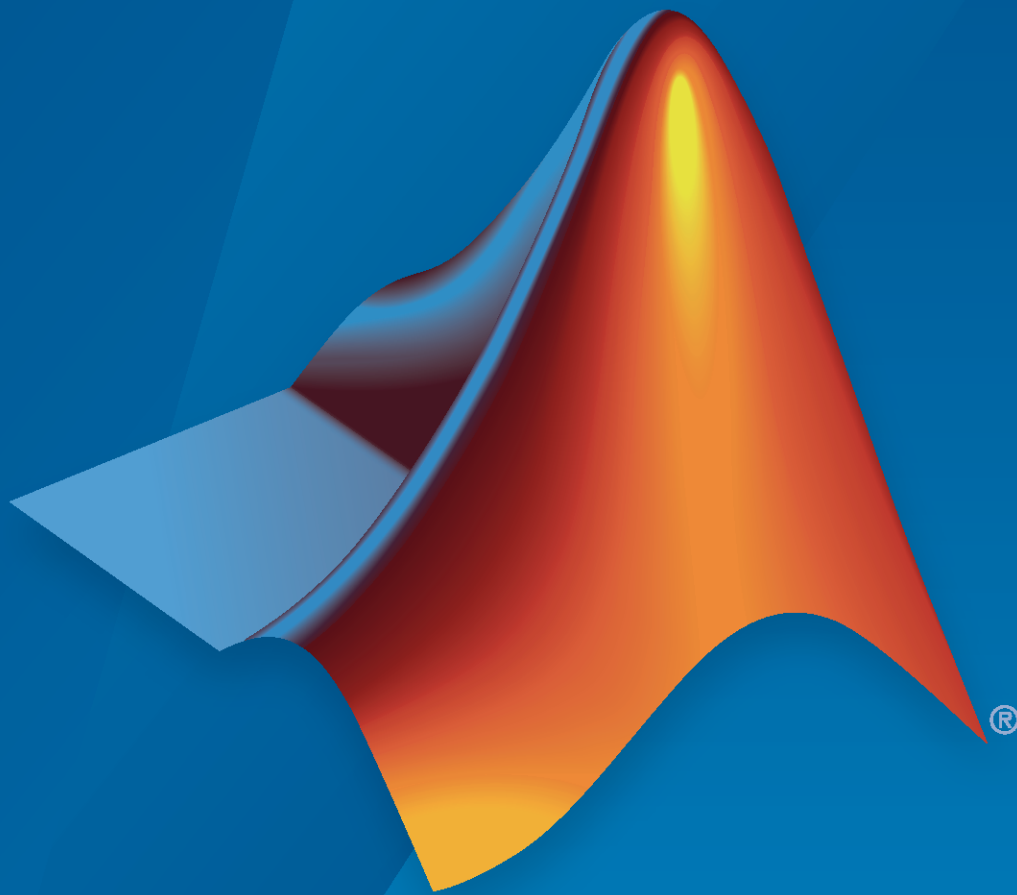# Polyspace® Bug Finder™ Access™

Getting Started Guide

MathWorks®

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

Revision History

Contents

Manage Polyspace Access License

2

Get Started with Polyspace Bug Finder Access

3

Install Polyspace Bug Finder Access

Polyspace Bug Finder Access Product Description

Review code analysis results and monitor software quality metrics

Polyspace Bug Finder Access provides a web browser interface to Polyspace static code analysis results and quality metrics. It includes a central repository for analysis results that enables team-based collaboration. Results from Polyspace Bug Finder Server™ can be published to Polyspace Bug Finder Access for triage and resolution. With Polyspace Bug Finder Access you can create and assign tickets in defect-tracking systems such as Jira.

Polyspace Bug Finder Access dashboards display information that you can use to monitor software quality, project status, number of defects, and code metrics (such as lines of code, cyclomatic complexity, and recursion). The dashboards help you graphically track compliance to coding rules standards such as MISRA C:2012 and AUTOSAR C++14 and security standards such as CERT-C and ISO/IEC TS 17961.

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 Engine on Linux.

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

  - Ubuntu
  - Debian
  - CentOS
  - Fedora

- The configuration of some Polyspace Access services requires the `openssl` toolkit to generate public and private keys. This toolkit is also required to configure Polyspace Access with HTTPS.
- To connect Polyspace Access with Polyspace desktop user interfaces over HTTPS, you must use the Java Platform, Standard Edition Development Kit (JDK). You use the JDK to generate a Java Key Store file.
- Polyspace Access licenses are network named user (NNU) licenses that require a license manager. Polyspace uses the FlexNet® Publisher (FLEXlm®) license manager. See "Install License Manager" on page 2-4.

Windows Requirements

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

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

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

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 Admin 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 and Port Configuration

Storage Configuration

- To ensure optimal data storage performance, use physical drives instead of networked storage solutions.

- The database is stored under *polyspaceAccessRoot*/appdata/polyspace-access where *polyspaceAccessRoot* is the folder where you unzip the Polyspace Access installation image. Make sure that you have adequate disk space for the Database mount point.

- It is a best practice to secure the database mount point with a RAID array and to back up the mount point regularly.

- Allocate this recommended amount of disk space for the mount points of the working directories of the Polyspace Access processes:

  - **Temporary upload directory**: 40 GB

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

  - **Upload directory**: 10 GB

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

  - **Storage directory**: 20 GB

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

  - **Working directory**: 10 GB

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

  - **Invalid results directory**: 50 GB

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

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

Network Port Configuration

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

Windows® PowerShell	`netstat -na	find "`*`portNumber`*`"`
Linux®	`netstat -na	grep `*`portNumber`*

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

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

- 9443 — Polyspace Access default port.
- 27000 — Inbound port of license manager that is required to manage license checkouts.
- 3268 — Lightweight Directory Access Protocol (LDAP) server default port. This port is not required if you do not use your company LDAP.

Depending on your Polyspace Access configuration, you might need to check the availability of a port for your bug tracking tool.

See Also

More About
- "Install Polyspace Access"
- "Database Backup" on page 1-38
- "Configure and Start the Cluster Admin" on page 1-13

Create a Linux Virtual Machine by Using Hyper-V

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

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

Prerequisites

Before you create the VM:

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

  Open 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 for Docker Engine or Docker Engine Enterprise(EE), see supported platforms for Docker Engine and Docker EE on Linux distros.

- Download and install the network license manager. See "Install License Manager" on page 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 on page 1-3 for Polyspace Access. The recommended minimum memory is 32 GB.

- For the **Configure Networking** step, select the switch that corresponds to the external connection type.

- For the **Connect Virtual Hard Disk** step, the size of the virtual hard drive must meet the requirements on page 1-3 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 <u>Get Docker Engine - Community for Ubuntu</u>.

  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 <u>scp</u> 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 pathTO\myFile.txt accessUser@access-vm-lnx:~
```

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

`pathTO` 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".

Prepare Your Installation

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

- **User Manager App**: Authenticates user logins against your company Lightweight Directory Access Protocol (LDAP) or against a custom internal database of users. The app issues signed JSON Web Tokens to authenticated users and provides a user interface to manage the custom database of users.

- **Issue Tracker App**: Manages the communication between Polyspace Access and your bug tracking tool (BTT) software and provides a user interface to create BTT tickets.

- **Polyspace Access App**: Manages results uploads to the Polyspace Access database and results exports from that database. Provides a user interface to review results.

Each app contains services that are deployed inside docker containers. A separate **Gateway** service handles communications between Polyspace Access and client machines.
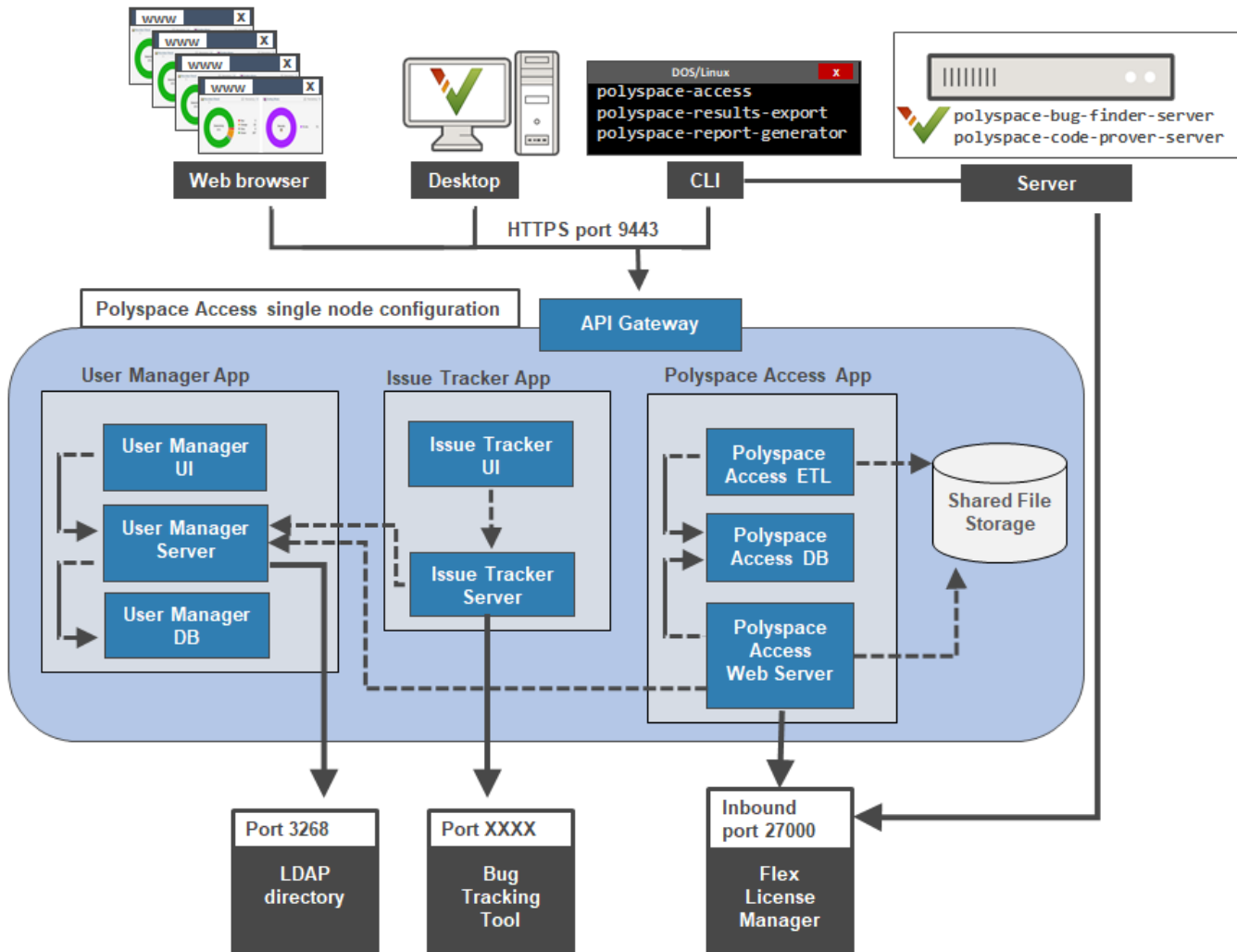
Before you begin your installation, decide whether to use the HTTPS protocol and how you to configure user authentication and bug tracking tool integration.

Prerequisites

- Install the required software and make sure that your system meets the minimum hardware requirements. See "System Requirements for Polyspace Access" on page 1-3.
- Verify that you have enough data storage available and that the ports that you use are available and not blocked by your firewall. See "Storage and Port Configuration" on page 1-5.
- Make sure that the license manager is installed and running, and that the license manager options file includes the users to whom you grant right-to-use privileges for Polyspace Access. See "Manage Polyspace Access License".
- To avoid any potential issues with license file operation, consider upgrading the network license manager software whenever you upgrade Polyspace Access. See "Update Network License Manager Software".
- Create and configure a Linux virtual machine (VM) if you are installing Polyspace Access inside a Linux VM on Windows Server 2016 or 2019. See "Create a Linux Virtual Machine by Using Hyper-V" on page 1-7.
- If you enable HTTPS to encrypt communications between Polyspace Access and client machines, obtain an SSL private key and a signed certificate from a certificate authority or use self-signed certificates. See "Configure Polyspace Access for HTTPS" on page 1-14.
- If you configure the Polyspace Access to use the HTTPS protocol, you must generate a Java® Key Store (JKS) file to enable communications between Polyspace Access and the Polyspace desktop interface. See "Generate a Client Keystore" on page 1-37.

User Manager Prerequisites

- The **User Manager** configuration requires the generation of an SSL private key file. See "Configure User Manager" on page 1-19. This private key must be different from the private key that you use for the HTTPS configurations.
- If you use your company LDAP to authenticate user logins, contact your LDAP administrator to:
  - Obtain the LDAP URL and LDAP base that your organization uses.
  - Obtain LDAP login credentials if access to the LDAP server is password-protected.
  - Discuss an LDAP search filter that enables you to retrieve specific subsets of users from the LDAP database. See LDAP filters.
- If you use LDAP configured over SSL (LDAPS), you must add the LDAP SSL certificate to the certificate trust store file that you use for Polyspace Access. See "Configure the User Manager for LDAP over SSL" on page 1-24.

Issue Tracker Prerequisites

- The **Issue Tracker** configuration requires the URL that you use to connect to your BTT interface.
- If you use the Jira software with the OAuth authentication method, you must first create an application link in Jira. See the first step on this page.
- If you use the Redmine BTT, contact your Redmine administrator to obtain the Redmine API key.

- If you use a BTT configured by using HTTPS, you must add the BTT SSL certificate to the certificate trust store file that you use for Polyspace Access. See "Add BTT Instance Configured by Using HTTPS" on page 1-28.

Configure and Start the Cluster Admin

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

Prerequisites

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

- You have downloaded the Polyspace Access installation image ZIP file. To download the ZIP file, go to the MathWorks® download page, click the **Download Rxxxxy** 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 Additional Rxxxxy Product Downloads. Rxxxxy corresponds to a release number, for instance R2019b.

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

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

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

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

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

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

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

Unzip Installation Image and Start Cluster Admin Agent

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

- `admin-docker-agent` and `admin-docker-agent.exe`.
- `polyspace-access-*.tar` files.
- `issue-tracker-*.tar` files.
- `usermanager-*.tar` files.
- `admin.tar`.
- `gateway.tar`.
- `issuetracker.tar`.
- `usermanager.tar`.
- `polyspace-access.tar`.

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

```
admin-docker-agent
```

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

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

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

By default, the **Cluster Admin** uses the HTTP protocol and starts with host name localhost and port 9443. To configure the **Cluster Admin** with HTTPS, see "Configure Polyspace Access for HTTPS" on page 1-14. If the port is already in use, you get `Permission denied` error message. Use the flag `--port` to specify a different port number, for instance:

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

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

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

To view the new password, restart the binary.

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

Configure Polyspace Access for HTTPS

To encrypt the data between Polyspace Access and client machines, configure the **Cluster Admin** with the HTTPS protocol. To complete the configuration, provide an SSL certificate and the private key that you used to generate the certificate as PEM files.

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

It is recommended that you 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.

Secure your private key by following best practices such as:

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

The configuration of HTTPS for the **Cluster Admin** enables HTTPS for the API Gateway service. This service handles all communications between the other Polyspace Access services and client machines.

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

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 `admin-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, `admin_cert.cer`.

3 Start `admin-docker-agent` and use the generated private key and signed certificate. Specify the FQDN *hostName* and the full path to the certificate trust store file `ca-certificates.crt`:

```
./admin-docker-agent --hostname hostName\
--ssl-cert-file fullPathTo/admin_cert.cer \
--ssl-key-file fullPathTo/myKey.key \
--ssl-ca-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 **Cluster Admin** web interface, your browser considers the connection secure if the browser uses the certificate trust store that you specify for `--ssl-ca-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 `admin-docker-agent` binary.

```
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 `admin-docker-agent` binary and use the generated `certificate.pem` and `private_key.pem` files. Specify the FQDN *hostName*.

```
./admin-docker-agent --hostname hostName\
--ssl-cert-file fullPathTo/certificate.pem \
--ssl-key-file fullPathTo/private_key.pem \
--ssl-ca-file  fullPathTo/certificate.pem
```

The *hostName* you specify in this command must match the *hostName* you specified in step 1. The self-signed `certificate.pem` file is also used as the certificate trust store file. *fullPathTo* is the full file path. If you use relative paths, you get an error message.

When you open the **Cluster Admin** web interface, your browser shows a warning about the certificate being untrusted.

Open the Cluster Admin Interface

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

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

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

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, return to the **Cluster Dashboard** and click **Restart Apps** for the changes to take effect. To save partially filled settings, clear **Validate on Save**.

Cluster Admin

Account ▾

Cluster Settings

Return to Dashboard

User Manager

Use internal directory	☑
Internal directory database volume	*(blurred)*
Internal directory database username	um
Internal directory database password	•••••
Administrator sign-in IDs	admin
Initial administrator password	••••
Authentication token expiration (sec)	86400
Authentication private key file	*(blurred)*
API keys and user IDs	5ea34345-a03b-4a20-821e-f10e45e0e863,jsmith

Polyspace Access Web Server

Upload directory	*(blurred)*
Temporary upload directory	*(blurred)*
License file	*(blurred)*

Validate Now

☑ Validate on Save   Save   Cancel

1-17

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

See Also
`admin-docker-agent`

More About
- "Configure and Start the Cluster Admin" on page 1-13
- "Configure User Manager" on page 1-19
- "Configure Issue Tracker" on page 1-27
- "Configure Polyspace Access App Services" on page 1-30

Configure User Manager

The **User Manager** manages the authentication of the Polyspace Access user logins. Depending on your configuration, user logins are authenticated by checking user names and passwords against information in your company Lightweight Directory Access Protocol (LDAP) database, or against user credentials stored in the **User Manager** internal directory.

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, return to the **Cluster Dashboard** and click **Restart Apps** for the changes to take effect. To save partially filled settings, clear **Validate on Save**.

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

Setting	Description
Use Internal directory	Enable this option to create and manage a custom list of user credentials. The **User Manager** stores these credentials in an internal database. See "Authenticate Users from Internal Directory" on page 1-25. Disable this option to authenticate users by using your organization LDAP. See "Configure LDAP" on page 1-22
Internal directory database settings	These options are visible only if you select **Use Internal directory**. See "Authenticate Users from Internal Directory" on page 1-25.
LDAP settings	These options are visible only if you clear **Use Internal directory**. See "Configure LDAP" on page 1-22.

Setting	Description
Administrator sign-in IDs	Enter a comma-separated list of user names to set specific users as Polyspace Access administrators. A user that has Polyspace Access administrator privileges can:
	• 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" on page 3-18.
	• Remove or add project owners.
	• "Manage LDAP Users in Polyspace Access" on page 1-26
	If you use your company LDAP, the use rname must match an existing user in the LDAP directory.
	To remove a user as a Polyspace Access administrator:
	1 Remove the user name from the list, save your changes, then restart the apps.
	2 After the restart, a Polyspace Access administrator must unassign the user from all top-level folders in the **PROJECT EXPLORER**, in the Polyspace Access web interface, by using the context menu. The administrator can also perform this task at the command line by using the `-unset-role` flag with the `polyspace-access` binary. For more information, see `polyspace-access -unset-role -h`.
	If you select **Use Internal directory**, the users you specify in this field are also **User Manager** administrators. These users can:
	• Create, edit, or delete other users in the **User Manager** interface, except for other administrators.
	• Reset user passwords, except for other administrators.
	To remove a **User Manager** administrator:
	1 Remove the user name from the list, save your changes, then restart the apps.
	2 After the restart, an administrator must log into the **User Manager** interface to delete that user from the database.
Initial administrator password	Password that you use to log into the **User Manager** interface. This field is visible only if you select **Use Internal directory**.
	The default password is "`pass`".

Setting	Description
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 into Polyspace Access, your license is checked out and your session refreshes periodically to keep it from expiring. The session ends once you explicitly log out or close your web browser and your license is checked back in. If your browser closes unexpectedly, your license stays checked out until the session expires. When you set the expiration time, consider: • If the expiration time is too short, frequent users are prompted to log back in frequently. On large teams, the license server experiences a high volume of license checkins and checkouts. • If the expiration time is too long, the session time of less frequent users might be overestimated in the license logs. Use this settings to set the licensing timeout. Polyspace Access ignores the license timeout value that you set through the license manager options file (`MLM.opt`) by using the `TIMEOUT feature seconds` syntax.
Authentication private key file:	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. The **User Manager** service does not support password-protected private keys. You can generate a private key by using the `openssl` utility. For example: `openssl genrsa -out private.pem 2048` Restrict access to this private key to only those administrators who manage the **User Manager** service. Do not reuse the private key that you use to generate the SSL certificates, which you provide if you enable the HTTPS protocol.

Setting	Description

API keys and user IDs	Enter an API key value and user name pair to assign an API key to a user. For example, to assign an API key to `jsmith`, enter:
	
	`5ea34345-a03b-4a20-821e-f10e45e0e863,jsmith`
	
	To assign API keys to other users, enter additional API key and user name pairs on separate lines. Each API key value must be unique.
	
	You pass the API key with the flag `-api-key` to these commands that require Polyspace Access credentials:
	
	• `polyspace-access`
	• `polyspace-results-export`
	• `polyspace-report-generator`
	
	The commands use the API key as a login credential for the corresponding users. If a user updates his or her password, you do not have to update the API key. If you use the API key as part of an automation script, make sure that the user associated with the key has enough permissions to perform all the operations in the script. See "Manage Project Permissions" on page 3-18.
	
	You can assign any combination of alphanumeric characters as an API key to a user. For example, to generate universally unique identifiers (UUID) for the API key, use these commands:
	
	• **Windows PowerShell**
	
	`[guid]::NewGuid()`
	• **Linux**
	
	`uuidgen`

Configure LDAP

To use the LDAP server of your organization, in the **User Manager** settings, clear **Use Internal directory**. Contact your network administrator to obtain the LDAP URL, base, and any other setting and to determine whether your LDAP server uses the Active Directory Global Catalog feature.

LDAP URL	Enter the LDAP URL as:
	
	`ldap://HOST:PORT`
	
	HOST is the LDAP host and *PORT* is the LDAP port number. The LDAP server uses different default port numbers if you configure it to use the global catalog. See "Configure User Manager for LDAP Server That Uses Global Catalog" on page 1-24.
	
	If you have configured your LDAP server over SSL, enter the URL as:
	
	`ldaps://HOST:PORT`
	
	For additional LDAPS configuration steps, see "Configure the User Manager for LDAP over SSL" on page 1-24.
	
	Because communications between the LDAP server and clients are not encrypted, the configuration and use of LDAP over SSL (LDAPS) is recommended.

LDAP username	User name of user who has read permission to the LDAP server. Leave this field blank if your access to the LDAP server is not password-protected.
LDAP password	Password of user who has read permission to the LDAP server. Leave this field blank if your access to the LDAP server is not password-protected. The password is stored in the `settings.json` 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 `admin-docker-agent` binary.
LDAP base	You can retrieve this parameter by using an LDAP explorer tool. For instance, connect to your LDAP server through Apache Directory Studio and open the properties for your connection. In the **Browser Options**, click **Fetch Base DNs** to get the LDAP base.
LDAP search filter	Use the search filter to retrieve a subset of users from the LDAP database. Polyspace Access loads this subset on startup instead of loading all users in your organization. Loading a smaller number of users for authentication improves the performance of Polyspace Access. Specify the search filter as *attribute=value*, for instance `CN=test*` matches all users who have a common name (CN) attribute that starts with "test". Use parentheses to combine multiple filter expressions in an AND (&) or OR (|) clause. For instance, `(|(CN=jdoe)(department=foo))` matches all users who have CN attribute "jdoe" or department attribute "foo". The default search filter is `objectClass=organizationalPerson`. For more information about search filters, see the LDAP filters. To check whether a search filter is returning a subset of users, enter a user name from that subset in the **Administrator sign-in IDs** field and click **Validate Now** at the bottom of the page. You get a warning if the user name cannot be found.

Other LDAP settings	Leave these settings unchanged unless instructed otherwise by your LDAP administrator. Polyspace Access does not use the LDAP display name, email, and image attributes.

Configure the User Manager for LDAP over SSL

If you use an LDAP configured over SSL (LDAPS), add the LDAPS SSL certificate to the certificate trust store file that you specify by using `--ssl-ca-file` when you configure the **Cluster Admin** with HTTPS. See "Configure Polyspace Access for HTTPS" on page 1-14. Depending on your trust store file, the LDAP SSL certificate might already be included in the trust store.

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

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

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

To complete the configuration, press **CTRL+C** and start the `admin-docker-agent` binary. Use `combined_cert.pem` as a trust store file:

```
./admin-docker-agent --hostname hostName\
--restart-gateway
--ssl-cert-file fullPathTo/certificate.pem \
--ssl-key-file fullPathTo/private_key.pem \
--ssl-ca-file  fullPathTo/combined_cert.pem
```

The flag `--restart-gateway` is required whenever you make changes to the HTTPS configuration. The *hostName* is the host name that you specified for the certificate common name. *fullPathTo* is the full file path.

If you did not configure the **Cluster Admin** by using HTTPS, start the `admin-docker-agent` and specify the LDAP SSL certificate for the trust store file:

```
./admin-docker-agent --hostname hostName\
--restart-gateway
--ssl-ca-file  fullPathTo/ldaps_cert.pem
```

Configure User Manager for LDAP Server That Uses Global Catalog

The global catalog (GC) is a mechanism that enables you to add users from different Active Directory® (AD) servers to Polyspace Access without having to provide information about those servers. For more information, see Global Catalog. If your LDAP server is configured to use the GC, you must specify a GC-specific port number when you provide the LDAP URL to the **User Manager** service. If you use secure LDAP (LDAPS), the default port for LDAP servers that use GC is 3268 or 3269. To determine whether your LDAP server is configured to use the GC, contact your LDAP administrator.

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

```
LDAP server 'ldap://dc-00.ad.mathworks.com:389' did not recognize
base DN 'DC=ad,DC=mathworks,DC=com' and search base ''.
...
Unprocessed Continuation Reference(s)
```

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

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

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

Authenticate Users from Internal Directory

If you cannot or choose not to use the LDAP server of your organization, you can create users who have custom user names and passwords in the **User Manager** interface. In the settings, select **Use Internal directory**. The **User Manager** checks user logins against the user credentials that you create. The credentials are stored in an internal directory database.

Internal directory database volume	Specify the full path to the folder where you store the database. By default, the database is stored under `appdata/usermanager` in the folder where you extracted the Polyspace Access installation image.
Internal directory database username	Use the user name and password that you specify in these two field if you need to interact with the internal database by using PostgreSQL commands. The default user name is 'UM" and the default password is "trust".
Internal directory database password	

To create or manage users, return to the **Cluster Dashboard**, restart the **User Manager** app, then click **Manager users** to open the **User Manager** interface. Only users listed in the **Administrator sign-in IDs** field of the **User Manager** settings can open the **User Manager** interface and manage users.

When you create a user, add the user sign-in ID to the `MLM.opts` file to grant that user right-to-use privileges for Polyspace Access. See "Manage Named Users for Polyspace Access" on page 2-5.

Manage LDAP Users in Polyspace Access

When the **Polyspace Access Web Server** service starts, Polyspace Access loads a list of users to its database from your organization's LDAP database or from the **User Manager** internal database. You can select users from only this list when you assign analysis findings or manage permission for a project or folder.

Polyspace Access periodically checks for and adds new users to its database from the LDAP database or the **User Manager** internal database. Polyspace Access does not remove users from its database of users if you remove a user from the LDAP database or from the **User Manager** internal database, even if you restart the **Polyspace Access Web Server** service.

To remove users from the Polyspace Access list of users:

1   Click **Restart Apps** in the **Cluster Dashboard**.

2   In a web browser, enter the URL that you use to "Open the Polyspace Access Web Interface" on page 1-35 and append `/users/list/removed` to the URL, for example `https://access-machine.company.com:9443/users/list/removed`.

    You must be logged in as a user who has **Administrator** privileges. To set a user as **Administrator**, see "Configure User Manager" on page 1-19.

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

See Also

More About
- "Configure Issue Tracker" on page 1-27
- "Configure Polyspace Access App Services" on page 1-30
- "Register Polyspace Desktop User Interface" on page 1-36
- "Start Polyspace Access and Upload Examples" on page 1-32

Configure Issue Tracker

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

On the **Cluster Dashboard**, click **Configure Apps** to go to the **Cluster Settings**. Whenever you change the settings, return to the **Cluster Dashboard** and click **Restart Apps** for the changes to take effect. To save partially filled settings, clear **Validate on Save**.

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

Setting	Description
Provider	Specify your bug tracking tool (BTT) provider. Polyspace Access supports integration with the Jira Software and Redmine BTT.
Jira deployment type	Specify whether your Jira instance is hosted on a local server on a cloud service provider. This field is available only if you select Jira as a provider.
Jira URL	Specify the URL of the Jira instance for your organization, for example, `https://jira.mycompany.com`. If your Jira instance is configured by using HTTPS, see "Add BTT Instance Configured by Using HTTPS" on page 1-28. This field is available only if you select Jira as a provider.
Authentication method	Specify the method that the **Issue Tracker** uses to authenticate logins to Jira from Polyspace Access users. This field is available only if you select Jira as a provider.
OAuth1 callback URL **OAuth1 consumer key** **OAuth1 private key file**	If you select OAuth as an authentication method, your Jira administrator must first create an application link in Jira. The Jira administrator specifies an application URL (the URL for Polyspace Access) and generates an RSA public/private keypair. For more information, see step 1 on this page. The **OAuth1 callback URL** must match the application URL specified in Jira. The **OAuth1 private key file** and **OAuth1 consumer key** must match the private and public key, respectively, that the Jira administrator generates to configure the application link. These fields are available only if you select Jira as a provider and OAuth as an authentication method.

Setting	Description
Redmine URL	Specify the URL of the Redmine instance for your organization, for example, `https://redmine.mycompany.com`. If your Redmine instance is configured by using HTTPS, see "Add BTT Instance Configured by Using HTTPS" on page 1-28. This field is available only if you select Redmine as a provider.
Redmine API key	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**. The **Issue Tracker** 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 that have custom fields if these fields are required fields, except if the fields are all numeric values or string only values.

- To create a redmine ticket from Polyspace Access, the user name used to log into Polyspace Access must match the user name 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, do not enable dark feature. See Enable Dark Feature in Jira.

Add BTT Instance Configured by Using HTTPS

If your BTT instance is configured by using HTTPS, add the BTT SSL certificate to the certificate trust store file that you specify with `--ssl-ca-file` when you configure the **Cluster Admin** by using HTTPS. See "Configure Polyspace Access for HTTPS" on page 1-14. Depending on your trust store file, the BTT SSL certificate might already be included in the trust store.

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

```
cat trust_store.pem btts_cert.pem > combined_cert.pem
```

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

To complete the configuration, press **CTRL+C** and start the `admin-docker-agent` binary. Use `combined_cert.pem` as a trust store file:

```
./admin-docker-agent --hostname hostName\
--restart-gateway
--ssl-cert-file fullPathTo/certificate.pem \
--ssl-key-file fullPathTo/private_key.pem \
--ssl-ca-file  fullPathTo/combined_cert.pem
```

The flag `--restart-gateway` is required whenever you make changes to the HTTPS configuration. The *hostName* is the host name that you specified for the certificate common name. *fullPathTo* is the full file path.

If you did not configure the **Cluster Admin** by using HTTPS, start the `admin-docker-agent` and specify the BTT SSL certificate for the trust store file:

```
./admin-docker-agent --hostname hostName\
--restart-gateway
--ssl-ca-file  fullPathTo/btts_cert.pem
```

See Also

More About
*   "Configure User Manager" on page 1-19
*   "Configure Polyspace Access App Services" on page 1-30
*   "Register Polyspace Desktop User Interface" on page 1-36
*   "Start Polyspace Access and Upload Examples" on page 1-32

Configure Polyspace Access App Services

Polyspace Access Database

Setting	Description
Data volume	Specify the full path to the folder where you store the database. By default, the database is stored under `appdata/polyspace-access` in the folder where you extracted the Polyspace Access installation image. Deleting the **Polyspace Access Database** service and uninstalling Polyspace Access does not erase the results that you uploaded to the database from the data volume. To delete a data volume and its content, manually delete the folder where you store the database.
Password	Specify a password to authenticate connections to the database. The different Polyspace Access services use this password when they communicate with the database. Use this password if you are prompted for one while performing a database backup on page 1-38 or clean up on page 1-40. You can specify a password only for a new database. To change the password of an existing database, use the PostgreSQL utilities instead. If you update the password of the database by using the PostgreSQL utilities, you must update the password in this field as well. The default password of the Polyspace Access database is "trust". To see the current database password, run this command. • **Windows** `docker inspect polyspace-access-db-main	FIND "POSTGRES_PASSWORD"` • **Linux**  `docker inspect polyspace-access-db-main	grep POSTGRES_PASSWORD`

Polyspace Access ETL

Setting	Description
Storage directory	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See "Storage and Port Configuration" on page 1-5.
Invalid results directory	
Working directory	By default, these folders are stored under `appdata/polyspace-access` in the folder where you extracted the Polyspace Access installation image.
Upload directory	The **Upload directory** path must be the same for the **Polyspace Access Web Server** and **Polyspace Access ETL** services.

Polyspace Access Web Server

Setting	Description
Upload directory: **Temporary upload directory:**	Specify the full path to folders with adequate write permissions. On Windows systems, the paths must point to local drives. See "Storage and Port Configuration" on page 1-5. By default, these folders are stored under `appdata/polyspace-access` in the folder where you extracted the Polyspace Access installation image. The **Upload directory** path must be the same for the **Polyspace Access Web Server** and **Polyspace Access ETL** services.
License file:	Specify the full path to the `network.lic` 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.

See Also

More About

- "Configure User Manager" on page 1-19
- "Configure Issue Tracker" on page 1-27
- "Register Polyspace Desktop User Interface" on page 1-36
- "Start Polyspace Access and Upload Examples" on page 1-32

Start Polyspace Access and Upload Examples

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

The indicator turns green when an app starts. The **Polyspace Access Web Server** service might take a few moments to start.

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

Use the output log to try to identify the cause of the stopped service. If you need additional assistance, see "Contact Technical Support About Polyspace Access Issues".

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

Configure Polyspace Access to Restart Automatically

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

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

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

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

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

Note Stopping the Polyspace Access services by using the Cluster Admin interface on Windows Server 2019 might result in a slow response time.

Upload Examples

Upload Results from the Command line

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

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

polyspaceroot is the path to your Polyspace installation. *hostname* is the fully qualified domain name (FQDN) of the machine that hosts Polyspace Access. *port* is the port number that you specified when starting the admin-docker-agent binary. For more information on uploading results from the command line, see "Upload Results at Command Line" on page 3-3.

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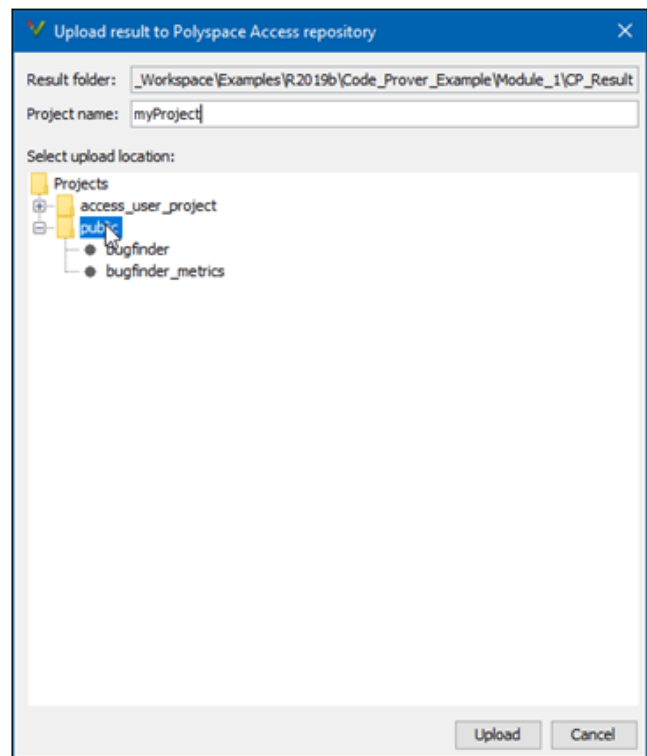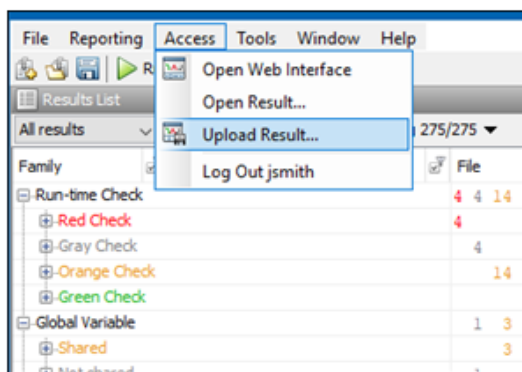
You can also upload to the Polyspace Access database by selecting a result in the **Project Browser** pane and using the context menu.

You must configure the desktop interface to communicate with Polyspace Access. See "Register Polyspace Desktop User Interface" on page 1-36.

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

See Also

More About

- "Review Results in Polyspace Bug Finder Access"

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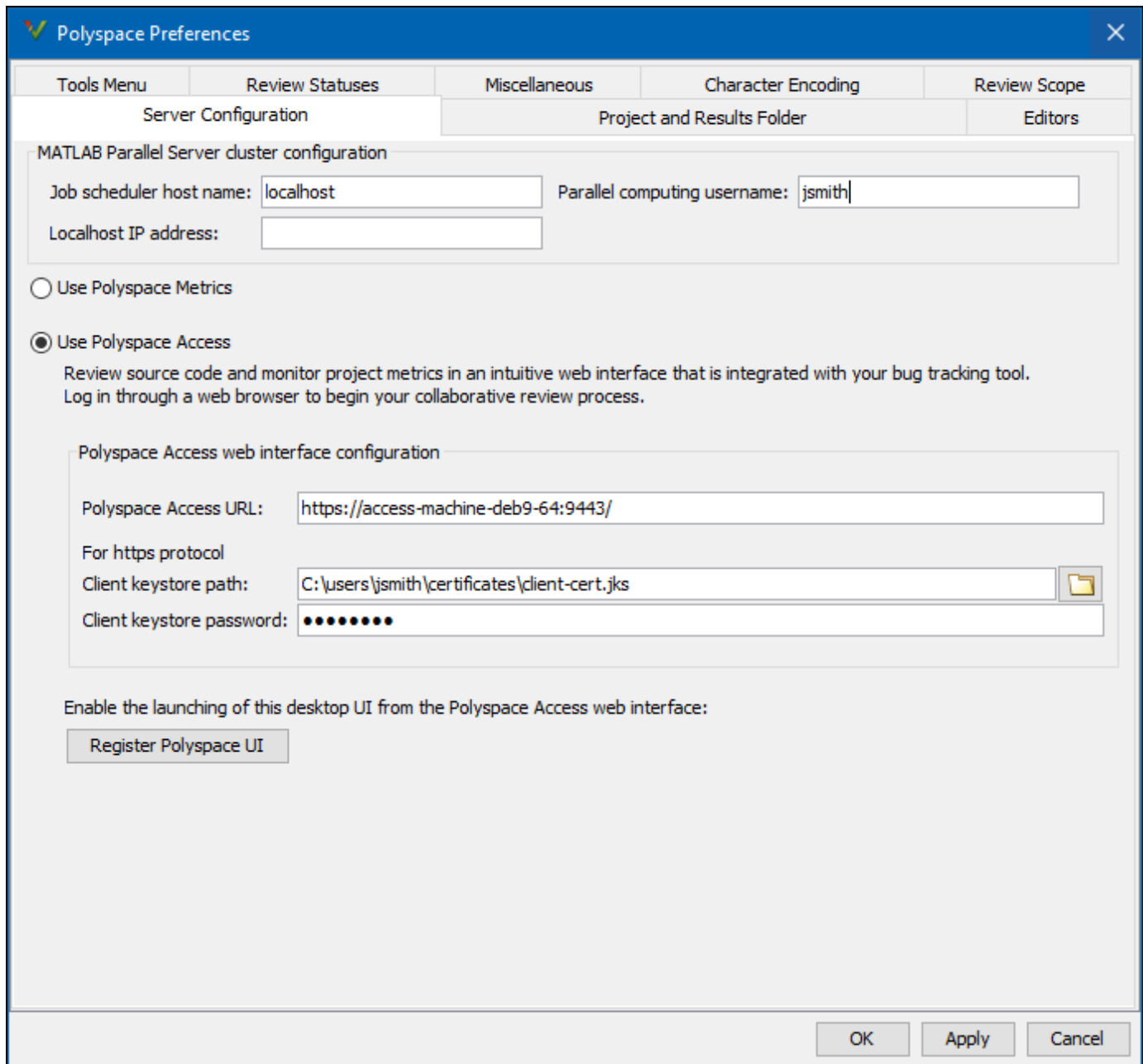
In the **Server Configuration** tab, complete these fields:

- **Polyspace Access URL**: Specify the URL you use to log into the Polyspace Access interface as `http(s)://hostName:port`. If you do not know the URL, contact your Polyspace Access administrator.

- **Client keystore path**: path to the key store file where you imported the signed certificate use to configure Polyspace Access with HTTPS. See "Generate a Client Keystore" on page 1-37.

  If the Polyspace Access URL does not use HTTPS, leave this field blank.

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

  If the Polyspace Access URL does not use HTTPS, leave this field blank.

To 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 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 Polyspace Access with HTTPS to the JKS file you generate. See "Configure Polyspace Access for HTTPS" on page 1-14.

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 1-33

Database Backup

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

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

Create Database Backup

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

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

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

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

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

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

3   To restart the **Polyspace Access ETL** and **Polyspace Access Web Server** services.

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

Restore Database from Backup

To recover your data from a database backup, use the `psql` utility. This utility is available in the `polyspace-access-db-main` container. The operation restores the data and the user permissions for the Polyspace Access projects. For example, you can restore your database from a backup stored in file `backup_db.sql`. You complete some steps in the **Cluster Admin** interface. Other steps require a terminal on the server that hosts the **Polyspace Access Database** service. On Linux, you might need superuser privileges to complete this operation.

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

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

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

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

       databaseFolderPath is the folder path that you specify in the **Data volume** field of the **Polyspace Access Database** service in the Admin **Cluster Settings**, for example, `/local/Polyspace/R2020b/appdata/polyspace-access/db`.

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

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

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

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

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

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

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

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

See Also

More About
- "Storage and Port Configuration" on page 1-5
- "Install and Manage Polyspace Access"

Database Clean Up

To optimize the performance of the Polyspace Access database, perform regular database clean up operations such as vacuuming and the deletion of old or obsolete projects. It is recommended that you back up your database before you perform a clean up operation. See "Create Database Backup" on page 1-38.

Perform Database Vacuuming

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

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

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

Note Stopping the Polyspace Access services by using the Cluster Admin interface on Windows Server 2019 might result in a slow response time.

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

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

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

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

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

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

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

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

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

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

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

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

Delete Outdated Projects

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

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

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

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

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

    - `clean_project projectPath DATE YYYY-MM-DD`

      The command deletes project runs that were uploaded before `YYYY-MM-DD`.

    - `clean_project projectPath MAXRUNS NNN`

      `NNN` is an integer. The command keeps the `NNN` most recent runs. To delete all the project runs, use `MAXRUNS 0`.

    - `clean_project projectPath AGE DDD`

      `DDD` is the number of days. To remove recently uploaded results, use this option. The command deletes project runs that are older than `DDD` days.

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

    ```
    delete_project projectPath
    ```

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

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

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

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

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

2 Copy the file to the **Storage directory** of the **Polyspace Access ETL** service:

```
cp cleanup.pscauto /local/Access_install_dir/polyspace/storage
```

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

You can also perform an automatic cleanup on a specific project every time you upload a run to that project. To keep only the 10 most recent runs every time you upload a result to `public/Bug_Finder_Example (Bug Finder)`, save these commands to your `.pscauto` file.

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

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

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

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

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

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

See Also

More About

- "Storage and Port Configuration" on page 1-5
- "Install and Manage Polyspace Access"

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 1-38.

1 Inform Polyspace Access users of the upcoming update or uninstallation.

2 Verify that the **Cluster Admin** agent. Use the command:

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

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

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

   Note Stopping the Polyspace Access services by using the Cluster Admin interface on Windows Server 2019 might result in a slow response time.

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

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

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

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

```
docker stop gateway \
usermanager-ui-main \
usermanager-db-main \
polyspace-access \
issuetracker \
usermanager
```

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

5 Go to the folder where you unzipped the installation image for Polyspace Access and delete the `admin-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 Admin" on page 1-13.

• If you reuse an existing configuration for the **User Manager**, **Issue Tracker**, and **Polyspace Access** apps, services, copy your backup `settings.json` file into the same folder as the `admin-docker-agent` and start the binary.

• To apply new configuration settings do not reuse the `settings.json` file.

To avoid any potential issues with license file operation, consider upgrading the network license manager software whenever you upgrade Polyspace Access. See "Update Network License Manager Software".

See Also

`admin-docker-agent`

More About

- "Configure and Start the Cluster Admin" on page 1-13
- "Configure User Manager" on page 1-19
- "Configure Issue Tracker" on page 1-27
- "Configure Polyspace Access App Services" on page 1-30

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 2-5 to add or remove users. Make sure that you install the license manager before you continue. See "Install License Manager" on page 2-4.

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 **User Manager** internal directory. See "Configure User Manager" on page 1-19.

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 --fqdn`

- *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 part, 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	cd *LM_Folder*\etc\win64 lmgrd.exe -c *pathToLicense* -l lm_log.log On Windows, you can also use lmtool.exe and go to the **Start/Stop/Reread** tab to start the license manager.
Linux	cd *LM_Folder*/etc/glnx64 ./lmgrd -c *pathToLicense* -l lm_log.log

LM_Folder is the folder where you installed the license manager.

pathToLicense is the path to the `license.dat` file that you saved on the machine where you installed the license manager. The command starts the license manager and outputs a log file `lm_log.log`. Refer to this log file for debugging purposes.

Note The license file path listed in the log and error messages of the license manager might not correspond to *pathToLicense*. The **Polyspace Access Web Server** service remaps *pathToLicense* to an internal path inside the docker container.

---

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

Windows	Use lmtool.exe and go to the **Config Services** tab, then check that **Start Server at Power Up** and **Use Services** are selected.
Linux	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 `/etc/inti.d` 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.

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

The license manager is shipped with the Polyspace Access software. The license manager binaries and utilities are located in `accessRoot/lm`. `accessROOT` is the folder where you extracted your Polyspace Access installation image.

To run the license manager on a separate server from the server where you run Polyspace Access, copy the folder that corresponds to your operating system from `accessRoot/lm`, for instance `accessRoot/lm/glnxa64`, to that separate server. The license manager folder includes these binaries:

- `lmgrd`: Core license manager binary. Use this binary to start the license manager from the command line. For a list of useful commands, enter `lmgrd -h`.
- `mlm`: The MATLAB vendor daemon.
- `lmutil`: a suite of tools for administering the license manager at the command line. For a list of useful commands, enter `lmutil -h`.
- `lmtools.exe` (Windows only): Graphical user interface for administering the license manager.
- For Linux systems, the license manager folder also includes command-line utilities. See "Using Command-Line Utilities".

To avoid any potential issues with license file operation, consider upgrading the network license manager software whenever you upgrade Polyspace Access. See "Update Network License Manager Software".

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	cd *LM_Folder*\etc\win64 lmutil lmdown lmgrd.exe -c *pathToLicense* -l lm_log.log On Windows, you can also use lmtool.exe and go to the **Start/Stop/Reread** tab to stop and restart the license manager.
Linux	cd *LM_Folder*/etc/glnx64 ./lmutil lmdown ./lmgrd -c *pathToLicense* -l lm_log.log

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	cd *LM_Folder*\etc\win64 lmutil.exe lmstat -c *pathToLicense* -a On Windows, you can also use lmtool.exe and go to the **Server Status** tab to stop and restart the license manager.
Linux	cd *LM_Folder*/etc/glnx64 ./lmutil lmdown ./lmutil lmstat -c *pathToLicense* -a

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 Bug Finder 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 user interface to view statistics about the quality of your code and to triage and review individual results.

Note You can upload up to 2GB of results per upload to Polyspace Access.

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 1-36.

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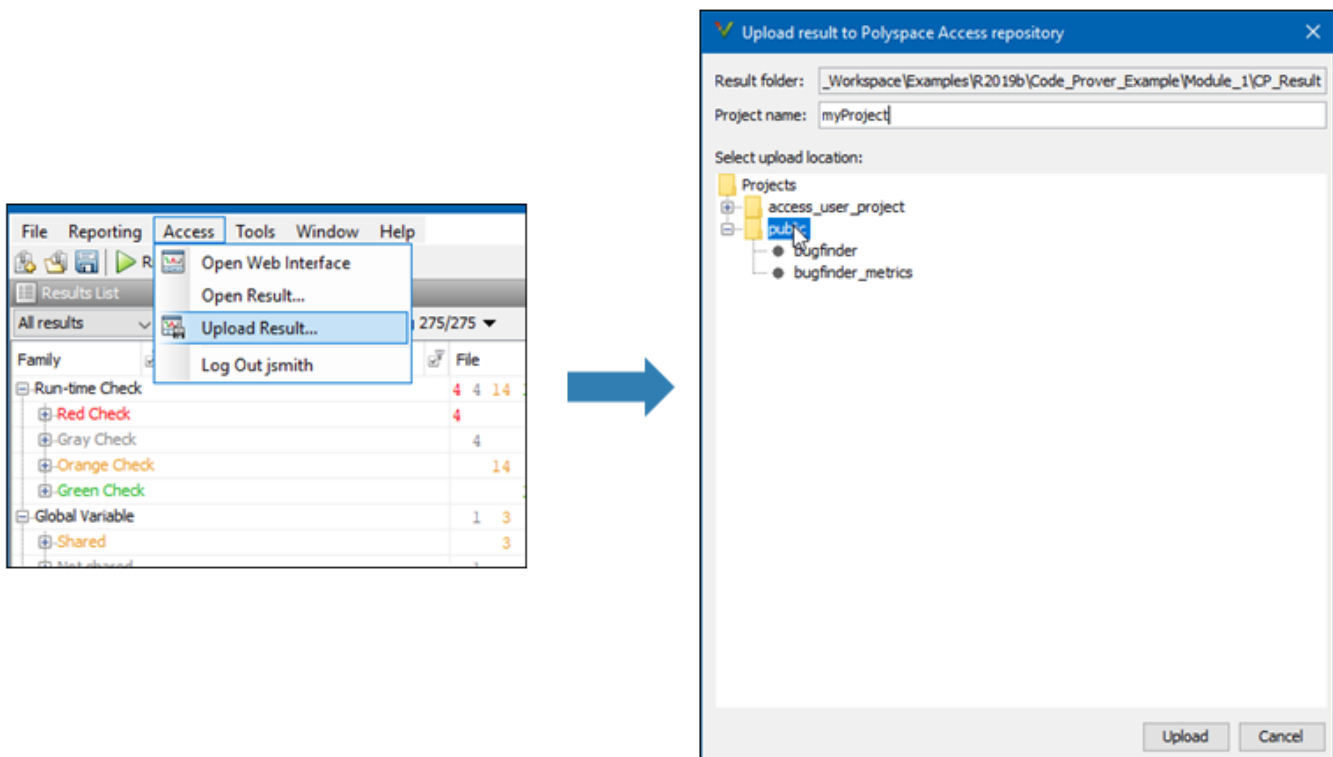
You can also upload results to Polyspace Access by selecting a result in the **Project Browser** pane and using the context menu.

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. To create custom statuses, see "Open Polyspace Access Results in a Desktop Interface" on page 3-5.

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. To upload results to a different folder, use the `-parent-folder` option. *hostName* and *port* correspond to the host name and port number you specify in the URL of the Polyspace Access interface, for example `https://hostName:port/metrics/index.html`. If you are unsure about which host name and port number to use, contact your Polyspace Access administrator. Depending on your configuration, you might also have to specify the `-protocol` option in the command. See .

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

Results Upload Compatibility and Permissions

Results Compatibility

You cannot upload analysis results to a Polyspace Access version that is older than the version of the Polyspace product that generated the results. For instance, you cannot upload results generated with a Polyspace product version R2019b to a Polyspace Access version R2019a.

If you upload results generated with a Polyspace product version R2018b or earlier, you cannot view these results in the Polyspace Access **REVIEW** perspective. To review R2018b or earlier results that you uploaded to Polyspace Access, see "Open Polyspace Access Results in a Desktop Interface" on page 3-5.

User Permissions for Uploaded Results

You are the project **Owner** for all the results that you upload. The project **Owner** or an **Administrator** must add other users as **Contributor** to grant them permission to see those results, unless you upload the results to a folder that other users already have permission to see.

Results that you upload to the **public** folder are visible to all Polyspace Access users. For more information, see "Manage Project Permissions" on page 3-18.

See Also

More About

- "Register Polyspace Desktop User Interface" on page 1-36
- "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 1-36.

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. To assign a custom **Status**, in the desktop interface:

- Go to **Tools** > **Preferences** and select the **Review Statuses** tab to create a custom status.
- In the **Result Details** pane, assign the status you created from the **Status** drop-down.

In the Polyspace Access interface, a custom status you assign in a project is not available in other projects.

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 and *port* correspond to the host name and port number you specify in the URL of the Polyspace Access interface, for example `https://hostName:port/metrics/index.html`. If you are unsure about which host name and port number to use, contact your Polyspace Access administrator. Depending on your configuration, you might also have to specify the `-protocol` option in the command. See .

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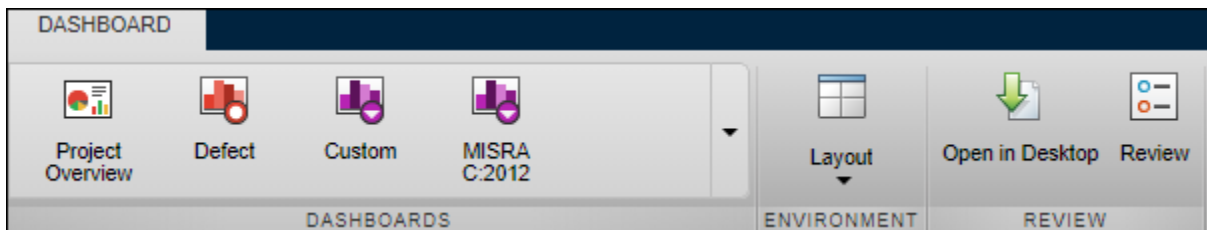
- Click a button in the **DASHBOARDS** section of the toolstrip to open the corresponding dashboard for the selected folder or project. Except for **Project Overview** and **Quality Objectives**, each dashboard shows information for a single family of findings.
- The **Open in Desktop** and **Review**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

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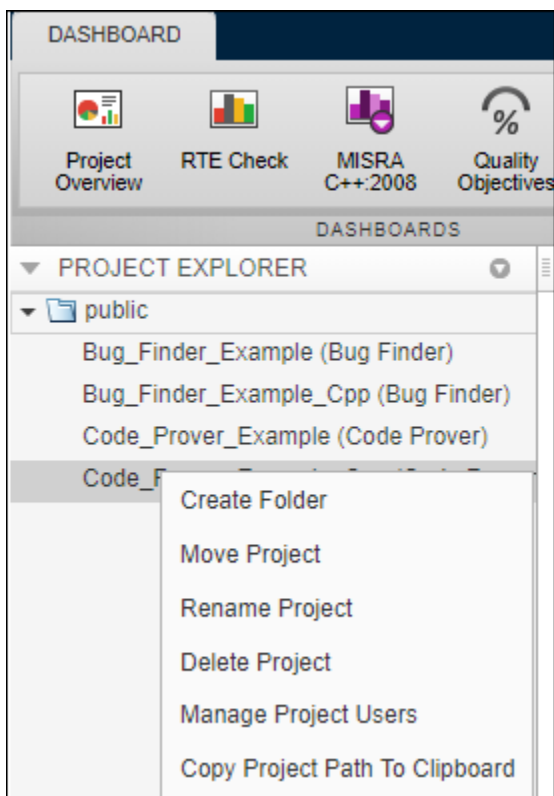
- View all projects and folders for which you are an **Administrator**, **Owner** or **Contributor**. All users are contributors to the **Public** folder

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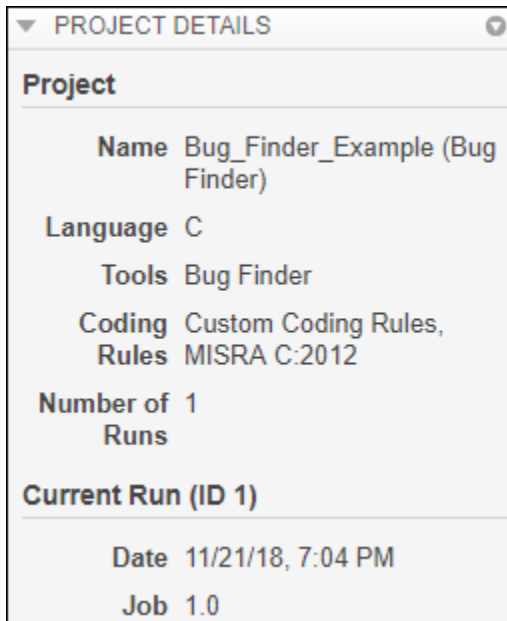
- View additional details about the selected folder or project. You can view information about which language and coding standards were enabled in the analysis configuration.

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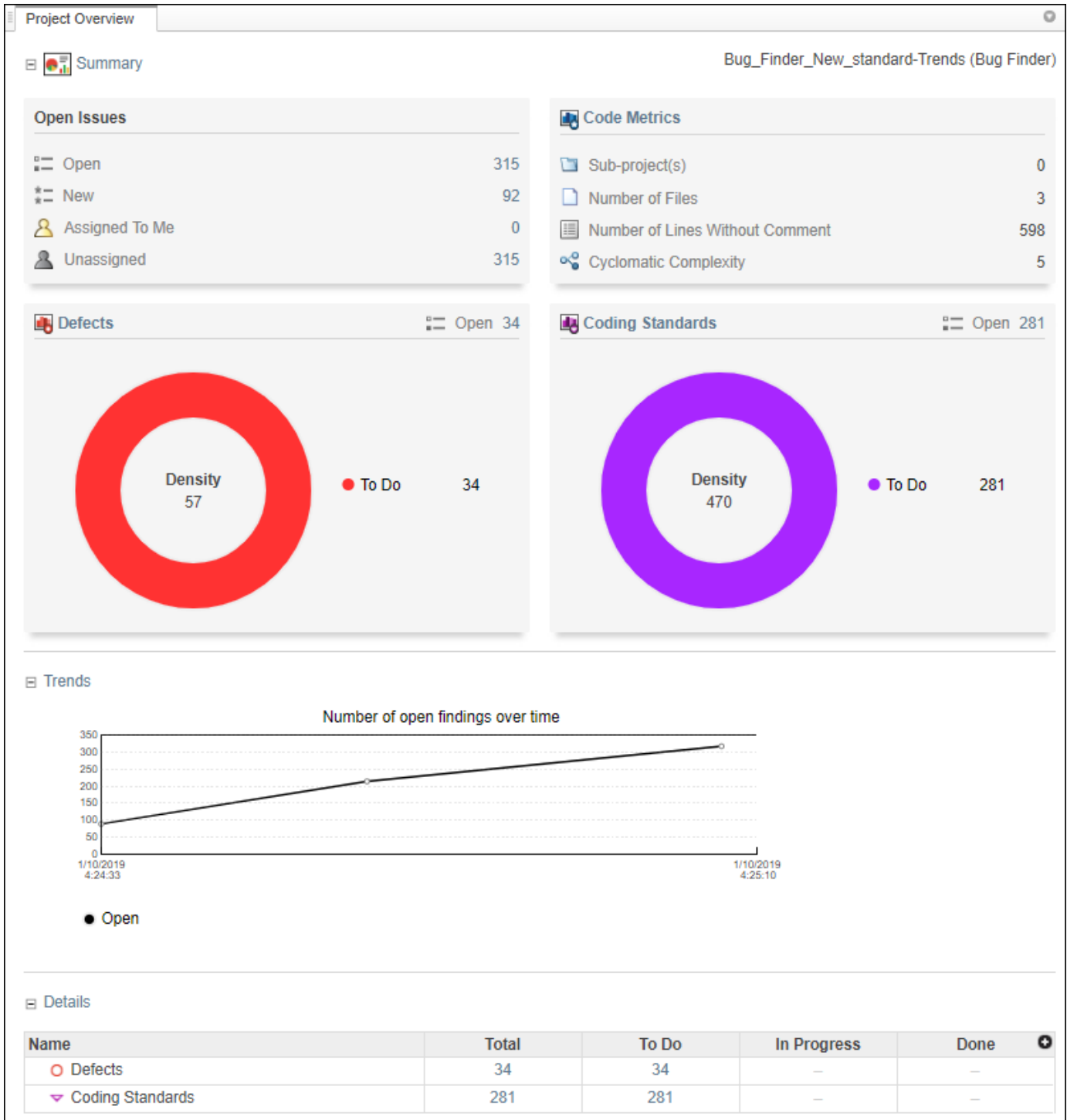
This dashboard gives you a snapshot of all the findings available for the selected folder or project. If you select a folder that includes multiple projects, the dashboard displays an aggregate of results for all the projects. The dashboard contains three collapsible sections:

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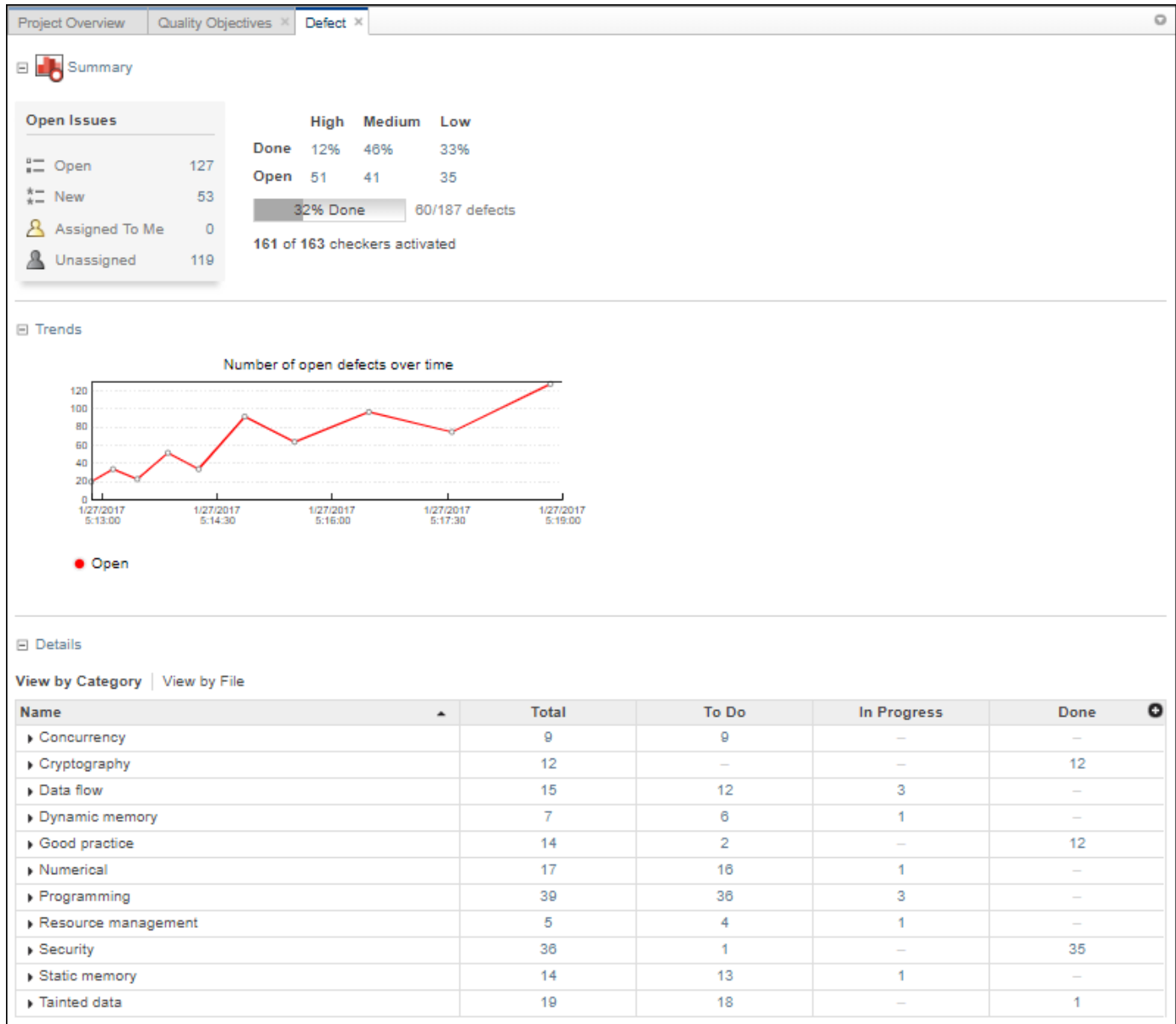
These dashboards give you a more in-depth overview for a family of findings. If you select a folder that includes multiple projects, the dashboards display an aggregate of results for all the projects. The dashboards contain three collapsible sections:

- **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 chard 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
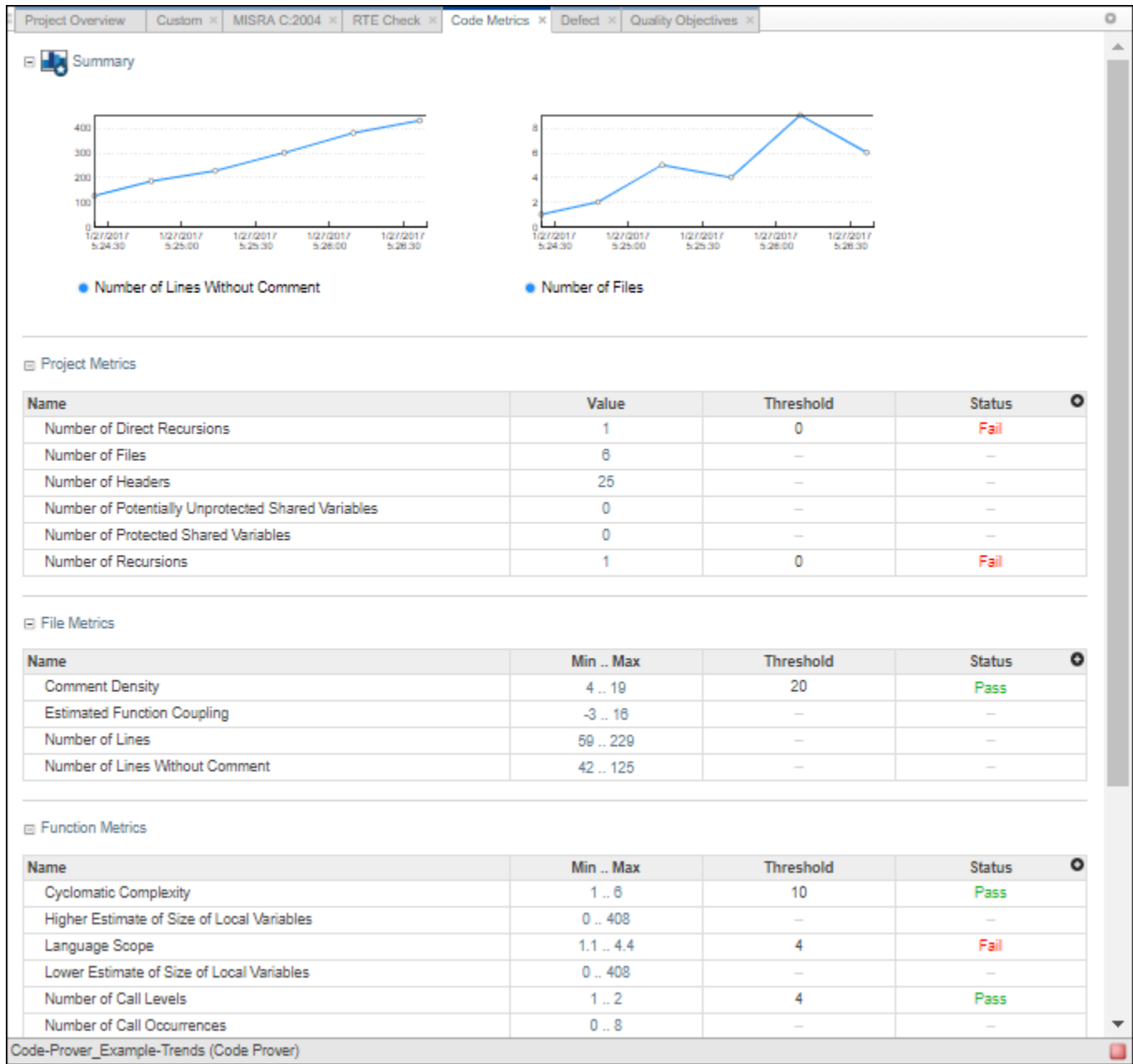
This dashboard contains four collapsible sections.

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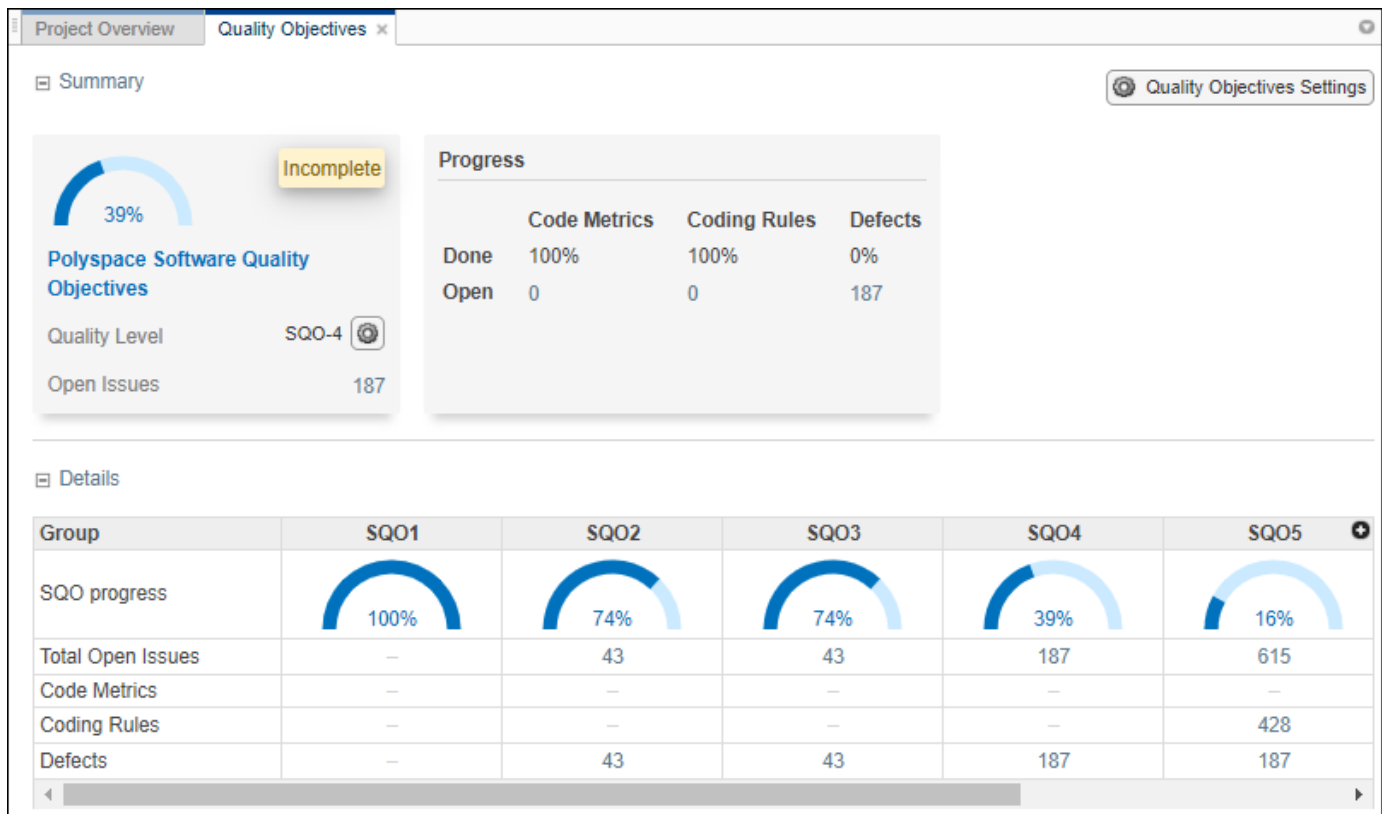
This dashboard displays a summary of the quality of your code against the threshold selected from the dropdown menu. The dashboard also shows a table with details of code quality for all quality objective thresholds.

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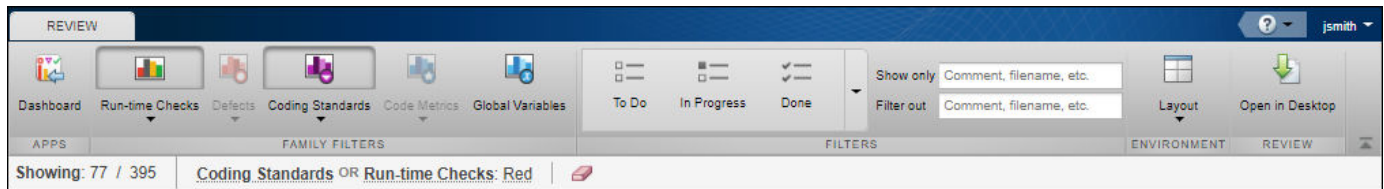
- Click a button in the **FAMILY FILTERS** section of the toolstrip to see the corresponding family of findings, or a subset of those findings. The filters bar underneath shows how many findings are displayed out of the total findings, along with which filters are currently applied.

  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.
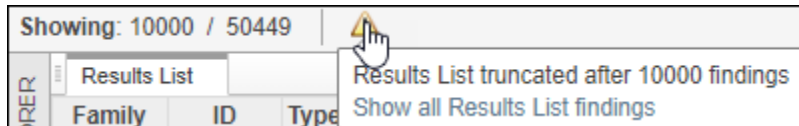
  The 10000 findings limit is preset and cannot be changed.
- The buttons in the **FILTERS** section of the toolstrip are global. They apply to all families of findings.

  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**

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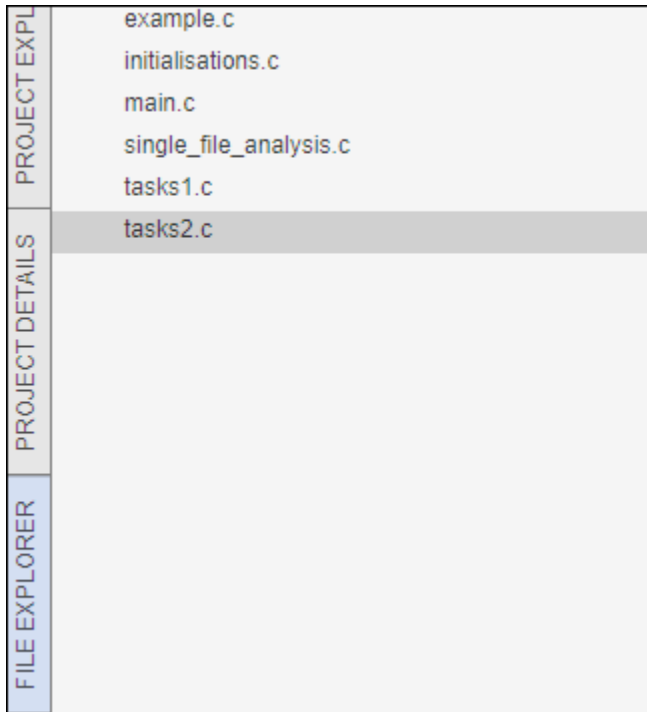
Use the file explorer to show findings by file in the **Results List** pane.

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**.
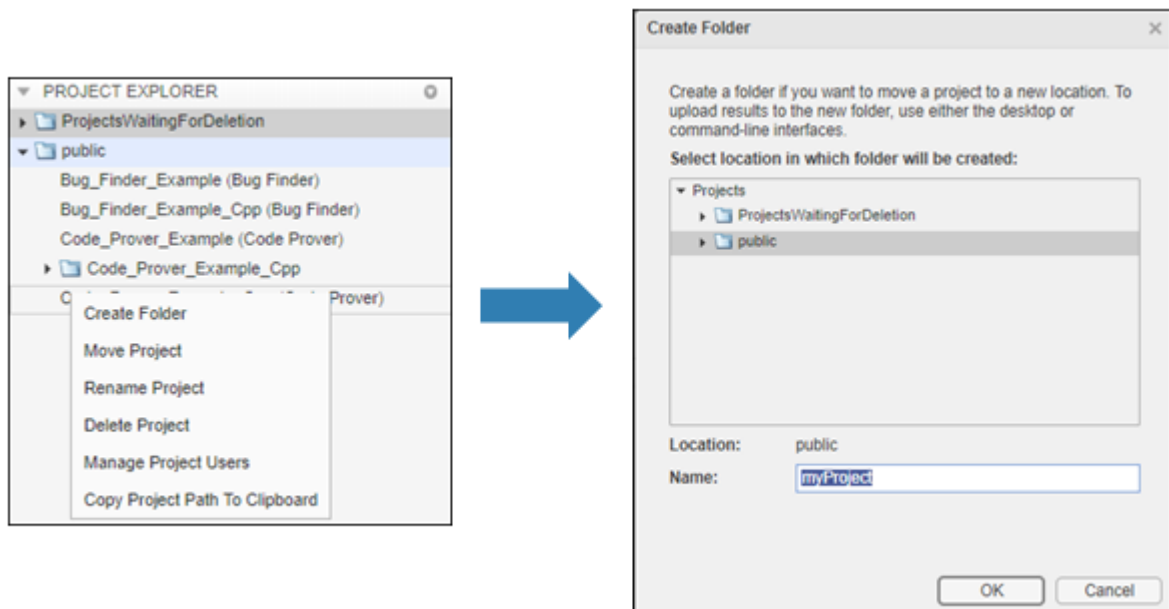
Create Project Folder at Command Line

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 Bug Finder or a Polyspace Bug Finder Server installation. *polyspaceroot* is the Polyspace product installation folder, for example `C:\Program Files\Polyspace Server\2019a`.

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

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

hostName and *port* correspond to the host name and port number you specify in the URL of the Polyspace Access interface, for example `https://hostName:port/metrics/index.html`. If you are unsure about which host name and port number to use, contact your Polyspace Access administrator. Depending on your configuration, you might also need to specify the `-protocol` option 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 1-41. 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 3-18 or "Configure Polyspace Access App Services" 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.

By default, all users have **Contributor** privileges for the **public** folder.

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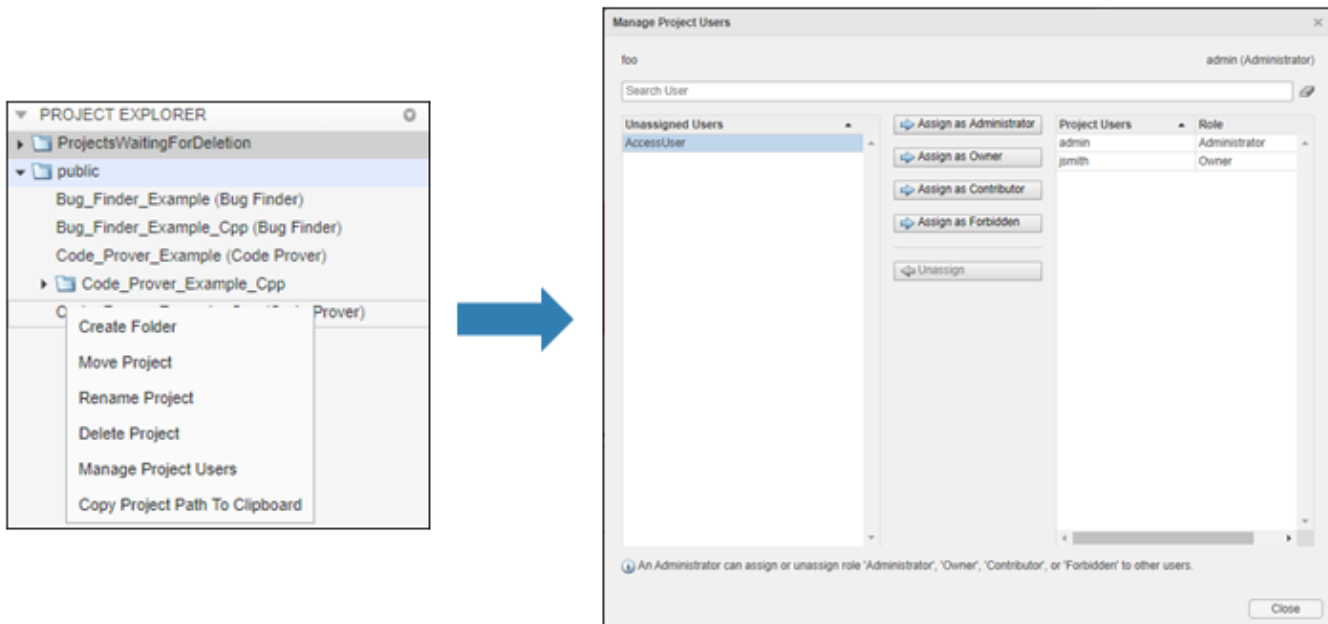
The **Assign as Administrator** button is visible only for users with role **Administrator**. You must assign at least one user as administrator in the cluster operator settings on page 1-30 before you can manage administrators from the web interface.

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 Bug Finder or a Polyspace Bug Finder 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 and *port* correspond to the host name and port number you specify in the URL of the Polyspace Access interface, for example `https://hostName:port/metrics/index.html`. If you are unsure about which host name and port number to use, contact your Polyspace Access administrator. Depending on your configuration, you might also need to specify the `-protocol` option 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
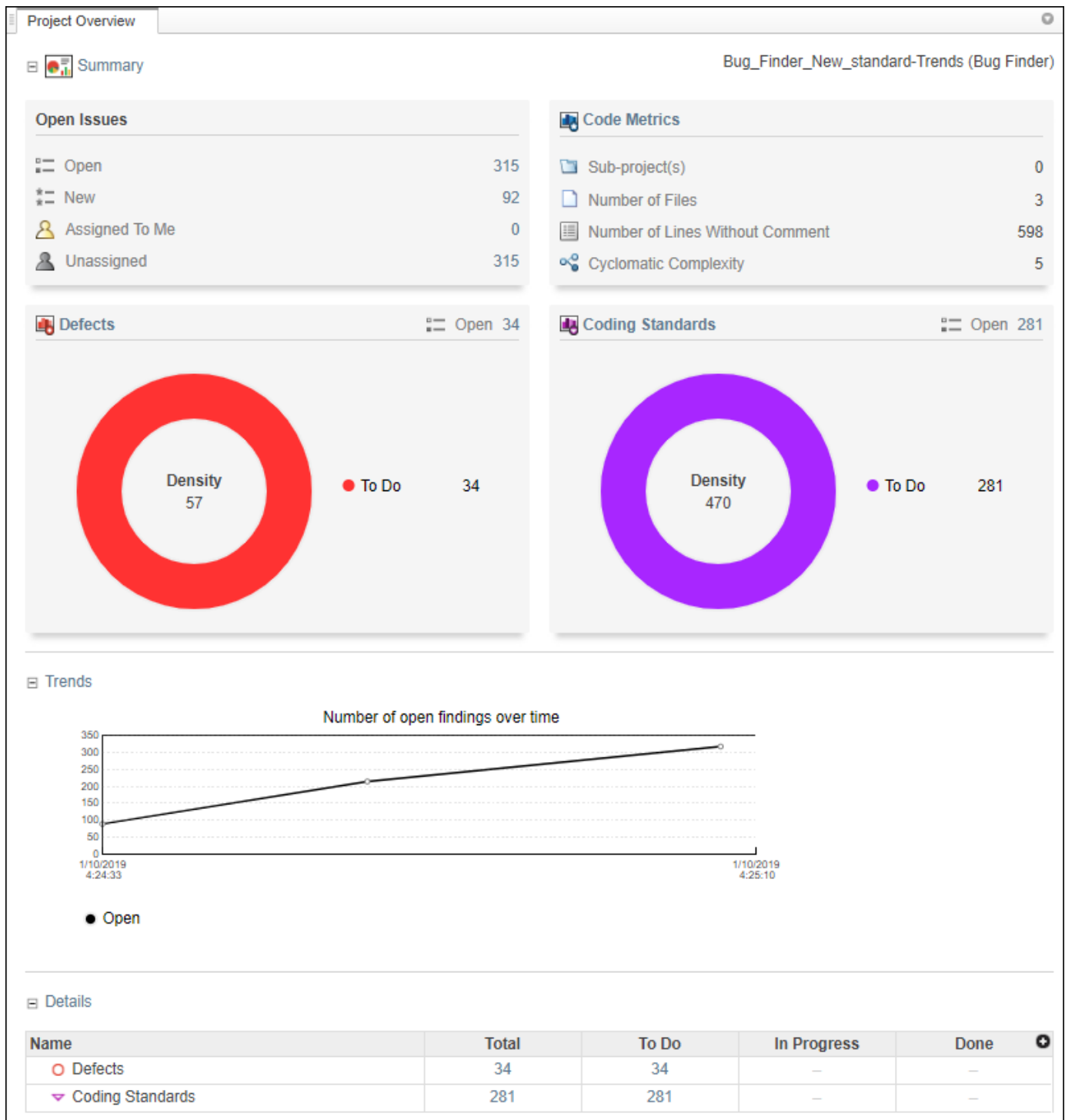
In the **DASHBOARD** perspective, select the project that you want to investigate from the **PROJECT BROWSER**.

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 3-22.

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 3-17.

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**.

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 3-5.

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.

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 Bug Finder or a Polyspace Bug Finder 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_RLDatas\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
`-max-project-runs` *int*	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.
`-project-date-after` *YYYY[-MM[-DD]]*	Only migrate results that were uploaded to Polyspace Metrics on or after the specified date.
`-product` *productName*	Product used to analyze and produce project findings, specified as `bug-finder`, `code-prover`, or `polyspace-ada`.
`-analysis-mode` *mode*	Analysis mode used to generate project findings, specified as `integration` or `unit-by-unit`.

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_RLDatas\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_RLDatas\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 -port port  ^
-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 and *port* correspond to the host name and port number you specify in the URL of the Polyspace Access interface, for example `https://hostName:port/metrics/index.html`. If you are unsure about which host name and port number to use, contact your Polyspace Access administrator. Depending on your configuration, you might also need to specify the `-protocol` option 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" (Polyspace Code Prover Access) (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 1-36
- "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 "Choose Between Polyspace Bug Finder and Polyspace Code Prover" (Polyspace Bug Finder Server).

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.
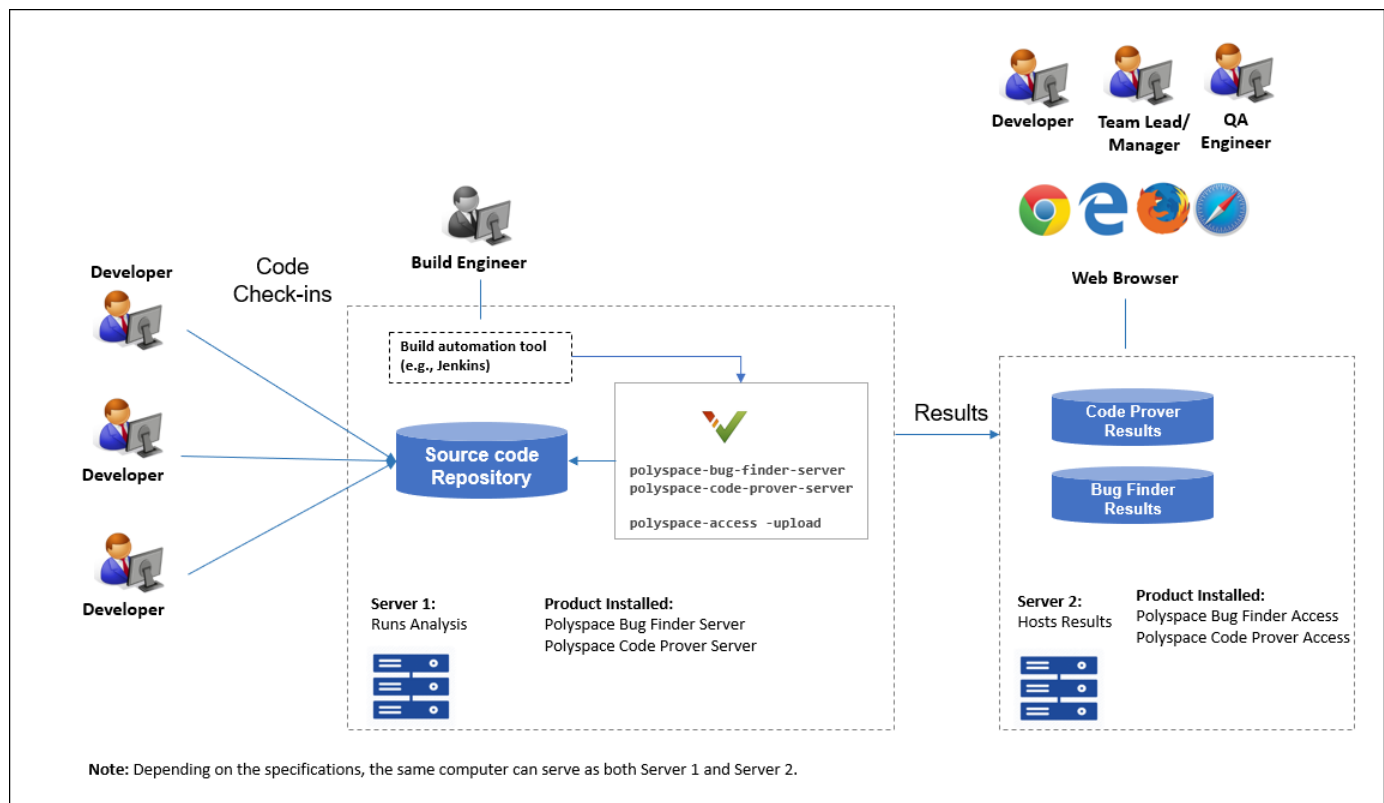
Installation

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 Bug Finder 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 user manager to authenticate user logins, an issue tracker to integrate your bug tracking tool with Polyspace, a database to manage results, 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 Admin.

To install Polyspace Access:

- Download the installer as a zip file.
- Unzip the file and start the Cluster Admin. From the Cluster Admin 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`

Install Network License Manager

Both Polyspace Server and Polyspace Access use licenses that require communication with a network license manager for license checkouts.

- To install, configure and start the network license manager for Polyspace Server, see "Administer Network Licenses".
- To install, configure and start the network license manager for Polyspace Access, see "Manage Polyspace Access License".

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 Bug Finder 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/R2020b`, see also "Installation Folder" (Polyspace Bug Finder Server)). To start running Polyspace Server by using sample C source files and sample scripts, see:

- "Run Polyspace Bug Finder on Server and Upload Results to Web Interface" (Polyspace Bug Finder Server)
- "Send Email Notifications with Polyspace Bug Finder Results" (Polyspace Bug Finder 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 Bug Finder 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 Bug Finder 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 Bug Finder 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 Bug Finder 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.