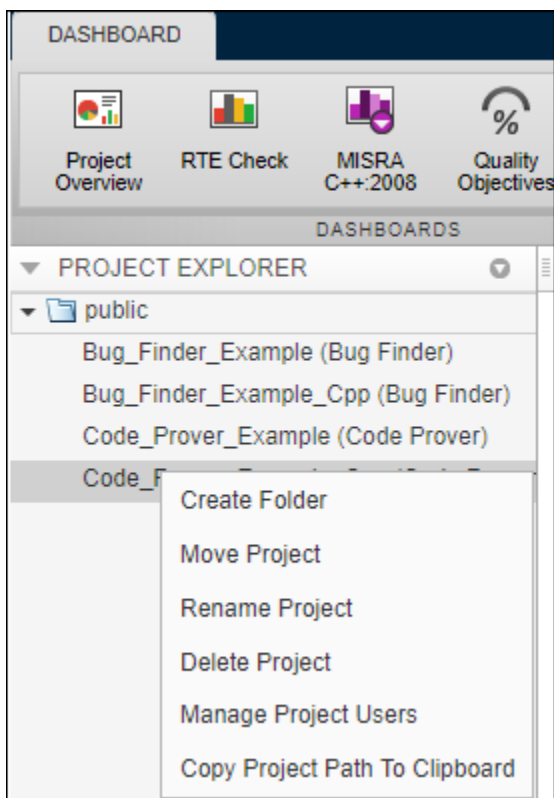
The **DASHBOARD** perspective provides an overview of the analysis results in graphical format, with clickable fields that enable you drill down into your findings by file, project, or category.

DASHBOARD toolstrip



- Click a button in the **DASHBOARDS** section of the toolstrip to open the corresponding dashboard for the selected folder or project. Except for **Project Overview** and **Quality Objectives**, each dashboard shows information for a single family of findings.
- The **Open in Desktop** and **Review** buttons in the toolstrip are not available when you select a folder in the **PROJECT EXPLORER** pane.

PROJECT EXPLORER pane



- View all projects and folders for which you are an **Administrator**, **Owner** or **Contributor**. All users are contributors to the **Public** folder

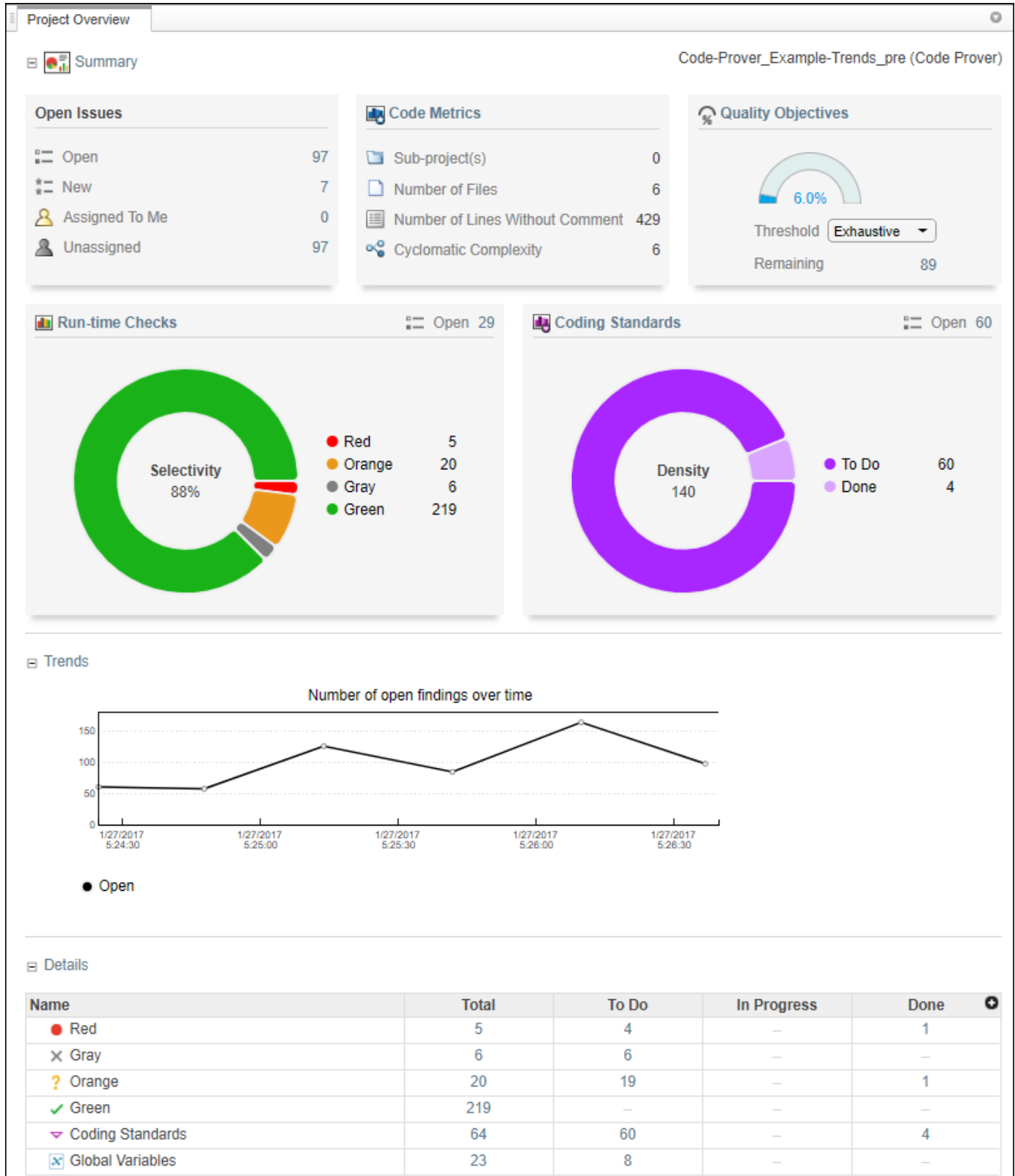
- To manage folders, projects, or user permissions, use the context menu.
- The dashboards on the right display information for the selected folder or project.

PROJECT DETAILS pane

PROJECT DETAILS	
Project	
Name	Bug_Finder_Example (Bug Finder)
Language	C
Tools	Bug Finder
Coding Rules	Custom Coding Rules, MISRA C:2012
Number of Runs	1
Current Run (ID 1)	
Date	11/21/18, 7:04 PM
Job	1.0

- View additional details about the selected folder or project. You can view information about which language and coding standards were enabled in the analysis configuration.

Project Overview dashboard



This dashboard gives you a snapshot of all the findings available for the selected folder or project. If you select a folder that includes multiple projects, the dashboard displays an aggregate of results for all the projects. The dashboard contains three collapsible sections:

- **Summary**

Displays cards with information about open issues, code metrics, quality objectives (when available), and the different families of findings. Click the card title to open its corresponding dashboard. Click a section of a pie chart or the pie chart legend (when applicable) to open a list filtered to this set of findings.

The **Run-time Check** card shows a distribution of findings as red, orange, gray, and green. The card also shows the selectivity, the number of green checks as a percentage of all detected run-time checks.

Defects and **Coding Rules** cards show a distribution of findings as **To Do**, **In Progress**, and **Done**.

The card also shows the density, the number of defects or coding standard violation per one thousand lines of code without comments. To view the density you must enable **Code Metrics** in your analysis configuration.

- **Trends**

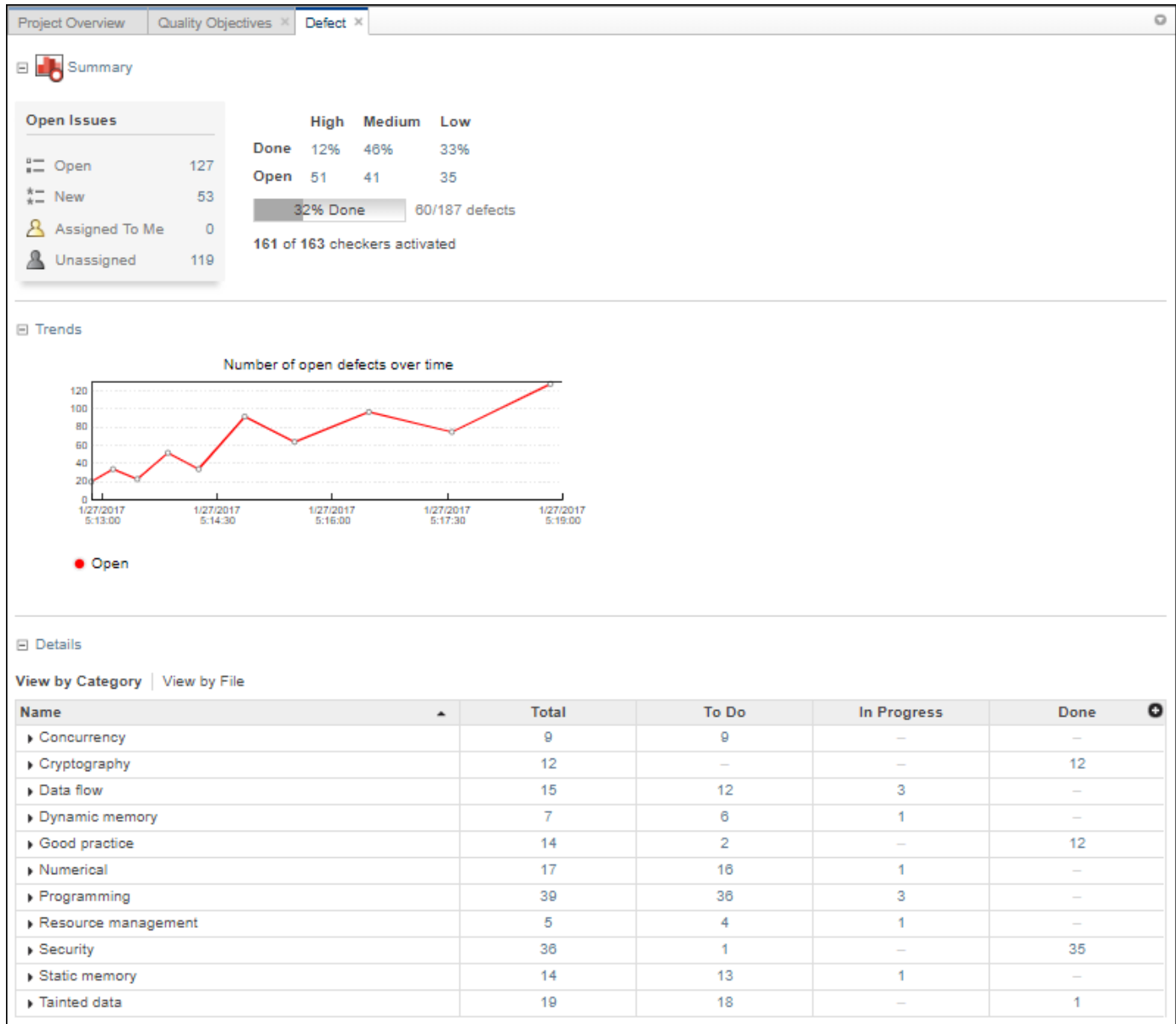
Displays a trend chart of the number of open findings over time as you upload additional runs for a project.

- **Details**

Displays a table with a row containing the number of **Total**, **To Do**, **In Progress**, and **Done** for each type of analysis finding. Click the number of findings in a row (when applicable) to open a list filtered to this set of findings.

Green run-time checks, green shared variables, not shared variable, and code metrics do not count toward the number of **To Do**, **In Progress**, and **Done** findings.

RTE Check, Defects and Coding Rules dashboards



These dashboards give you a more in-depth overview for a family of findings. If you select a folder that includes multiple projects, the dashboards display an aggregate of results for all the projects. The dashboards contain three collapsible sections:

- **Summary**

Displays a table with information about open issues and the progress in addressing these issues. Click the number of findings in the card (when applicable) to open a list filtered to this set of findings.

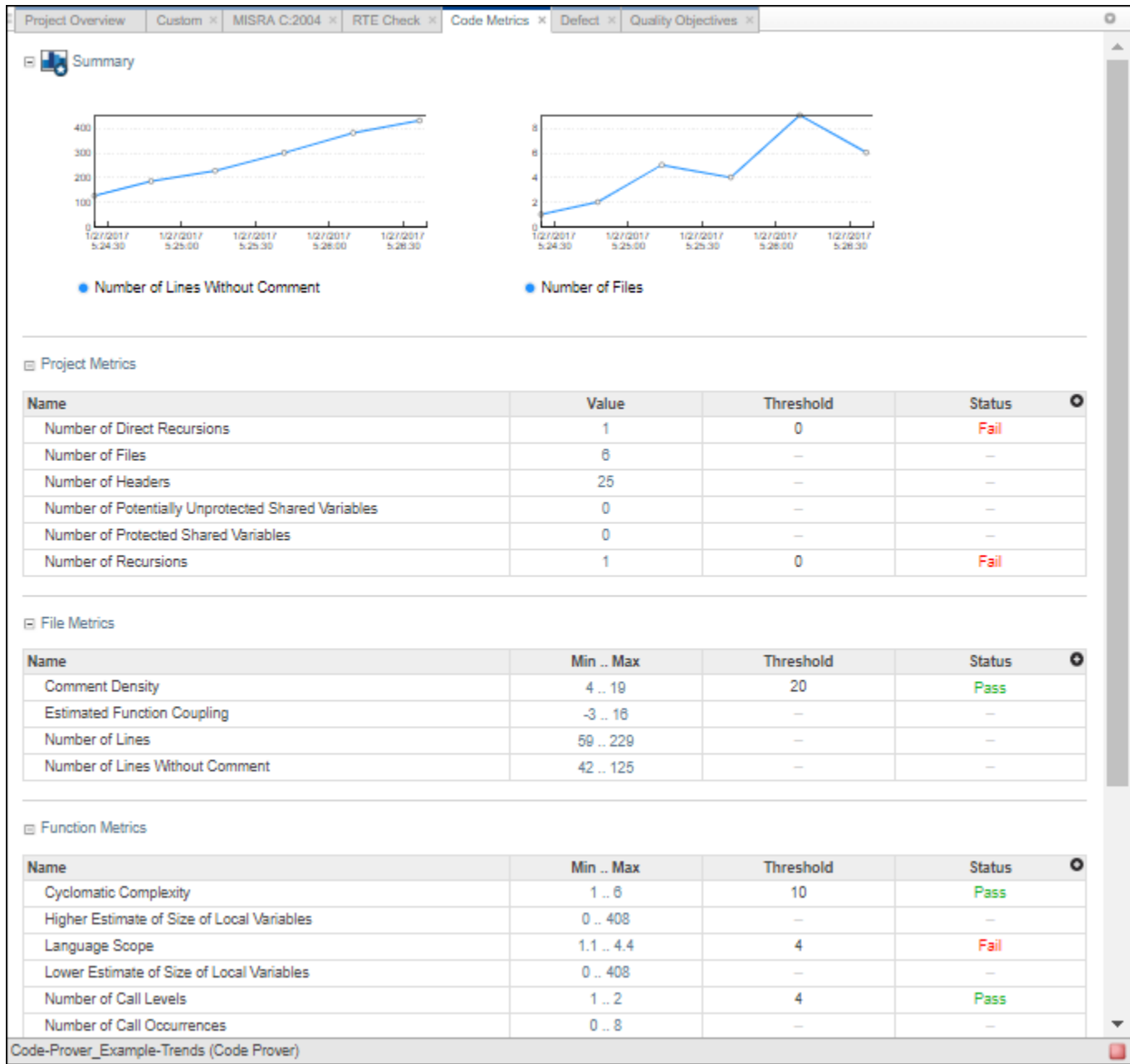
- **Trends**

Displays a trend chart of the number of open findings over time as you upload additional runs for a project.

- **Details**

Displays a table that allows you to drill down into the findings by category or by file. If you select a folder that contains multiple projects, you get a categorization by project instead of by file. Click the number of findings in a row (when applicable) to open a list filtered to this set of findings.

Code Metrics dashboard



This dashboard contains four collapsible sections.

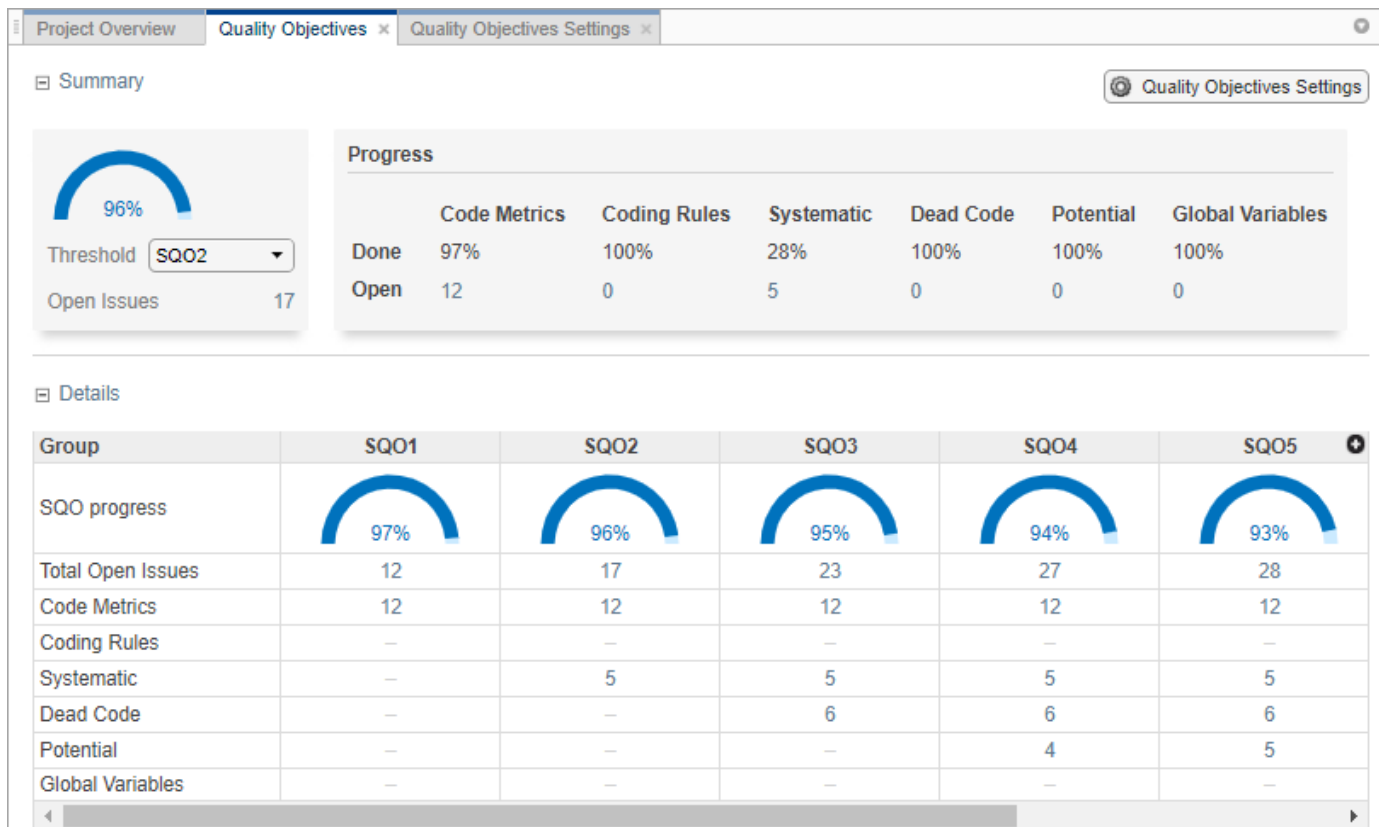
- **Summary**

Displays trends charts of the number of lines without comments and the number of files for the selected folder or project

- **Project Metrics, File Metrics, and Function Metrics**

These sections display tables with rows containing the value or range of a metric, along with its threshold and pass/fail status when applicable. Click the number of findings in a row (when applicable) to open a list filtered to this set of findings.

Quality Objectives dashboard



This dashboard displays a summary of the quality of your code against the threshold selected from the dropdown menu. The dashboard also shows a table with details of code quality for all quality objective thresholds.

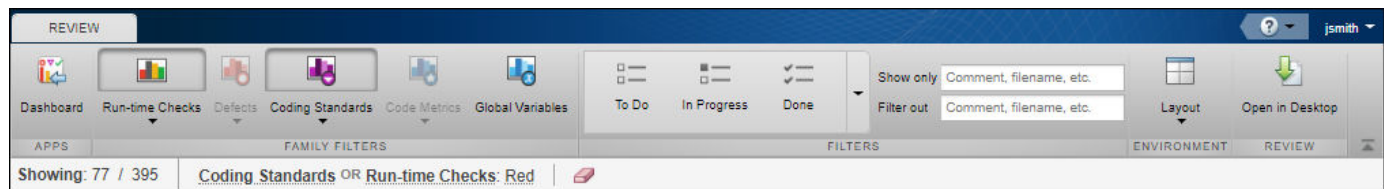
Review

The **REVIEW** perspective provides you with an environment that enables you to:


- Filter and investigate individual findings in your code.
- Add a review status, severity or comment to findings.
- Assign an owner to a finding and create a ticket in your bug tracking tool to track the issue.

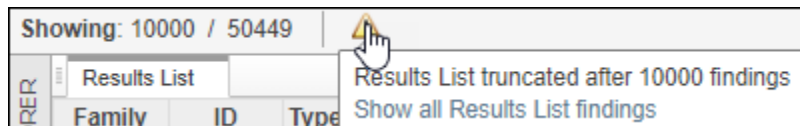
Note The **REVIEW** perspective is only available for analysis results generated with a Polyspace product version R2019a or later.

REVIEW toolstrip



- Click a button in the **FAMILY FILTERS** section of the toolstrip to see the corresponding family of findings, or a subset of those findings. The filters bar underneath shows how many findings are displayed out of the total findings, along with which filters are currently applied.

If the **Results List** exceeds 10000 findings, Polyspace Access truncates the list and displays this icon  in the filters bar. To show all findings, see the contextual help of the icon.



The 10000 findings limit is preset and cannot be changed.

- The buttons in the **FILTERS** section of the toolstrip are global. They apply to all families of findings.

When you select the **To Do**, **In Progress**, or **Done** filters, the filtered **Results List** does not show green run-time checks, green shared variables, not shared variables, or code metrics findings.

Default view: Results List, Results Details, and Source Code

The screenshot displays the Polyspace Code Prover interface with the following components:

- Top Bar:** Includes navigation icons for Dashboard, Run-time Checks, Defects, Coding Standards, Code Metrics, and Global Variables. It also features status indicators for To Do, In Progress, and Done, along with filters and layout options.
- Results List (Table):** A table with columns for Family, ID, Type, Group, and Check. The first few rows show Red Checks (e.g., 58538, 58603, 58686) and Orange Checks (e.g., 58534, 58627, 58681).
- Results Details (Panel):** Shows the selected finding (ID 58538) with a status of 'Unreviewed' and severity of 'Unset'. It includes a comment field and a 'Track issue' button. A detailed error message is displayed: "Illegally dereferenced pointer" with an explanation of the pointer error and a stack trace table.
- Source Code (Panel):** Displays the C code snippet corresponding to the finding, with a red asterisk marking the dereference operation on line 101.

Event	File	Scope
1	Entering function 'RTE'	main.c main()
2	Entering function 'Point...	example.c RTE()
3	Illegally dereference...	example.c Pointer_Arithmetic()

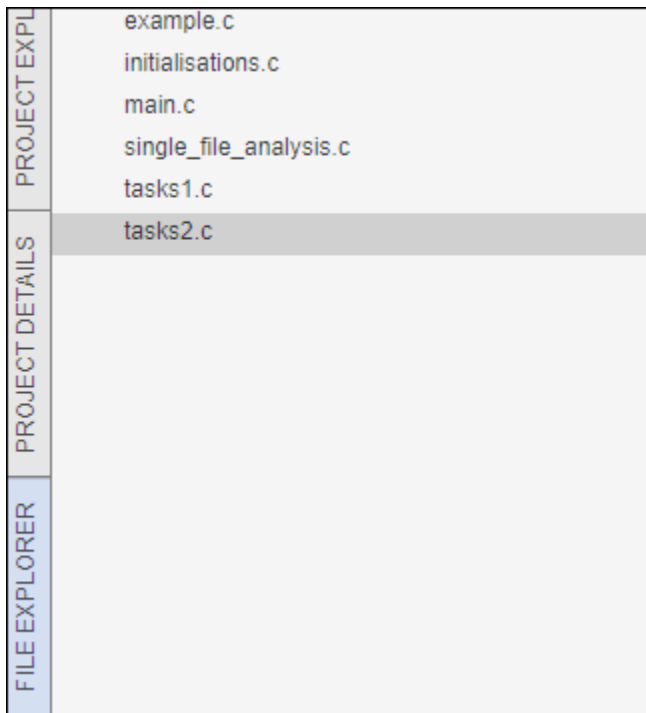
```

94   for (i = 0; i <= 100; i++) {
95       *p = 0;
96       p++;
97   }
98
99   if (get_bus_status() > 0) {
100       if (get_oil_pressure() > 0) {
101           *p = 5; /* Out of bounds */
102       } else {
103           i++;
104       }
105   }
106
107   i = get_bus_status();
108
109   if (i >= 0) {*(p - i) = 10;}
  
```

In the default layout, you see the **Results List**, **Results Details**, and **Source Code** panes.

- Click a finding in the **Results List** to see its location in the **Source Code** pane. Additional information about the finding is available in the **Results Details** pane. To open contextual help for the finding, in the **Results Details** click . When available, click the to see fix suggestions for the defect.
- Click a column heading in the **Results List** to sort findings according to that heading.
- Right-click a cell in the **Results List** to show only/exclude findings based on the content of that cell.

To open additional panes, use **Layout > Show/Hide View**.

FILE EXPLORER pane

Use the file explorer to show findings by file in the **Results List** pane.

Manage Permissions and View Project Trends

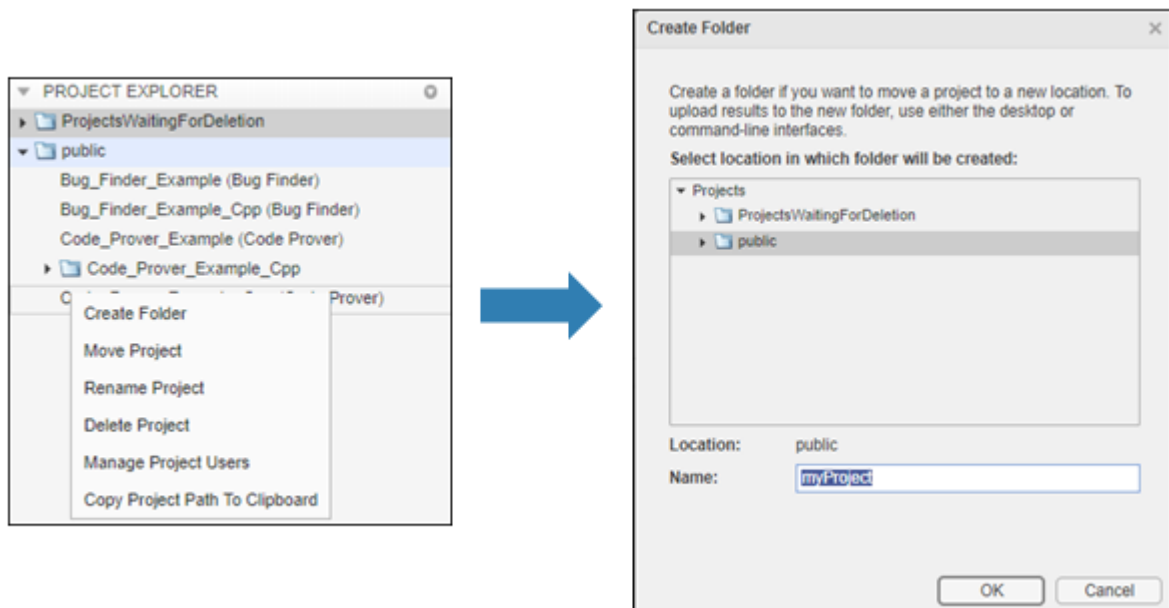
Before you start reviewing the overall quality of a project and investigating findings in your code, create project folders and set permissions to allow or restrict team members access to your projects.

Create a Project Folder

To facilitate the review process, create folders in Polyspace Access to group related results.

Create Folder from the Polyspace Access Interface

From the **PROJECT EXPLORER** in the **DASHBOARD** perspective, select any existing folder or project and click **Create Folder** in the context menu. In the **Create Folder** window, click an existing folder to create a subfolder. To create a folder at the top of the **PROJECT EXPLORER** hierarchy, click **Projects**.



Create Project Folder at Command Line

To create a folder in Polyspace Access from the DOS or UNIX command lines, use the `polyspace-access` binary. This binary is available under the `polyspaceroot/polyspace/bin` folder with a Polyspace Code Prover or a Polyspace Code Prover Server installation. `polyspaceroot` is the Polyspace product installation folder, for example `C:\Program Files\Polyspace Server\2019a`.

For instance, to create `myProject` under the node `myRelease`, use this command:

```
polyspace-access -host hostName -create-project myRelease/myProject
```

`hostName` is the host name of the machine where the Polyspace Access **Gateway** service is running. You specify this host name in the URL of the Polyspace Access web interface. Depending on your configuration, you might also need to specify the `-port` and `-protocol` options in the migration command.

For more information on `polyspace-access`, see the Polyspace Bug Finder Server or Polyspace Code Prover Server documentation.

Manage Project Permissions

To set permissions for folders or projects in Polyspace Access, assign user roles. These are the permissions that correspond to each role.

Role	Permission
Administrator	Move, rename, or delete specified folders or projects and review their content. Assign roles Administrator , Owner , Contributor , or Forbidden . View and manage contents of ProjectsWaitingForDeletion folder. See “Delete Outdated Projects” on page 1-42. You cannot move a folder or project to a new location if a folder or project with the same name already exists at that location.
Owner	Move, rename, or delete specified folders or projects and review their content. Assign roles Contributor or Forbidden . You cannot move a folder or project to a new location if a folder or project with the same name already exists at that location.
Contributor	Review content of specified folder or project. See the roles of other users in the project.
Forbidden	No access to the specified folder or project. Set this role to restrict the access to a project inside a folder that is accessible to the user.

Only **Administrator** or **Owner** roles can allow or restrict the access of other team members to a project or folder. You are the owner of folders that you create and of project results that you upload.

Only **Administrator** roles can assign other users as administrators or as owners to a project or folder. To set a user as **Administrator**, see “Manage Permissions in Polyspace Access Web Interface” on page 3-17 or “Configure the Web Server and Gateway” on page 1-30.

The permissions that you set on a folder apply to all projects in that folder. For instance, if user `jdoe` has **Contributor** privileges for folder `myRelease`, `jdoe` is a contributor for all projects under `myRelease`. You can set additional permissions for each project under `myRelease`. The **Administrator** roles applies to all projects.

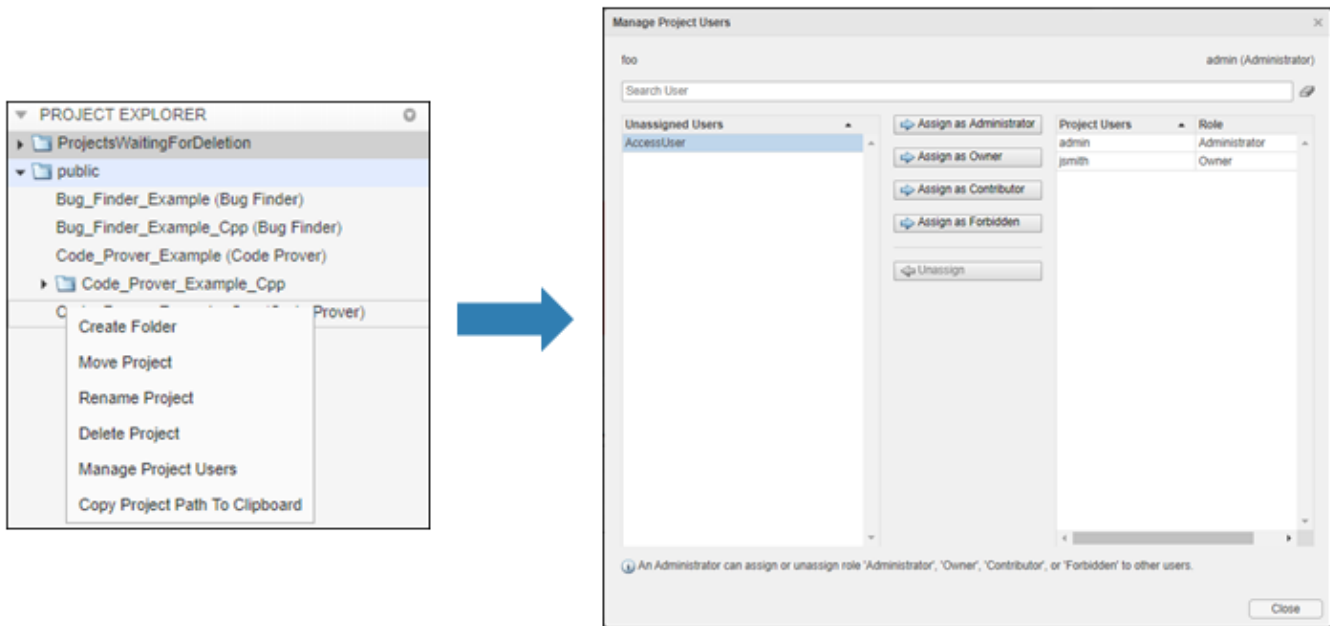
Project results uploaded to the **public** folder are accessible to all team members.

Manage Permissions in Polyspace Access Web Interface

From the **PROJECT EXPLORER** in the **DASHBOARD** perspective, select any existing folder or project and click **Manage Project Users** in the context menu. You can search for a user, assign a role to a user with no role, or change the role of a user.

- **Administrator** role can assign other users as **Administrator**, **Owner**, **Contributor**, or **Forbidden**.
- **Owner** role can assign other users as **Contributor** or **Forbidden**.

- **Contributor** and **Forbidden** roles cannot assign roles to other users.



The **Assign as Administrator** button is visible only for users with role **Administrator**. You must assign at least one user as administrator in the cluster operator settings on page 1-30 before you can manage administrators from the web interface.

Manage Permissions at Command Line

To manage access to uploaded results from the DOS or UNIX command lines, use the `polyspace-access` binary. This binary is available under the `polyspaceroot/polyspace/bin` folder with a Polyspace Code Prover or a Polyspace Code Prover Server installation. `polyspaceroot` is the Polyspace product installation folder, for example `C:\Program Files\Polyspace Server\2019a`.

For instance to assign `jsmith` as **Contributor** for project `myProject`, use this command:

```
polyspace-access -host hostName ^
-set-role contributor -user jsmith ^
-project-path myProject
```

`hostName` is the host name of the machine where the Polyspace Access **Gateway** service is running. You specify this host name in the URL of the Polyspace Access web interface. Depending on your configuration, you might also need to specify the `-port` and `-protocol` options in the migration command.

You cannot assign the **Administrator** role to a user from the command line.

For more information on `polyspace-access`, see the Polyspace Bug Finder Server or Polyspace Code Prover Server documentation.

View Project Trends

Project Overview
Code-Prover_Example-Trends_pre (Code Prover)


Open Issues

- ☰ Open 97
- ✚ New 7
- 👤 Assigned To Me 0
- 👤 Unassigned 97

Code Metrics

- 📁 Sub-project(s) 0
- 📄 Number of Files 6
- 📄 Number of Lines Without Comment 429
- 🔗 Cyclomatic Complexity 6

Quality Objectives

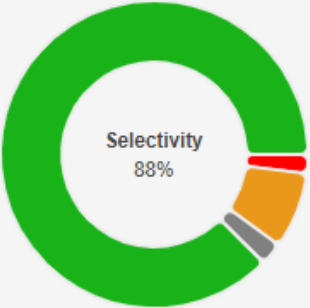


6.0%

Threshold Exhaustive

Remaining 89

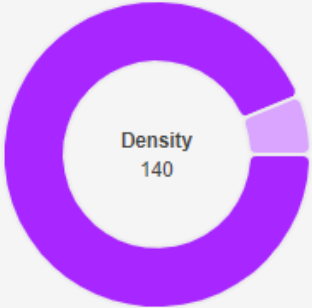
Run-time Checks ☰ Open 29



Selectivity
88%

● Red	5
● Orange	20
● Gray	6
● Green	219

Coding Standards ☰ Open 60

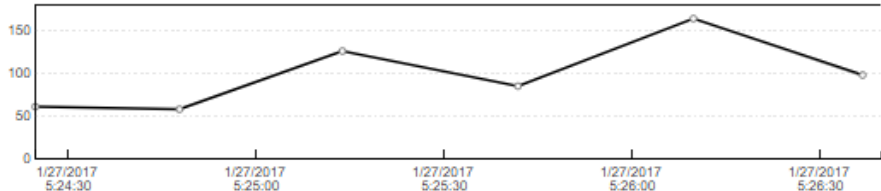


Density
140

● To Do	60
● Done	4

Trends

Number of open findings over time



● Open

Details

Name	Total	To Do	In Progress	Done
● Red	5	4	–	1
✕ Gray	6	6	–	–
? Orange	20	19	–	1
✓ Green	219	–	–	–
▼ Coding Standards	64	60	–	4
⊠ Global Variables	23	8	–	–

In the **DASHBOARD** perspective, select the project that you want to investigate from the **PROJECT BROWSER**.

If you select a folder, the project overview displays an aggregate of all the project results in that folder.

In the **Project Overview** dashboard, you see a summary of **Open Issues**, including the number of **New** results since the previous analysis run and the number of results that are **Unassigned**.

Other cards provide statistics for each family of findings. The **Run-time Checks** card shows the **Selectivity**, that is, the percentage of all findings that are green. When you enable the calculation of code metrics in your analysis, the **Defects** and **Coding Standards** cards show the **Density**, the number of findings per thousand lines of code without comments.

In the **Details** section, you see the review progress for each family of results. The results are classified as:

- **To Do**, with a status of Unreviewed.
- **In Progress**, with a status of To fix, To investigate, or Other.
- **Done**, with a status of Justified, No action planned, or Not a defect.

Green run-time checks, green shared variables, non-shared variables, and code metrics do not count toward the number of **To Do**, **In Progress**, and **Done** findings.

If the number of open issues increases, open additional dashboards by using the buttons in the **DASHBOARDS** section of the toolstrip. Each button opens a dashboard for a family of findings, for instance **Defects**. To determine the root cause of the increase, Use the information on these dashboards. Once you determine the set of findings that you want your team to focus on, open the **REVIEW** perspective to start managing the results. See “Manage Results” on page 3-21.

See Also

More About

- “Upload Results to Polyspace Access” on page 3-2

Manage Results

After you identify the results that you want to review, use the **REVIEW** perspective to manage these results. See “Manage Permissions and View Project Trends” on page 3-16.

If you open the **REVIEW** perspective from the **DASHBOARD** perspective, you see the **Results List** filtered down to the set of results you selected in a dashboard. If you open the **REVIEW** perspective by clicking a finding URL, you see only that finding in the **Results List**.

The screenshot displays the Manage Results interface with the following components:

- Toolstrip:** Includes navigation icons for Dashboard, Run-time Checks, Defects, Coding Standards, Code Metrics, and Global Variables. It also features filters for 'To Do', 'In Progress', and 'Done', and options for 'Show only' and 'Filter out' (both set to 'Comment, filename, etc.').
- Results List Table:**

Family	ID	Type	Group	Check
● *	58538	Red Check	Static memory	Illegally deref
● *	58603	Red Check	Other	Invalid use of
● *	58686	Red Check	Control flow	Non-terminat
● *	58701	Red Check	Static memory	Out of bound
● *	58845	Red Check	Control flow	Non-terminat
× *	58534	Gray Check	Data flow	Unreachable
× *	58627	Gray Check	Data flow	Unreachable
× *	58681	Gray Check	Data flow	Unreachable
× *	58725	Gray Check	Data flow	Unreachable
× *	58767	Gray Check	Data flow	Unreachable
× *	58847	Gray Check	Data flow	Unreachable
? *	58543	Orange Check	Static memory	Illegally deref
? *	58570	Orange Check	Numerical	Division by ze
? *	58582	Orange Check	Numerical	Overflow
? *	58585	Orange Check	Numerical	Overflow
? *	58589	Orange Check	Numerical	Overflow
? *	58597	Orange Check	Numerical	Overflow
? *	58599	Orange Check	Data flow	Non-initialize
? *	58601	Orange Check	Other	User assertio
? *	58626	Orange Check	Data flow	Non-initialize
? *	58674	Orange Check	Data flow	Non-initialize
? *	58675	Orange Check	Data flow	Non-initialize
? *	58676	Orange Check	Static memory	Illegally deref
? *	58707	Orange Check	Data flow	Non-initialize
? *	58712	Orange Check	Other	User assertio
? *	58766	Orange Check	Numerical	Overflow
? *	58773	Orange Check	Static memory	Out of bound
? *	58778	Orange Check	Data flow	Non-initialize
? *	58783	Orange Check	Other	User assertio
? *	58785	Orange Check	Data flow	Non-initialize
? *	58790	Orange Check	Other	User assertio
? *	58818	Orange Check	Numerical	Overflow
? *	58833	Orange Check	Numerical	Overflow
▼ *	58879	MISRA C:2012	9 Initialization	9.1 The value
▼ *	58880	MISRA C:2012	9 Initialization	9.1 The value
- Results Details Pane:** Shows details for the selected finding (ID 58538). It includes fields for Status (Unreviewed), Severity (Unset), and Assigned to. A comment field is available. The main content area displays the error message: "Illegally dereferenced pointer" with a detailed description: "Error: pointer is outside its bounds. Dereference of local pointer 'p' (pointer to int 32, size: 32 bits): Pointer is not null. Points to 4 bytes at offset 400 in buffer of 400 bytes, so is outside bounds. Pointer may point to variable or field of variable: 'array', local to function 'Pointer_Arithmetic'." Below this is an Event table:

Event	File	Scope
1	Entering function 'RTE'	main.c
2	Entering function 'Point...	example.c
3	Illegally dereference...	example.c
- Source Code Pane:** Displays the source code for the file 'single_file_analysis.c'. The code shows a loop where a pointer 'p' is incremented and then dereferenced, leading to the error. The relevant lines are:

```

94     for (i = 0; i < 100; i++) {
95         *p = 0;
96         p++;
97     }
98
99     if (get_bus_status() > 0) {
100         if (get_oil_pressure() > 0) {
101             *p = 5; /* Out of bounds */
102         } else {
103             i++;
104         }
105     }
106
107     i = get_bus_status();
108
109     if (i >= 0) {*(p - i) = 10;}

```

Apply additional filters to the **Results List** by using the toolstrip, or select a finding and use the context menu. To decide how to address each finding that you review, use the **Results Details** and **Source Code** panes. To open additional panes such as the **Call Hierarchy**, see **Layout > Show/Hide View** in the toolstrip. Once you decide how to address the finding, set or update the **Status**, **Severity**, **Assigned to**, or comment fields in the **Results Details** pane.

To create a bug tracking tool (BTT) ticket and keep track of the workflow that addresses a finding from an existing BTT project, click **Create** in the **Results Details** pane. Creating a BTT ticket is available only if Polyspace Access is configured to create BTT tickets. The ticket entry is populated with details of the finding and a URL to open the finding in Polyspace Access. See “Track Issue in Bug Tracking Tool”.

See Also

More About

- “Interpret Results”
- “Manage Results”

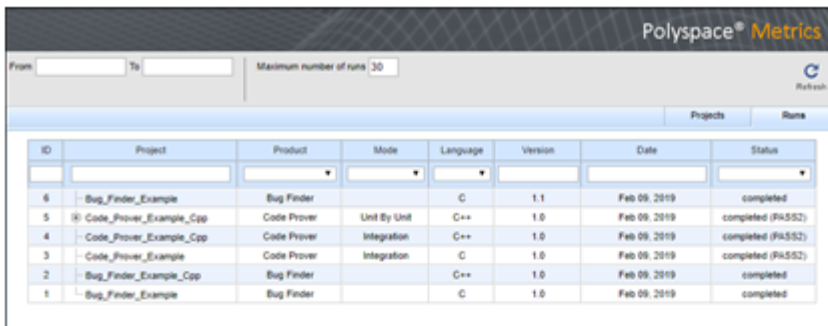
Migrate Results from Polyspace Metrics to Polyspace Access

If you use Polyspace Metrics to store results and monitor the quality of your source code, you can transfer those results to Polyspace Access.

The Polyspace Access **DASHBOARD** perspective offers a web interface with navigation between projects and categories of results. From the **Project Overview** dashboard, view aggregated statistics for all your projects or drill down to view results details by category or file. For each family of findings, open an additional dashboard to see details. After you narrow down the set of findings that you want to address, open them in the **REVIEW** perspective to start reviewing individual results.

Note The **REVIEW** perspective is only available for analysis results generated with a Polyspace product version R2019a or later. To review R2018b or earlier results that you migrated to Polyspace Access, see “Open Polyspace Access Results in a Desktop Interface” on page 3-4.


You can also review results from Polyspace Access by opening them in the Polyspace desktop interface. You do not need to download a local copy of Polyspace Access results to view those results in the desktop interface. The edits that you make to the results are saved directly in Polyspace Access and enable you to perform collaborative reviews.



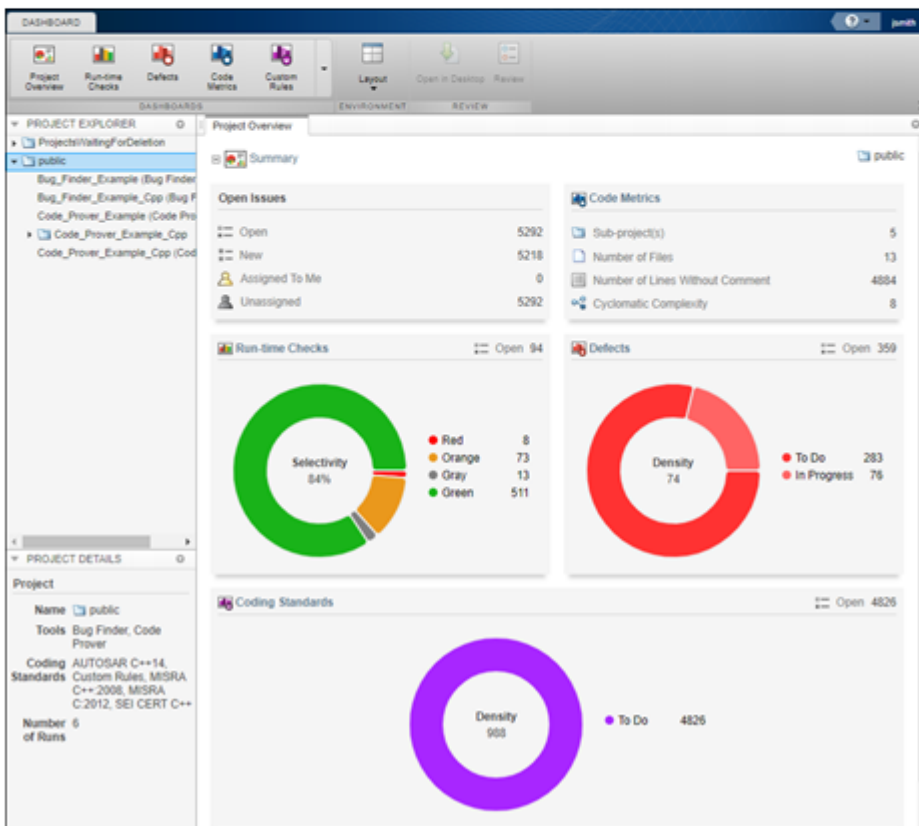
Polyspace Metrics

From: [] To: [] Maximum number of runs: 30 Refresh

ID	Project	Product	Mode	Language	Version	Date	Status
6	Bug_Finder_Example	Bug Finder		C	1.1	Feb 09, 2019	completed
5	Code_Prover_Example_Cpp	Code Prover	Unit By Unit	C++	1.0	Feb 09, 2019	completed (P4552)
4	Code_Prover_Example_Cpp	Code Prover	Integration	C++	1.0	Feb 09, 2019	completed (P4552)
3	Code_Prover_Example	Code Prover	Integration	C	1.0	Feb 09, 2019	completed (P4552)
2	Bug_Finder_Example_Cpp	Bug Finder		C++	1.0	Feb 09, 2019	completed
1	Bug_Finder_Example	Bug Finder		C	1.0	Feb 09, 2019	completed



```
polyspace-access -generate-migration-commands
polyspace-access -migrate
```



Requirements for Migration

The transfer of results from the Metrics repository to the Polyspace Access database requires the `polyspace-access` binary. This binary is available under the `polyspaceroot/polyspace/bin` folder with a Polyspace Code Prover or a Polyspace Code Prover Server installation. `polyspaceroot` is the Polyspace product installation folder, for instance `C:\Program Files\Polyspace Server\2019a`.

For more information on `polyspace-access`, see the Polyspace Bug Finder Server or Polyspace Code Prover Server documentation.

Migration of Results

To migrate results from Polyspace Metrics to Polyspace Access, follow these steps. You must be logged in to your Metrics server to complete this operation.

- 1 Identify the Metrics results repository location. The Polyspace Metrics results are stored in the `results-repository` folder at that location.

To view the path to this location, from the desktop interface, go to **Tools > Metrics Server Settings**. Or, at the command line, run the command `psqueue-check-config`.

By default, results are stored under `C:\Users\username\AppData\Roaming\Polyspace_RLData\results-repository` on Windows and `/home/username/.polyspace/results-repository` on Linux. `username` is your computer login user name.

- 2 Generate migration scripts.

Once you identify the folder of the repository from which you want to transfer results, define a migration strategy. You can choose to transfer all your projects or you can narrow down the scope of the transfer to a specific set of projects.

Specify a set of projects with the options listed in this table.

Option	Description
<code>-max-project-runs int</code>	Number of most recent analysis runs you want to migrate for each project. For instance, to migrate only the last two analysis runs of a project, specify 2.
<code>-project-date-after YYYY[-MM[-DD]]</code>	Only migrate results that were uploaded to Polyspace Metrics on or after the specified date.
<code>-product productName</code>	Product used to analyze and produce project findings, specified as <code>bug-finder</code> , <code>code-prover</code> , or <code>polyspace-ada</code> .
<code>-analysis-mode mode</code>	Analysis mode used to generate project findings, specified as <code>integration</code> or <code>unit-by-unit</code> .

For example, to transfer only Polyspace Bug Finder analysis results that you uploaded to Polyspace Metrics on or after June 2017, use this command:

```
polyspace-access -generate-migration-commands ^
C:\Users\username\AppData\Roaming\Polyspace_RLData\results-repository ^
-output-folder-path C:\Polyspace_Workspace\Migrate^
-project-date-after 2017-06^
-product bug-finder
```

The command outputs a migration script file for each project stored in `C:\Users\username\AppData\Roaming\Polyspace_RLData\results-repository` that matches the specified product and date. The migration scripts are stored under `C:\Polyspace_Workspace\Migrate`.

Before you continue, you can optionally open the migration scripts in a text editor and modify the `-project` or `-parent-project` parameters. The parameters correspond to the name of the project and the folder under which it is stored in Polyspace Access, respectively.

3 Migrate the projects.

After you generate the migration scripts, to transfer all the selected projects use those scripts with this migration command :

```
polyspace-access -host hostName ^  
-migrate -option-file-path ^  
C:\Polyspace_Workspace\Migrate
```

The command looks for migration scripts under `C:\Polyspace_Workspace\Migrate` and uploads the results to the Polyspace Access instance that you specify with *hostName*. Enter your Polyspace Access user name and password at the prompt.

hostName is the host name of the machine where the Polyspace Access **Gateway** service is running. You specify this host name in the URL of the Polyspace Access web interface. Depending on your configuration, you might also need to specify the `-port` and `-protocol` options in the migration command.

During the execution of a migration script, the command generates a temporary `STARTED` file. After each successful execution of a migration script, the command deletes the `STARTED` file and generates a corresponding `DONE` file in the same folder as the script. For example, the command generates `foo.started` during the execution of `foo.cmd`, and then `foo.done` once `foo.cmd` is done. Do not delete these `DONE` files until you have completed your migration from Metrics to Access.

Depending on the amount of data that you are transferring and on your network configuration, the migration might take a long time. You can interrupt the transfer, and then continue from where you left off at a later time. To stop the transfer, press **CTRL+C**. To restart the transfer:

- a Go to the folder where you store the migration scripts and open the `STARTED` file in a text editor. The file might be in a subfolder of the migration scripts folder.
- b Follow the instructions in the file, then reuse the same migration command that you used when you started the migration. The command skips projects that uploaded successfully.

If a project migration fails, go to the migration script folder. See the `FAILED` file for more information.

Differences in SQO Between Polyspace Metrics and Polyspace Access

After you migrate your projects from Polyspace Metrics to Polyspace Access, you might notice differences when you examine your code quality against “Software Quality Objectives” (SQO).

The difference is due to the way Polyspace Metrics and Polyspace Access calculate the thresholds for the quality objectives. Polyspace Metrics looks at individual files to determine whether your code achieves a given SQO threshold. For instance, if file `foo.c` does not achieve threshold `SQO2`, then the whole project does not achieve that threshold.

Polyspace Access looks at the whole project to determine whether your source code meets a given SQO threshold. Even if file `foo.c` does not achieve the threshold, the whole project can still meet the specified quality objective threshold.

See Also

More About

- “Register Polyspace Desktop User Interface” on page 1-33
- “Upload Results to Polyspace Access” on page 3-2

Quick Start Guide for Polyspace Server and Access Products

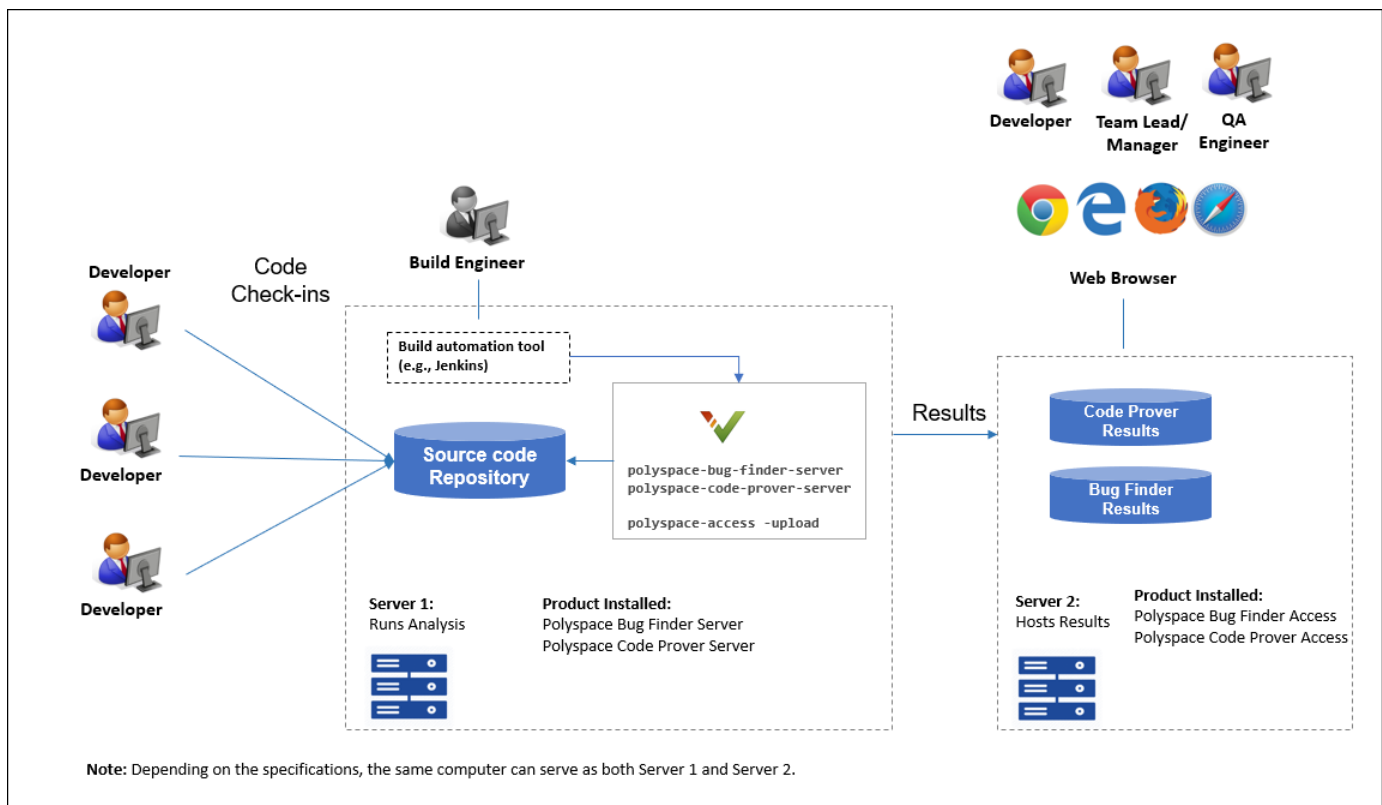
To avoid finding bugs late in the development process, run static analysis by using Polyspace products.

- **Polyspace Bug Finder** checks C/C++ code for bugs, coding standard violations, security vulnerabilities, and other issues.
- **Polyspace Code Prover** performs exhaustive checks for divide by zero, overflow, array access out of bounds, and other common types of run-time errors.

See also .

If you run Polyspace checkers regularly as part of continuous integration, you can protect against regressions from new code check-ins. To run Polyspace on a server during continuous integration, use **Polyspace Bug Finder Server** and **Polyspace Code Prover Server**. To host the Polyspace analysis results, use **Polyspace Bug Finder Access** and **Polyspace Code Prover Access**.

A typical workflow looks like this figure.



Installation

Prerequisites

Depending on the needs of your project, team or organization, you have decided to obtain a certain number of licenses of Polyspace Server and Polyspace Access products. This guide helps you to install individual instances of these products on a machine.

Install Polyspace Server

To install Polyspace Server products, download and run the MathWorks installer. Enter a license for the Polyspace Server products (or request a trial license). See also [Request a Trial License](#). The Polyspace Server products are installed in a separate folder from other MathWorks products. See also [“Install Polyspace Server and Access Products”](#) (Polyspace Code Prover Server).

Install Polyspace Access

Before installing Polyspace Access, consider the number of users who will potentially review Polyspace results simultaneously. The system requirements depend on the number of simultaneous reviewers. See also [“System Requirements for Polyspace Access”](#) on page 1-3.

Polyspace Access consists of several services: a database to manage results, a user manager to handle user logins, a web server to show results, and a gateway to handle communications. The services are deployed in Docker containers. You can start the services from a common interface called the Cluster Operator.

To install Polyspace Access:

- Download the installer as a zip file.
- Unzip the file and start the Cluster Operator. From the Cluster Operator interface, start the various services. See [“Install Polyspace Access”](#).

After installation, to see uploaded results, you and other reviewers can log in to:

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

Setting Up Polyspace Analysis

Prerequisites

You or your IT department in your organization must install the required number of Polyspace Server and Polyspace Access instances. This guide helps you to set up a Polyspace analysis as part of continuous integration using a single instance of Polyspace Server and Polyspace Access.

To check that your Polyspace Server and Polyspace Access installations can communicate with each other, see [“Check Polyspace Installation”](#) (Polyspace Code Prover Server).

Run Polyspace Server and Upload Results to Polyspace Access

You can run the Polyspace Server products at the command line of your operating system:

- To run the analysis, use the `polyspace-bug-finder-server` and `polyspace-code-prover-server` executables.
- To upload analysis results, use the `polyspace-access` executable. You can also use this executable to export the results from Polyspace Access as text files for archiving or email attachments.

You can run all Polyspace executables from the `polyspace/bin` subfolder of the Polyspace installation folder (for instance, `/usr/local/Polyspace Server/R2020a`, see also “Installation Folder” (Polyspace Code Prover Server)). To start running Polyspace Server by using sample C source files and sample scripts, see:

- “Run Polyspace Code Prover on Server and Upload Results to Web Interface” (Polyspace Code Prover Server)
- “Send Email Notifications with Polyspace Code Prover Results” (Polyspace Code Prover Server)

You can also preconfigure the Polyspace analysis options from your build command (makefile), and then append a second options file with analysis specifications such as checkers. See “Prepare Scripts for Polyspace Analysis” (Polyspace Code Prover Server).

If you have an installation of the Polyspace desktop products, you can prepare the analysis configuration in the user interface of the desktop products. You can then generate Polyspace options files to run during continuous integration. See “Configure Polyspace Analysis Options in User Interface and Generate Scripts” (Polyspace Code Prover Server).

Include Polyspace Runs in Continuous Integration by Using Tools Such as Jenkins

Once you have working scripts to run a Polyspace analysis, you can run those scripts at predefined intervals using continuous integration tools such as Jenkins and Bamboo. In Jenkins, you can use a Polyspace plugin to point to your Polyspace installations and send email notifications to developers after the analysis, based on criteria such as new defects.

From within the Jenkins interface, search for and install the Polyspace plugin. For a quick start on using the Jenkins plugin and sample scripts, see the Polyspace plugin GitHub repository. For the full workflow with Jenkins, see “Sample Scripts for Polyspace Analysis with Jenkins” (Polyspace Code Prover Server).

Create a Workflow for Result Reviewers

Depending on tools that you already use, you can set up a convenient workflow for result reviewers. For example:

Reviewers receive alerts for new results and log into Polyspace Access

- When new results are available, the continuous integration tool alerts a group of users. The email alert contains the Polyspace Access URL of the project where the results are uploaded.
- In the Polyspace Access interface, a reviewer can open this project URL, filter results based on files, and fix the issues or set a status for the results. See also:

- “Filter and Sort Results”
- “Address Polyspace Results Through Bug Fixes or Justifications”

Reviewers get customized email alerts with results in attachment

- Before upload to Polyspace Access, using the `-set-unassigned-findings` option of the `polyspace-access` executable, the continuous integration (CI) tool assigns owners to new analysis results based on file or component ownership or another criteria.
- After upload, using the `-export` option of the `polyspace-access` executable, the CI tool exports analysis results for each owner to a separate text file. The tool then sends the text file in an email attachment to the owner. The text file contains results with the corresponding URLs in the Polyspace Access interface.

If you use Jenkins as your CI tool, the Polyspace plugin in Jenkins directly supports this workflow. See “Sample Scripts for Polyspace Analysis with Jenkins” (Polyspace Code Prover Server).

- On receiving the email, the owner opens the attached text file, copies the URL of each result to their web browser and reviews the result.

Reviewers open tickets from bug tracking tools

- A reviewer, such as a quality engineer, reviews all new results and creates JIRA tickets for developers. See “Track Issue in Bug Tracking Tool”.
- Developers open each JIRA ticket and navigate to the corresponding Polyspace result in the Polyspace Access interface.

