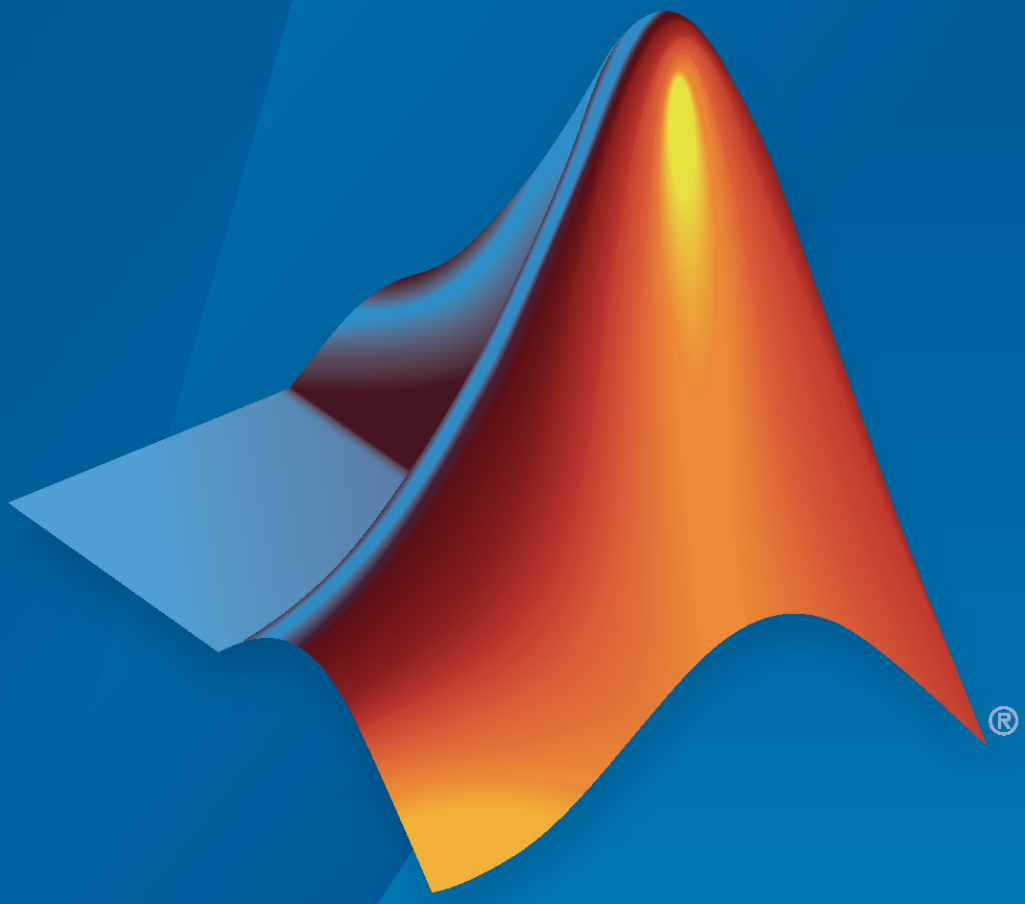
Polyspace® Code Prover™ Access™ Release Notes



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

Polyspace® Code Prover™ Access™ Release Notes

© COPYRIGHT 2019–2021 by The MathWorks, Inc.

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

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

Trademarks

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

Patents

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

Contents

R2021a

Dashboard and Review in Web Browser **1-2**

Results Review Scope: Define and share custom families of filters **1-2**
Results Review Layout: Select view to prioritize review of code or results list **1-2**
Code Quality Comparison Between Runs: Filter and view information for previous findings fixed in the current run **1-4**
Code Prover Result Messages: Redundant tooltips removed from += and similar operations **1-4**

Installation **1-6**

License Management: Uploading of results to Polyspace Access no longer requires a license checkout **1-6**
User Manager: Enable pagination when requesting large set of users from LDAP server **1-6**
Bug Tracking Tool: Create Jira tickets for Jira projects that use single select custom fields **1-6**
Admin Interface: Improved logging for Polyspace Access services **1-6**

R2020b

Dashboard and Review in Web Browser **2-2**

Code Quality Improvement Progress: Compare results from current run to previous runs and determine progress in code quality improvement . . . **2-2**
Code Quality Objectives: Define custom quality objectives definitions and apply them to specific projects **2-2**
Project Selection: Find a project in the Project Explorer through a text filter **2-3**

Installation **2-4**

Bug Tracking Tool: Integrate with Jira Software Cloud **2-4**
Cluster Admin Settings: Validate values of settings on demand or on save **2-4**
HTTPS Configuration: Configure services without specifying ports or SSL certificates **2-4**
Functionality Replaced: Polyspace Access embedded LDAP **2-4**
Changes in Polyspace Access docker containers, options, and binaries . . . **2-5**

R2020a

Dashboard and Review in Web Browser **3-2**

AUTOSAR Support: Navigate from Polyspace findings to AUTOSAR ARXML specifications **3-2**
Bug Tracking Tool Support: Create Redmine tickets for Polyspace Access results and assign to developers **3-2**
Simulink Support: Navigate from generated code in Polyspace Access to blocks in model **3-3**
Results Review: See review history of findings **3-4**
Results Review: See the configuration options used for analysis **3-5**
Code Quality Objectives: Customize thresholds used to track the quality of your code **3-7**
Project Dashboard: Open results by clicking Dashboard charts **3-7**
Bug Tracking Tool Support: Manage tickets for multiple findings **3-8**
Results Review: View error call graph **3-8**
Results Review: View variable access graph **3-8**

Installation **3-9**

Installation and Configuration: Issue Tracker service **3-9**
Installation and Configuration: Change in default location of Polyspace Access data volume and working directories **3-9**

R2019b

Installation **4-2**

User Authentication: Use LDAP search filters to restrict number of users to authenticate **4-2**
User Management: Update list of users from LDAP database or LDIF file **4-2**

R2019a

Dashboard and Review in Web Browser **5-2**

Project Dashboard: Track progress of code quality via Polyspace results **5-2**
Project Dashboard: Compare Polyspace Code Prover results against Software Quality Objectives **5-4**
Collaborative Review Support: Review Polyspace Code Prover results and source code in web browser **5-5**
Collaborative Review Support: Share Polyspace Code Prover results using web links **5-6**

Project Authorization Management: Create and enforce authorization policies for access to project **5-6**
Bug Tracking Tool Support: Create JIRA issues for Polyspace Code Prover results and assign to developer **5-7**

R2021a

Version: 2.4

New Features

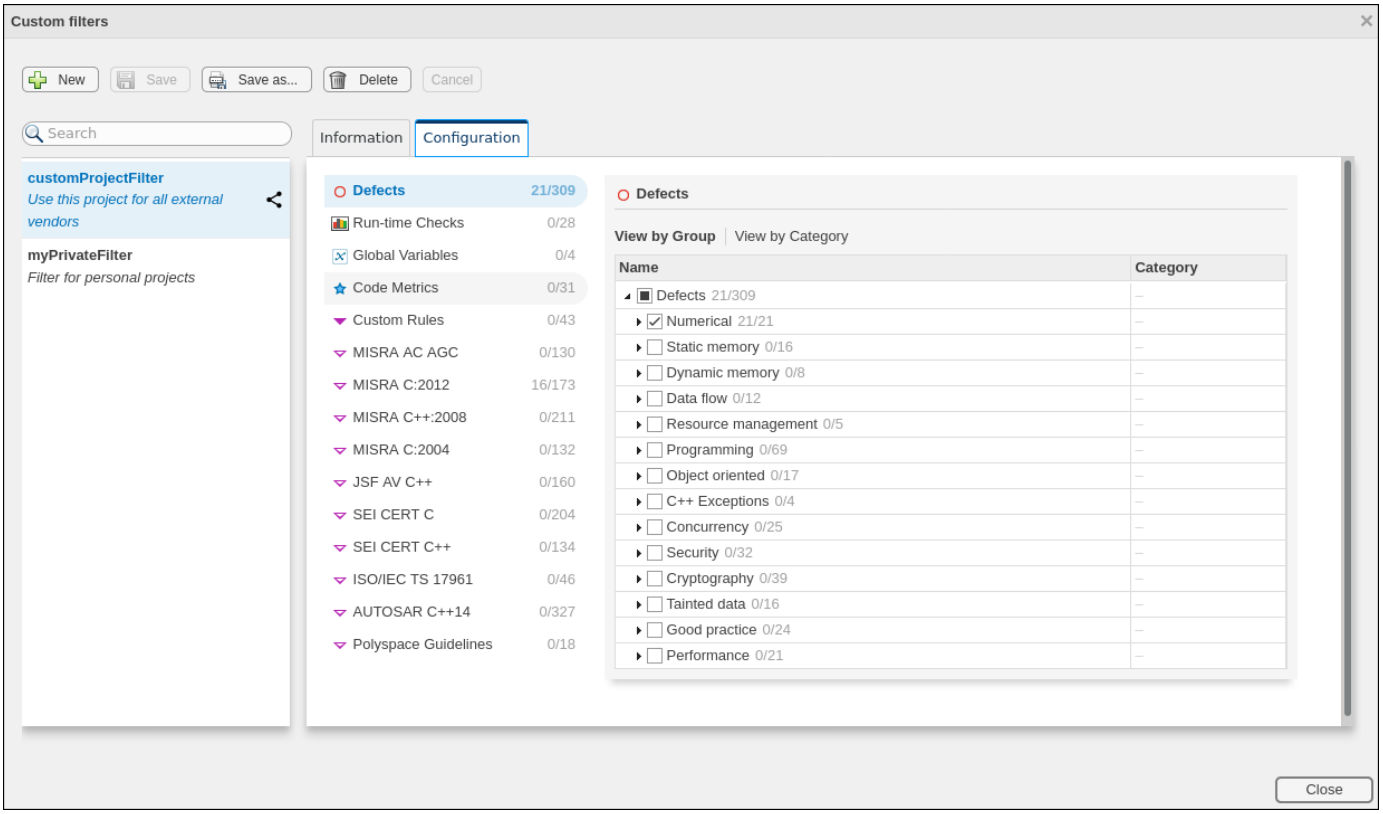
Bug Fixes

Dashboard and Review in Web Browser

Results Review Scope: Define and share custom families of filters

In R2021a, you can create custom families of filters to tailor the scope of your review to results that are relevant to only your project or organization. You can then share the customized review scopes with other Polyspace® Access users. See “Create Custom Filter Groups in Polyspace Access Web Interface”.

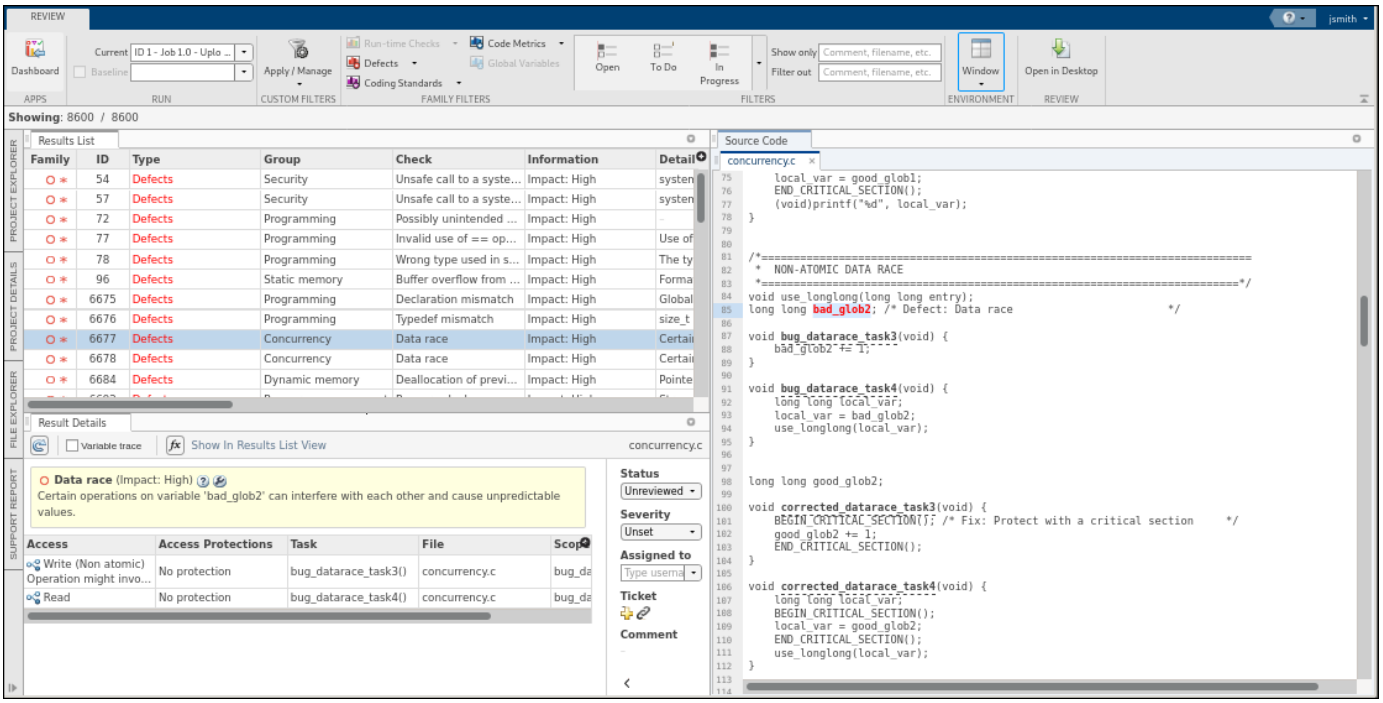
For example, you might want to review your code for violations of a subset of only **Numerical** defects and **MISRA C®: 2012** rules.



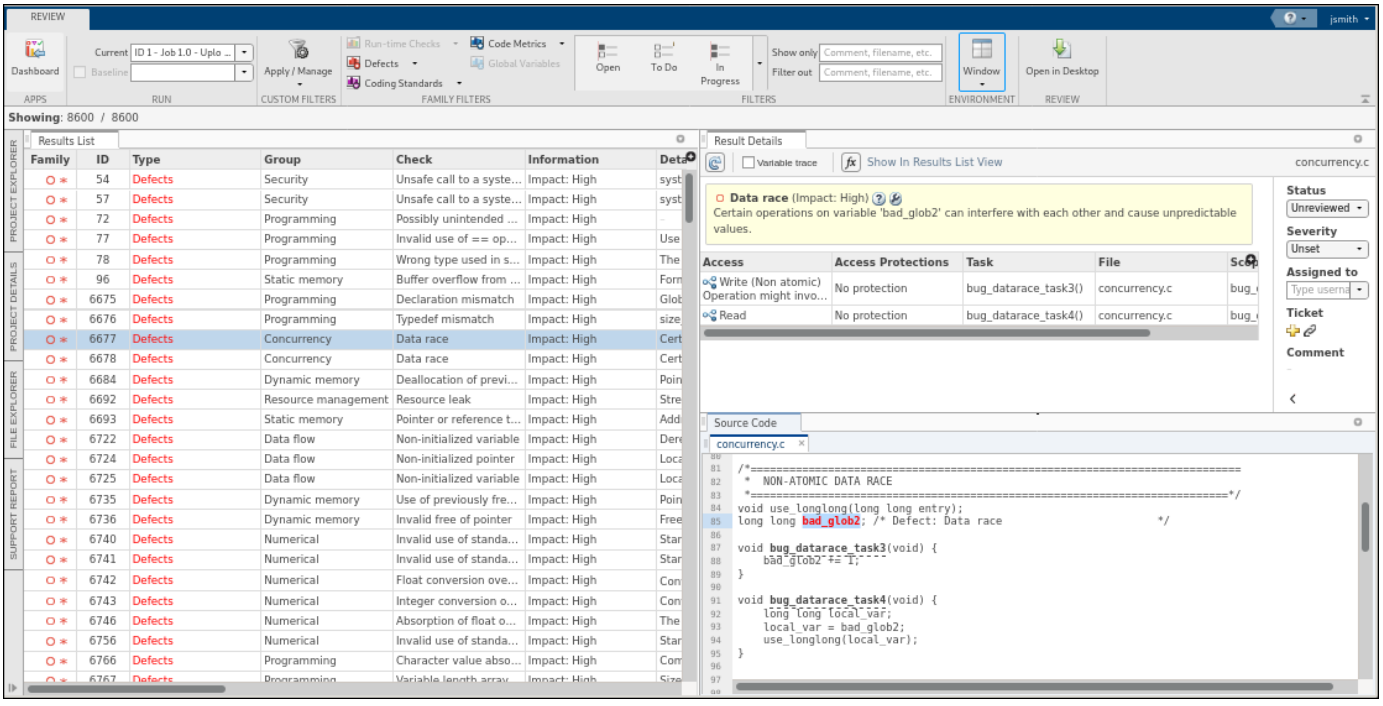
Results Review Layout: Select view to prioritize review of code or results list

In R2021a, the Polyspace Access interface has different layouts to match your results review workflow.

The default **Code Review** layout enables you to focus on the **Source Code** while you investigate issues in your code.



The **Results Review** layout prioritizes the **Results List** and **Result Details** panes as you review and triage findings.

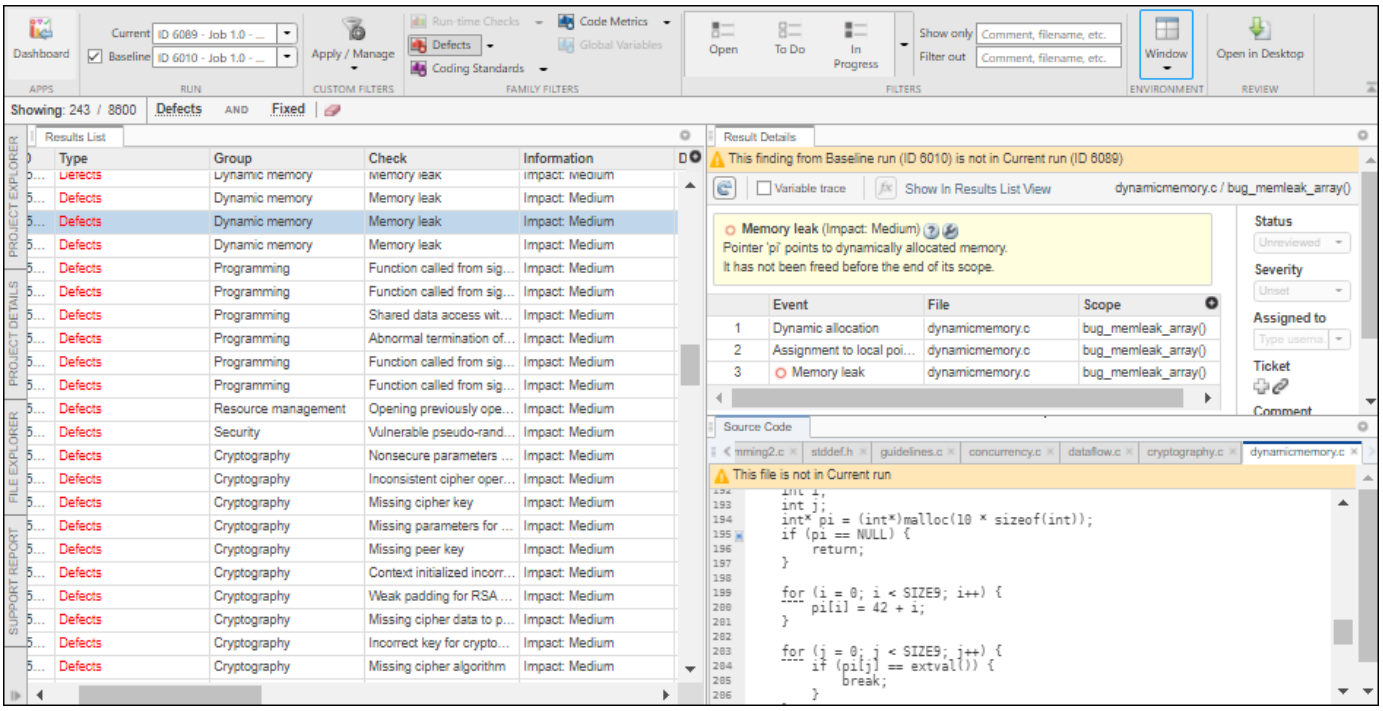


Code Quality Comparison Between Runs: Filter and view information for previous findings fixed in the current run

In R2021a, if you compare two project runs and some of the findings from the **Baseline** run are **Fixed** in the **Current** run, you can filter for and view the source code and result details for these findings. See “Compare Analysis Results to Previous Runs”.

Polyspace Access considers a finding **Fixed** if either:

- You make changes to your code that fix an issue.
- The source code that contains an issue is deleted or is not part of the current analysis.



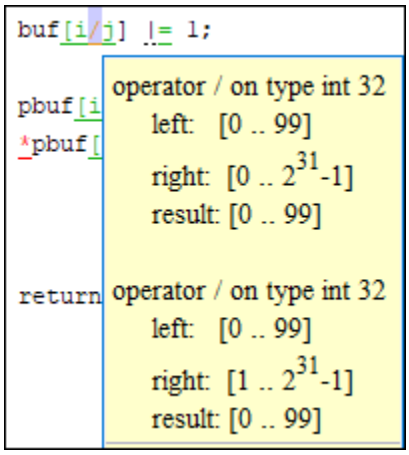
Previously, you had to open the **Baseline** run as a separate tab to view the source code and result details for **Fixed** findings.

Code Prover Result Messages: Redundant tooltips removed from += and similar operations

Summary: In R2021a, in code operations that use the shorthand form `x+=y`, `x-=y`, `x*=y`, and so on, if `x` contains an operation that triggers an orange check, you see only this orange check and the associated tooltip in the Code Prover results.

Benefits: Previously, you saw a green check in addition to an orange check because the operation in `x` happens twice, once on the left and once on the right side of the assignment. The orange check eliminated a problematic path and resulted in a subsequent green check. You also saw two copies of essentially the same tooltip. Starting in R2021a, these redundant tooltips have been removed.

For instance, this division operation previously triggered two checks and showed two copies of essentially the same range information in the tooltip. Starting in R2021a, you see only the upper copy, which represents range information before either of the two divisions are performed.



Installation

License Management: Uploading of results to Polyspace Access no longer requires a license checkout

In R2021a, the upload of analysis results to the Polyspace Access database does not trigger a Polyspace Access license checkout.

If you upload results as part of an automation script, you no longer consume a license when you run the script. Previously, each results upload triggered a license checkout.

User Manager: Enable pagination when requesting large set of users from LDAP server

In R2021a, if you use an LDAP server to retrieve user profiles and authenticate user logins, you can enable pagination to retrieve a large set of users from the LDAP server. See “Authenticate Users from Your Organization LDAP Server”.

Typically, LDAP servers limit the number of entries that they return in a result set. If the number of entries exceed that limit, the result set is truncated. When you enable pagination, the number of results is broken up into smaller sets. You are able to retrieve all entries from the LDAP server when you query a large set of users.

Bug Tracking Tool: Create Jira tickets for Jira projects that use single select custom fields

In R2021a, if you integrate the Jira software bug tracking tool (BTT) with Polyspace Access, you can create Jira tickets for Jira projects that are configured with single select custom fields. See “Configure Jira Software Bug Tracking Tool”.

Previously, Polyspace Access did not support the creation of Jira tickets in projects that used single select custom fields.

Admin Interface: Improved logging for Polyspace Access services

In R2021a, when you view the logs for the Polyspace Access services in the **Admin** user interface, the logs are automatically refreshed. You do not need to reload the page to view new events.

R2020b

Version: 2.3

New Features

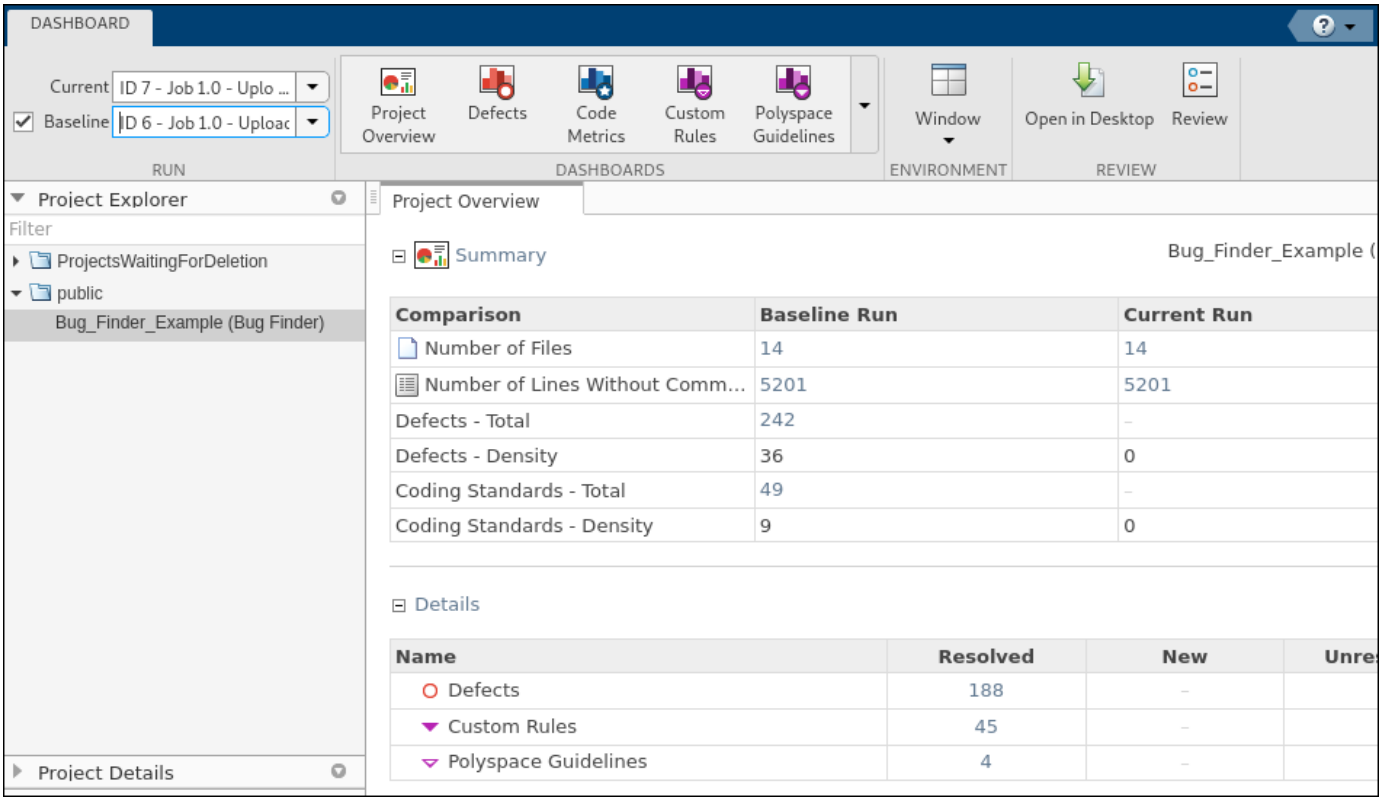
Bug Fixes

Compatibility Considerations

Dashboard and Review in Web Browser

Code Quality Improvement Progress: Compare results from current run to previous runs and determine progress in code quality improvement

In R2020b, you can select any two runs of a project in the Polyspace web interface (current and baseline runs) and compare them. You can compare a current run to only older baseline runs.

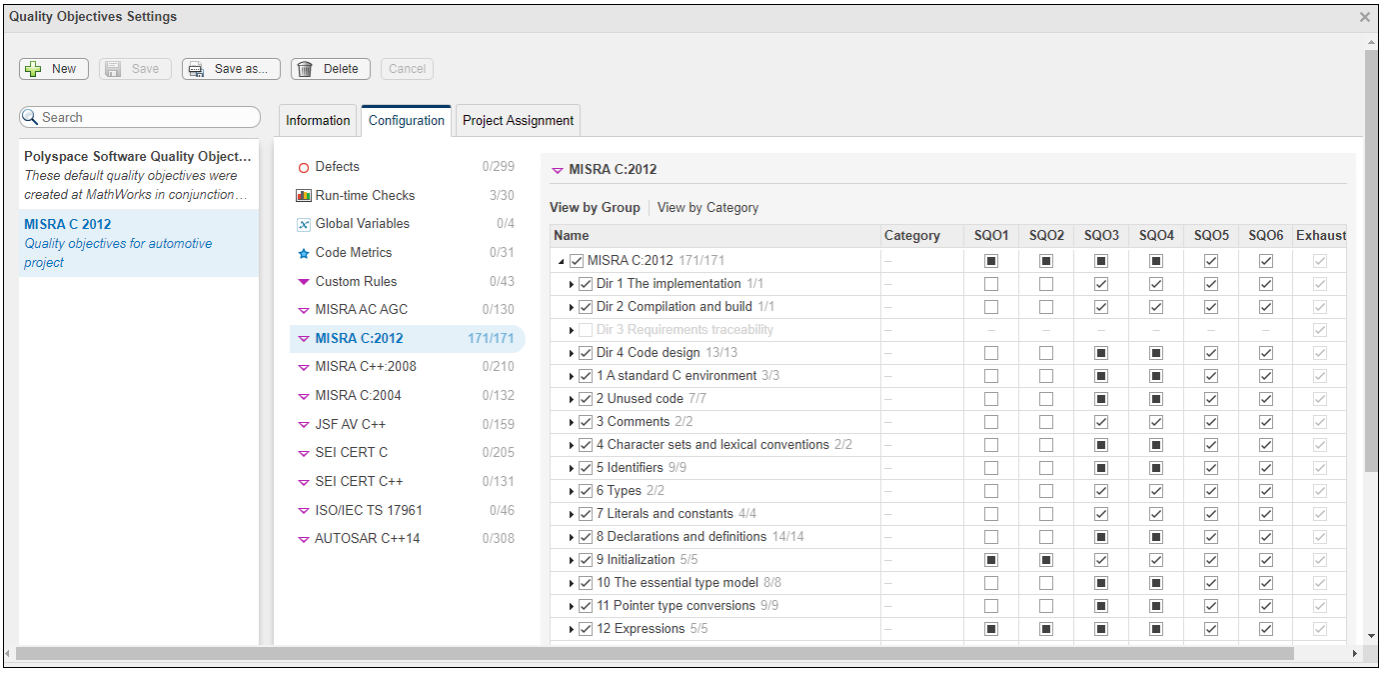


The comparison shows the number of analysis findings that are:

- **Resolved**. Findings from the baseline run no longer found in the current run.
- **New**. Findings in the current run that were not present in the baseline run.
- **Unresolved**. Findings from the baseline run that are still present in the current run.

Code Quality Objectives: Define custom quality objectives definitions and apply them to specific projects

In R2020b, you can create custom quality objectives definitions and apply those definitions to specific projects. For instance, if you want to track the compliance of a project with a coding standard, you can create Quality Objective thresholds for that coding standard and apply them to your project.

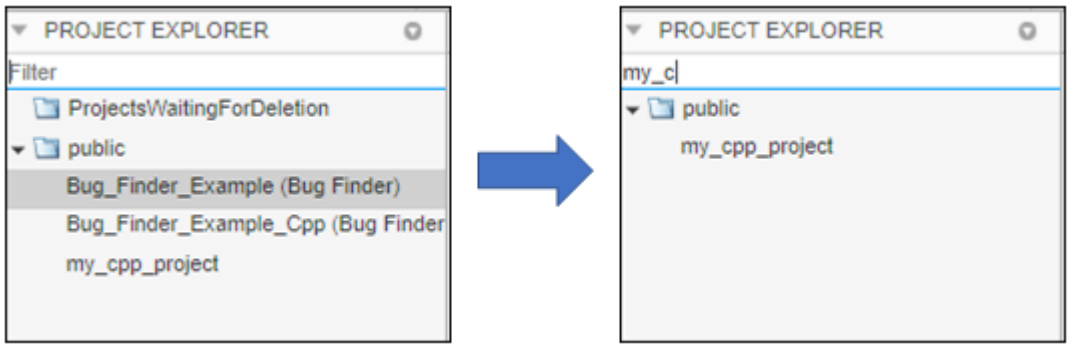


To create custom quality objectives definitions, you must be an **Administrator** or **Owner**.

Previously, custom quality objectives applied to all projects.

Project Selection: Find a project in the Project Explorer through a text filter

In R2020b, you can use a text filter in the **Project Explorer** to find projects that are not visible in a folder hierarchy. The text filter is not case sensitive.



Installation

Bug Tracking Tool: Integrate with Jira Software Cloud

In R2020b, you can integrate Jira Software Cloud with Polyspace Access. After you configure Polyspace Access, you can create a Jira ticket to track Polyspace findings. The ticket is populated with details of the finding and a link to open that finding in Polyspace Access. See (Polyspace Bug Finder Access)Configure Issue Tracker (Polyspace Code Prover Access).

Previously, you could integrate Polyspace Access with only self-managed Jira Software.

Cluster Admin Settings: Validate values of settings on demand or on save

In R2020b, the **Cluster Admin** validates the settings that you enter in the **Cluster Settings** when you save those settings. You can also validate the settings before you save by clicking **Validate now** at the bottom of the page.

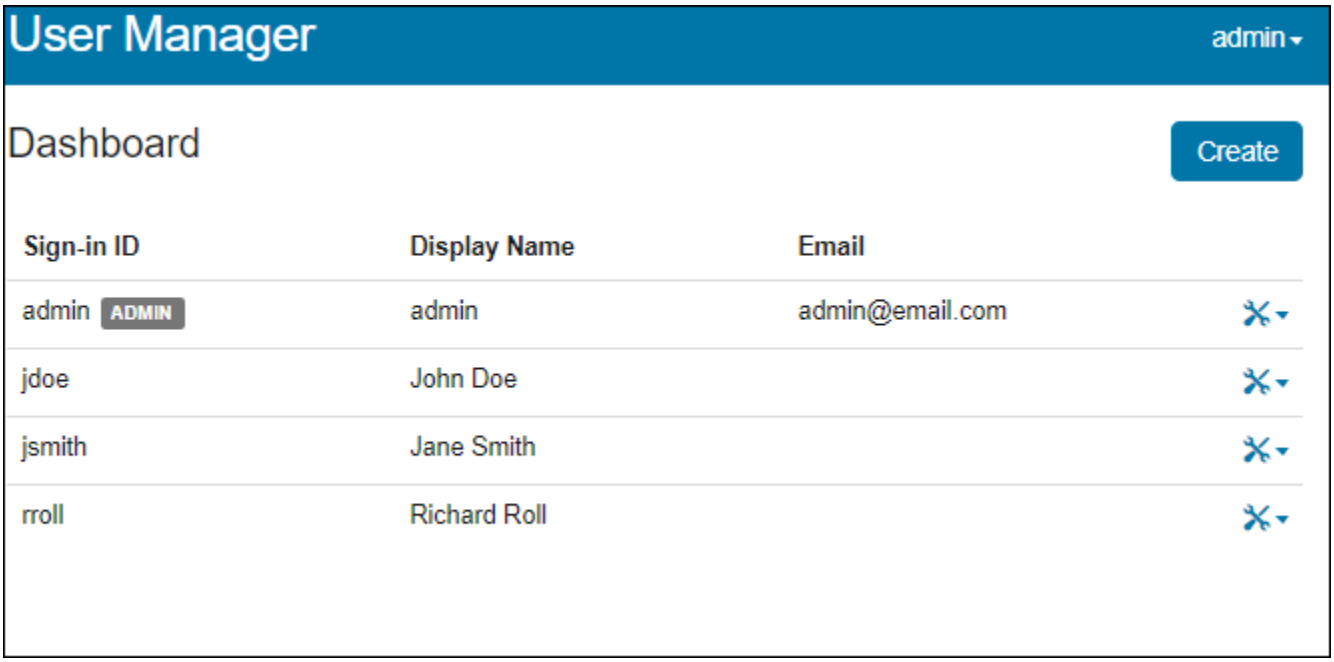
HTTPS Configuration: Configure services without specifying ports or SSL certificates

In R2020b, if you install Polyspace Access on a single node, the ports of the Polyspace Access services are no longer exposed. You do not need to specify port numbers for the services or to provide SSL private keys and certificates for the HTTPS configuration. See (Polyspace Bug Finder Access)Configure Polyspace Access for HTTPS (Polyspace Code Prover Access)

Previously, you had to check the availability of the ports for the services, and then you provided a private key and SSL certificate file to enable the HTTPS protocol for Polyspace Access.

Functionality Replaced: Polyspace Access embedded LDAP

The Polyspace Access embedded LDAP is removed in R2020b. To continue using custom login credentials for Polyspace Access, use the **User Manager** internal directory instead. See Authenticate Users from Internal Directory (Polyspace Code Prover Access) (Polyspace Bug Finder Access).



Compatibility Considerations

In the **User Manager** interface, create users to transfer the user names and passwords that you stored in the embedded LDAP LDIF file to the **User Manager** database.

Changes in Polyspace Access docker containers, options, and binaries

In R2020b, the following docker containers, options, and binaries have been renamed:

- The `cop-docker-agent` binary is now called the `admin-docker-agent`
- **HTTPS Options**

Previous Option Name	Current Option Name
`--https-certificate-file`	`--ssl-cert-file`
`--https-private-key-file`	`--ssl-key-file`
`--https-trusted-certificates-file`	`--ssl-ca-file`

- **Containers**

Previous Container Name	Current Container Name
`polyspace-db`	`polyspace-access-db-main`
`polyspace-etl`	`polyspace-access-etl-main`
`polyspace-gateway`	`gateway`
`polyspace-issuetracker`	`issuetracker-server-main`
`polyspace-web-server`	`polyspace-access-web-server-main`

Compatibility Considerations

In your scripts, replace instances of the previous names with the current names. You cannot reuse a settings configuration file (`settings.json`) from a previous release of Polyspace Access with the R2020b software.

R2020a

Version: 2.2

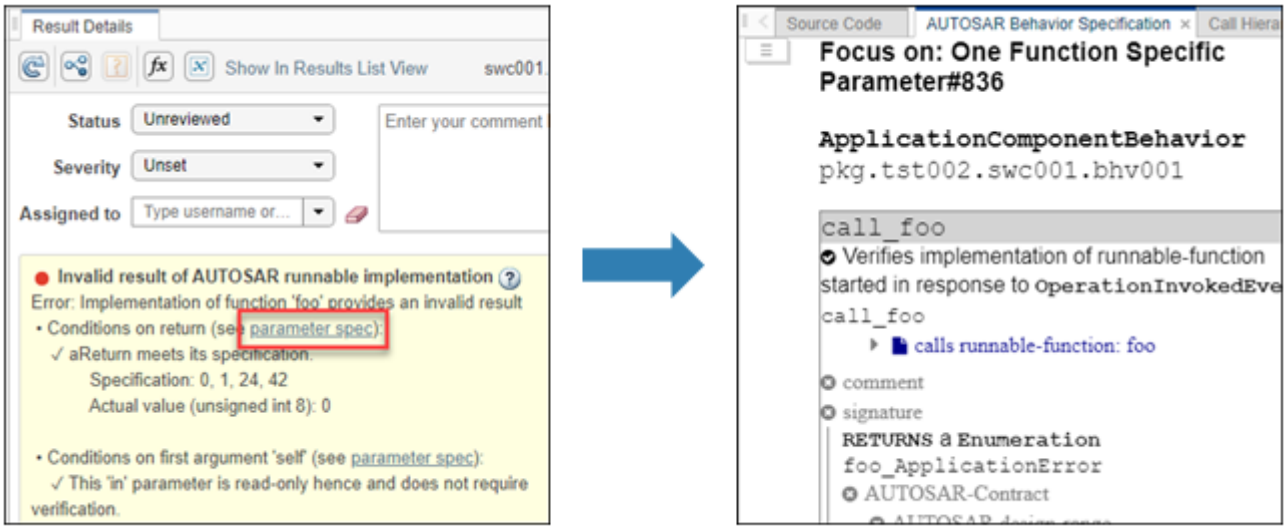
New Features

Bug Fixes

Dashboard and Review in Web Browser

AUTOSAR Support: Navigate from Polyspace findings to AUTOSAR ARXML specifications

In R2020a, if you run Polyspace on AUTOSAR code and upload the results to Polyspace Access, you can navigate from the AUTOSAR finding to the corresponding AUTOSAR behavior specification.

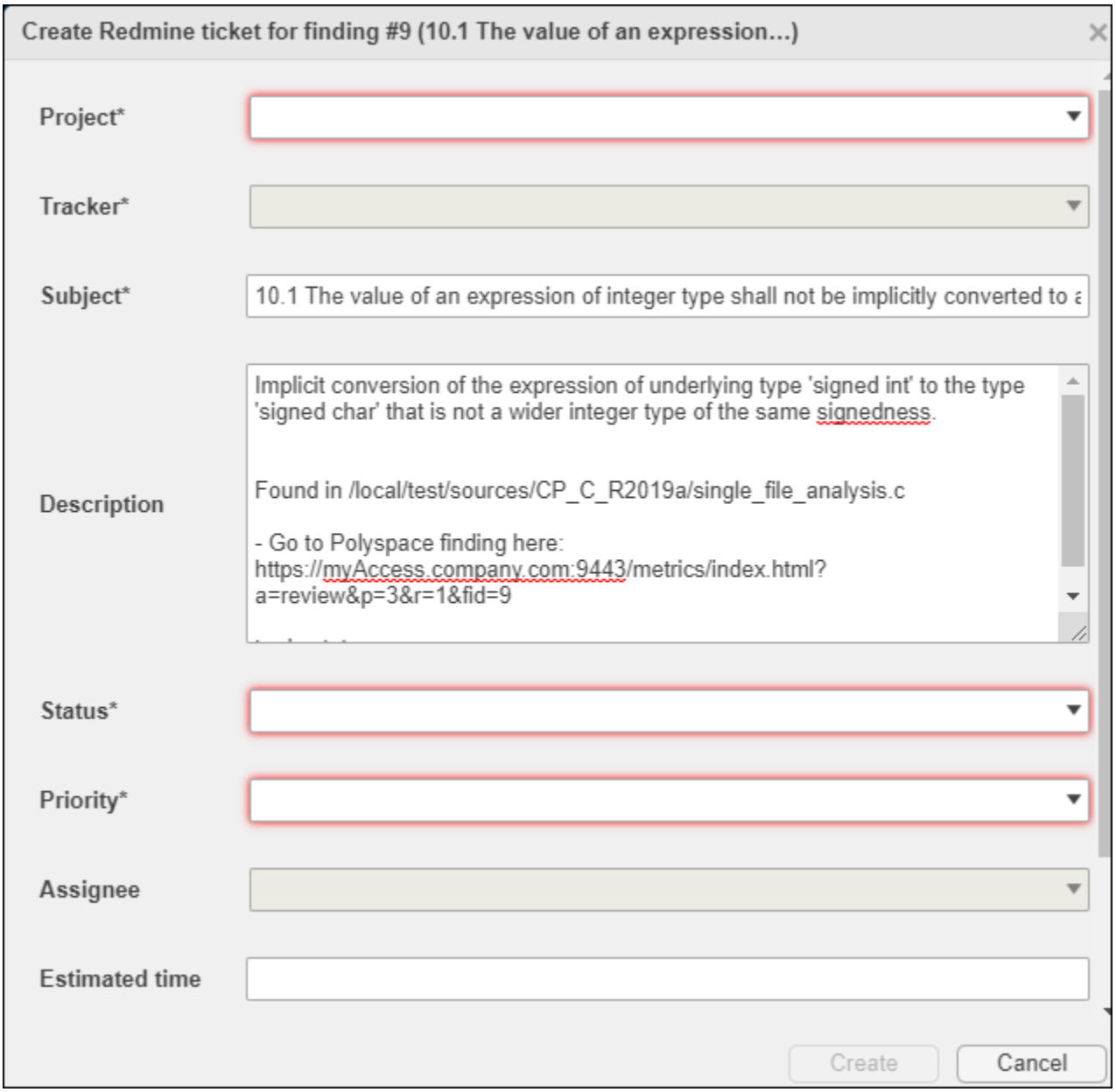


From the **Result Details** pane for an AUTOSAR finding, click **parameter spec** to open the **AUTOSAR behavior specification**. You can use the information in the spec to further investigate the cause of an AUTOSAR finding.

See also Review Polyspace Results on AUTOSAR Code.

Bug Tracking Tool Support: Create Redmine tickets for Polyspace Access results and assign to developers

In R2020a, Polyspace Access supports integration with the Redmine bug tracking tool. If you use Redmine, after you configure Polyspace Access, you can create a Redmine ticket to track Polyspace findings. The ticket is populated with details of the finding and a link to open that finding in Polyspace Access. You can add the ticket to any existing Redmine project.



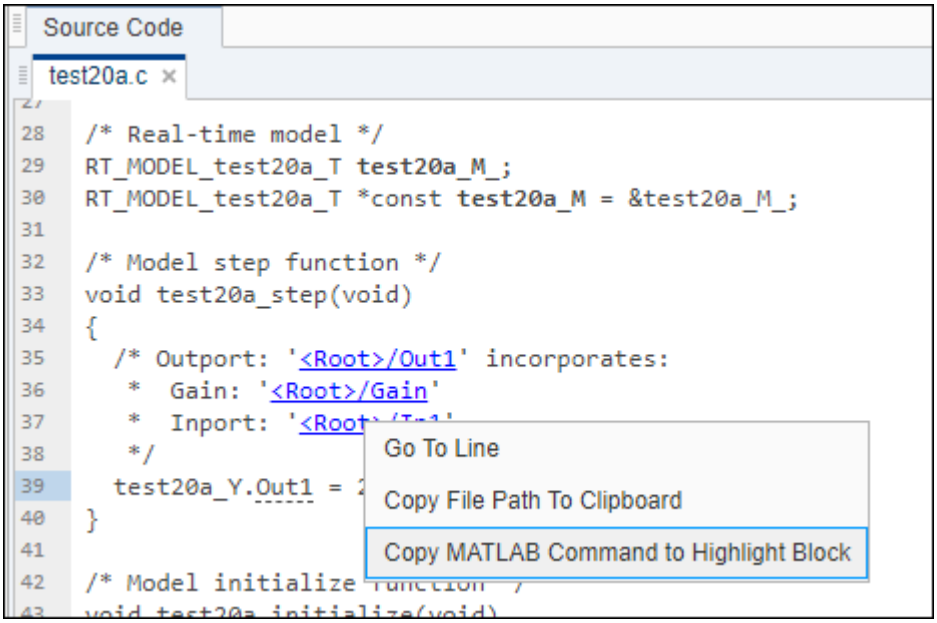
Once you create a ticket, the **Result Details** pane displays a link that you can click to open the ticket in the Redmine interface. See also Track Issue in Bug Tracking Tool.

Simulink Support: Navigate from generated code in Polyspace Access to blocks in model

In R2020a, if you run Polyspace on generated code in Simulink® and upload the results to Polyspace Access, you can navigate from the source code in Polyspace Access to blocks in the model.

On the **Source Code** pane in the Polyspace Access web interface, links in code comments show blocks that generate the subsequent lines of code. To see the block in the model:

1 Right-click a link and select **Copy MATLAB Command to Highlight Block**.

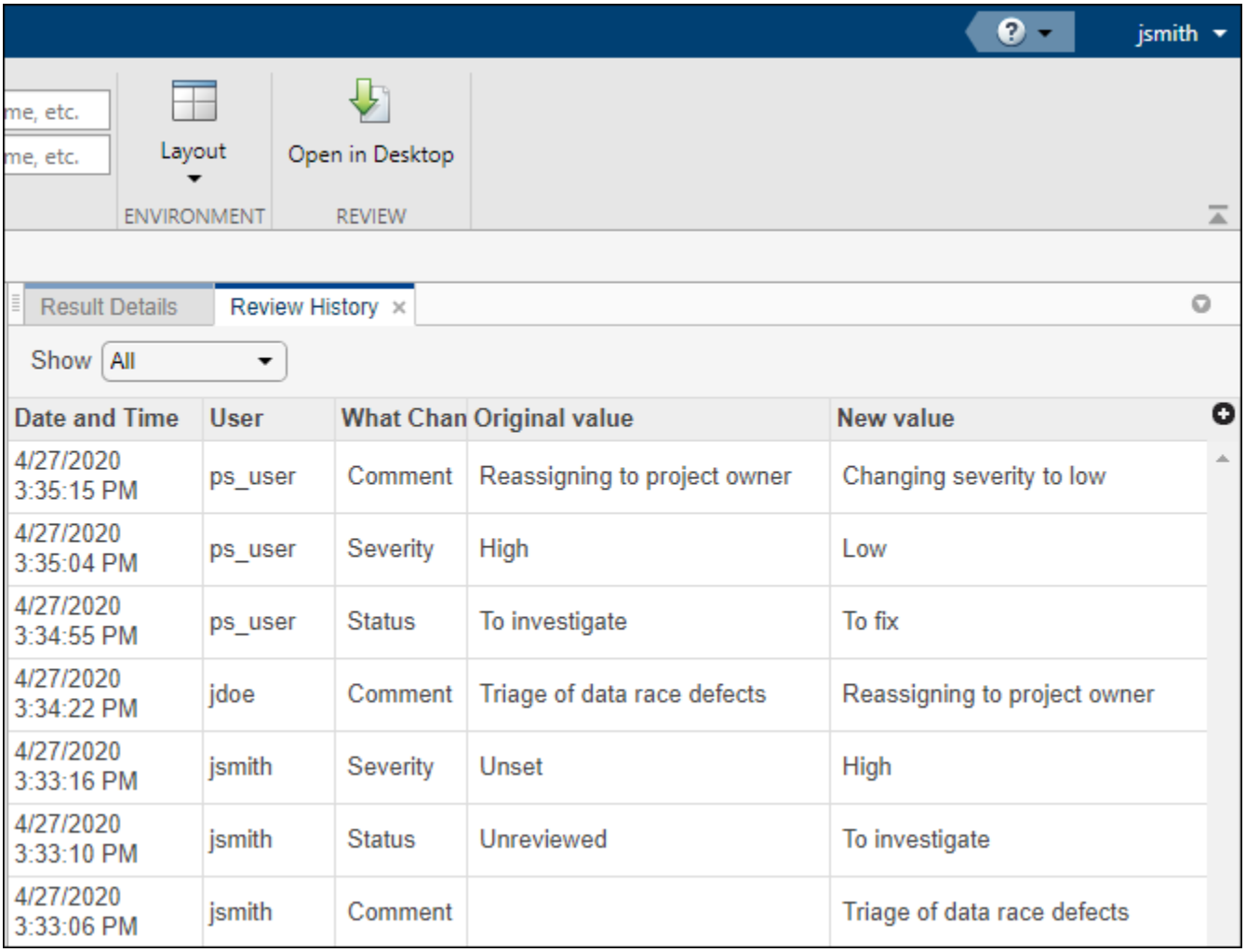


This action copies the MATLAB® command required to highlight the block. The command uses the `Simulink.ID.hilite` function.

2 In MATLAB, with the model open, paste and run the copied command.

Results Review: See review history of findings

In R2020a, you can open the **Review History** pane to see all the changes to the review fields of findings with a timestamp and the name of the user who made the change. On the Polyspace Access toolstrip, select **Layout > Show/Hide View**.



Date and Time	User	What Chan	Original value	New value
4/27/2020 3:35:15 PM	ps_user	Comment	Reassigning to project owner	Changing severity to low
4/27/2020 3:35:04 PM	ps_user	Severity	High	Low
4/27/2020 3:34:55 PM	ps_user	Status	To investigate	To fix
4/27/2020 3:34:22 PM	jdoe	Comment	Triage of data race defects	Reassigning to project owner
4/27/2020 3:33:16 PM	jsmith	Severity	Unset	High
4/27/2020 3:33:10 PM	jsmith	Status	Unreviewed	To investigate
4/27/2020 3:33:06 PM	jsmith	Comment		Triage of data race defects

You can use this information to better understand how and why the **Severity** or **Status** of a finding has changed, and retrieve previous comments that were overwritten.

For more information, see Review History.

Results Review: See the configuration options used for analysis

In R2020a, you can open the **Configuration Settings** pane to view the Polyspace configuration options that were enabled to generate the analysis results. On the Polyspace Access toolstrip, select **Layout > Show/Hide View**.

Results List | Configuration Settings ×

Verification Options | Checkers configuration

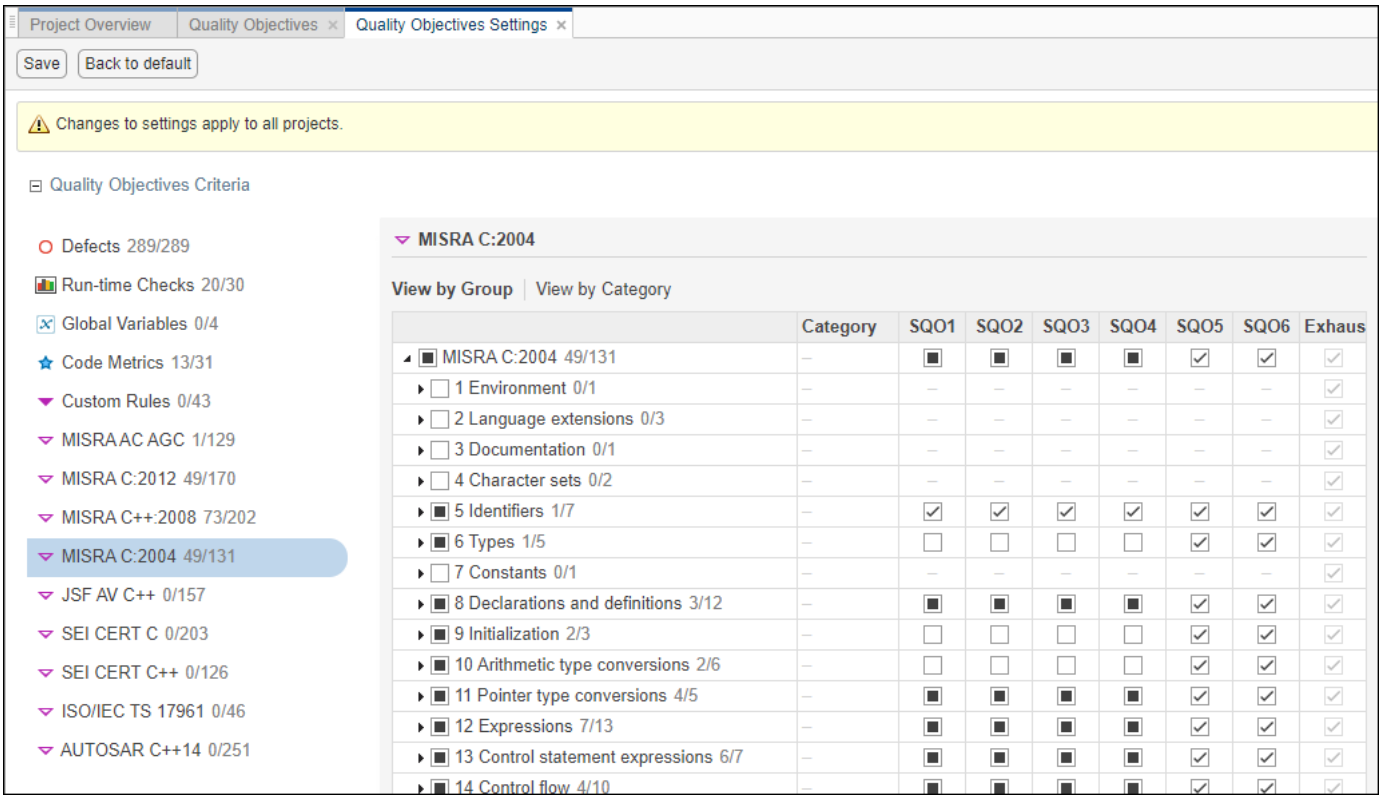
Options	Value
-author	MathWorks
-checkers	BAD_PLAIN_CHAR_USE, BITWISE_NEG, FLOAT_ABSORPTION, FLOAT_CONV_OVFL, FLOAT_OVFL, FLOAT_STD_LIB, FLOAT_ZERO_DIV, INT_CONSTANT_OVFL, INT_CONV_OVFL, INT_OVFL, INT_PRECISION_EXCEEDED, INT_STD_LIB, INT_TO_FLOAT_PRECISION_LOSS, INT_ZERO_DIV, INVALID_OPERATION_ON_BOOLEAN, SHIFT_NEG, SHIFT_OVFL, SIGN_CHANGE, UINT_CONSTANT_OVFL, UINT_CONV_OVFL, UINT_OVFL
-compiler	gnu4.6
-critical-section-begin	BEGIN_CRITICAL_SECTION:Cs10, acquire_sensor:Cs11, acquire_printer:Cs12, acquire_sensor2:Cs13, acquire_printer2:Cs14
-critical-section-end	END_CRITICAL_SECTION:Cs10, release_sensor:Cs11, release_printer:Cs12, release_sensor2:Cs13, release_printer2:Cs14
-date	08/12/2019
-do-not-generate-results-for	all-headers
-dos	true
-entry-points	bug_datarace_task1, bug_datarace_task2, bug_datarace_task3, bug_datarace_task4, bug_deadlock_task1, bug_deadlock_task2, bug_doublelock_task, bug_doubleunlock_task, bug_badlock_task, bug_badunlock_task, bug_dataracestdlib_task1, bug_dataracestdlib_task2, bug_destroylocked_task, corrected_datarace_task1, corrected_datarace_task2, corrected_datarace_task3, corrected_datarace_task4, corrected_deadlock_task1, corrected_deadlock_task2, corrected_doublelock_task, corrected_doubleunlock_task, corrected_badlock_task, corrected_badunlock_task, corrected_dataracestdlib_task1, corrected_dataracestdlib_task2, corrected_destroylocked_task
-lang	C
-misra3	mandatory
-prog	Bug_Finder_Example
-results-dir	D:\Polyspace\Bug_Finder_Example\BF_Result_1
-target	x86_64
-verif-version	1.0

You can use this information to better understand your results. For instance, you might expect to see a certain coding rule violation but the checker for this rule is not enabled. Previously, you had to parse the **Run Log** to see which options and checkers were enabled.

For more information, see Configuration Settings.

Code Quality Objectives: Customize thresholds used to track the quality of your code

In R2020a, if you use Quality Objectives to track the quality of your code, you can customize the thresholds you use as pass/fail criteria to better align with your company or project requirements. For instance, you can define quality gates to ensure adherence to a specific external coding standard.

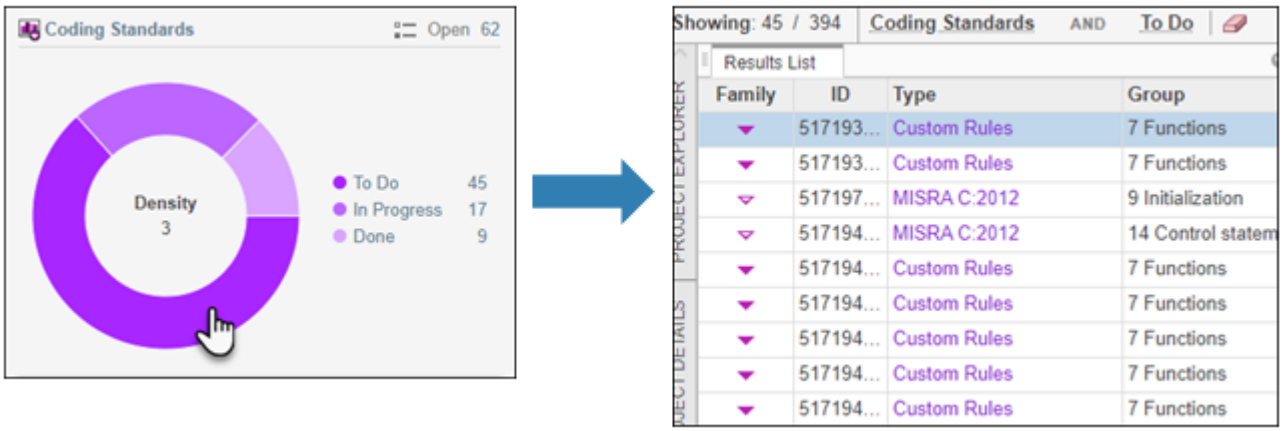


To make changes to the quality objectives settings, you must have a role of **Administrator**.

Previously, you could track the quality of your code only against (Polyspace Bug Finder Access)predefined thresholds. See Customize Software Quality Objectives.

Project Dashboard: Open results by clicking Dashboard charts

In R2020a, you can click a section of a pie chart or the legend of a pie chart to open the corresponding findings in the **Results List** and more easily narrow the scope of your review.



Bug Tracking Tool Support: Manage tickets for multiple findings

In R2020a, if you create a bug tracking tool ticket in Polyspace Access, you can select multiple findings that you associate with the ticket. If a ticket already exists, you can add that ticket to additional findings or you can detach the ticket from findings that are associated with the ticket.

Previously, you could create a ticket for only one finding at a time and you could not detach a ticket from a finding.

For more information, see Track Issue in Bug Tracking Tool.

Results Review: View error call graph

In R2020a, if you review **Run-Time Checks**, click the icon to open the **Error Call Graph** pane.

The pane displays the call sequence that leads to the detected finding. Click a node on the graph to navigate back to the source code.

Results Review: View variable access graph

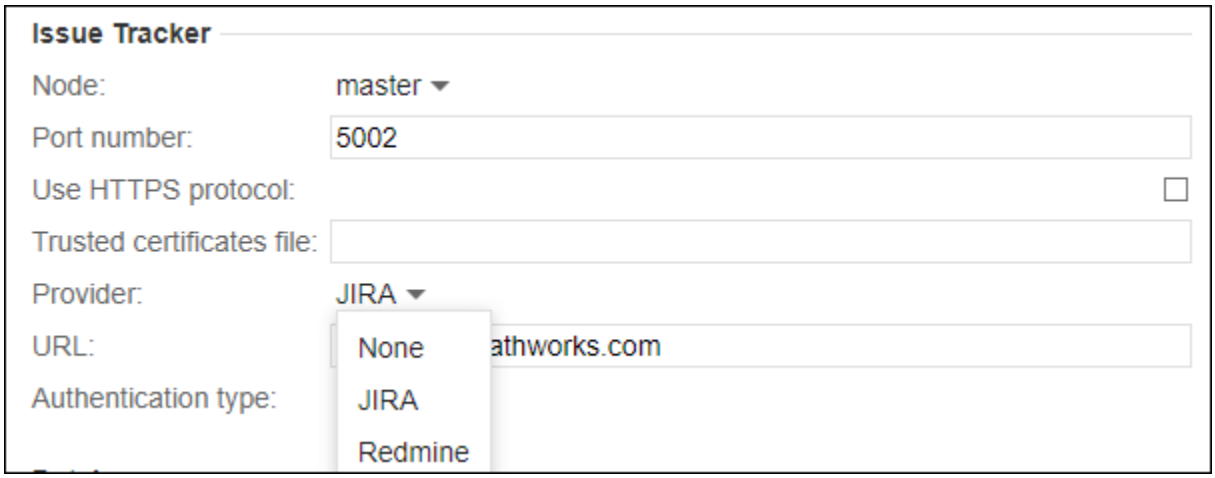
In R2020a, if you review **Global Variables** findings, click the icon to open the **Variable Access Graph** pane.

The pane displays a graphical representation of the access operations on global variables. Click a node on the graph to navigate back to the source code at the location of calling and called functions.

Installation

Installation and Configuration: Issue Tracker service

In R2020a, use the new **Issue Tracker** service to configure Polyspace Access to integrate with the Jira software or Redmine bug tracking tools.



See Configure the **User Manager** and **Issue Tracker**.

Installation and Configuration: Change in default location of Polyspace Access data volume and working directories

In R2020a, the default location of the working directories of the Polyspace Access **Web Server** and **ETL** services and of the data volume is inside the folder where you unzipped the Polyspace Access ZIP file, under the `polyspace` folder.

Previously, the working directories of the **Web Server** and **ETL** were stored in the temporary files folder of your system (`/tmp` on Linux or `%TEMP%` on Windows). The data volume was stored under `/var/lib/docker/volumes` on Linux.

R2019b

Version: 2.1

New Features

Bug Fixes

Installation

User Authentication: Use LDAP search filters to restrict number of users to authenticate

In R2019b, if you use your organization's Lightweight Directory Access Protocol (LDAP) to authenticate users, you can filter for and load a subset of users from your LDAP database when you start Polyspace Code Prover™ Access™. Previously, you loaded all LDAP users listed under the **LDAP base** that you specified when you started Polyspace Code Prover Access.

To filter the LDAP users, use the new **LDAP search filter** field in the Cluster Operator settings for the **User Manager** service. For more information, see Use Your Organization LDAP.

User Management: Update list of users from LDAP database or LDIF file

In R2019b, if you remove users from your organization's Lightweight Directory Access Protocol (LDAP) database or from the Polyspace Access embedded LDAP LDIF file, you can update the list of users stored in the Polyspace Access database. Previously, users that were removed from the LDAP database or from the LDIF file were still visible in the list of users you selected when assigning findings or managing project permissions.

To update the list of users stored in the Polyspace Access database, append `/users/list/removed` to the URL that you use to Open the Polyspace Access Web Interface. Only an **Administrator** can perform this operation. For more information, see Manage LDAP Users in Polyspace Access.

R2019a

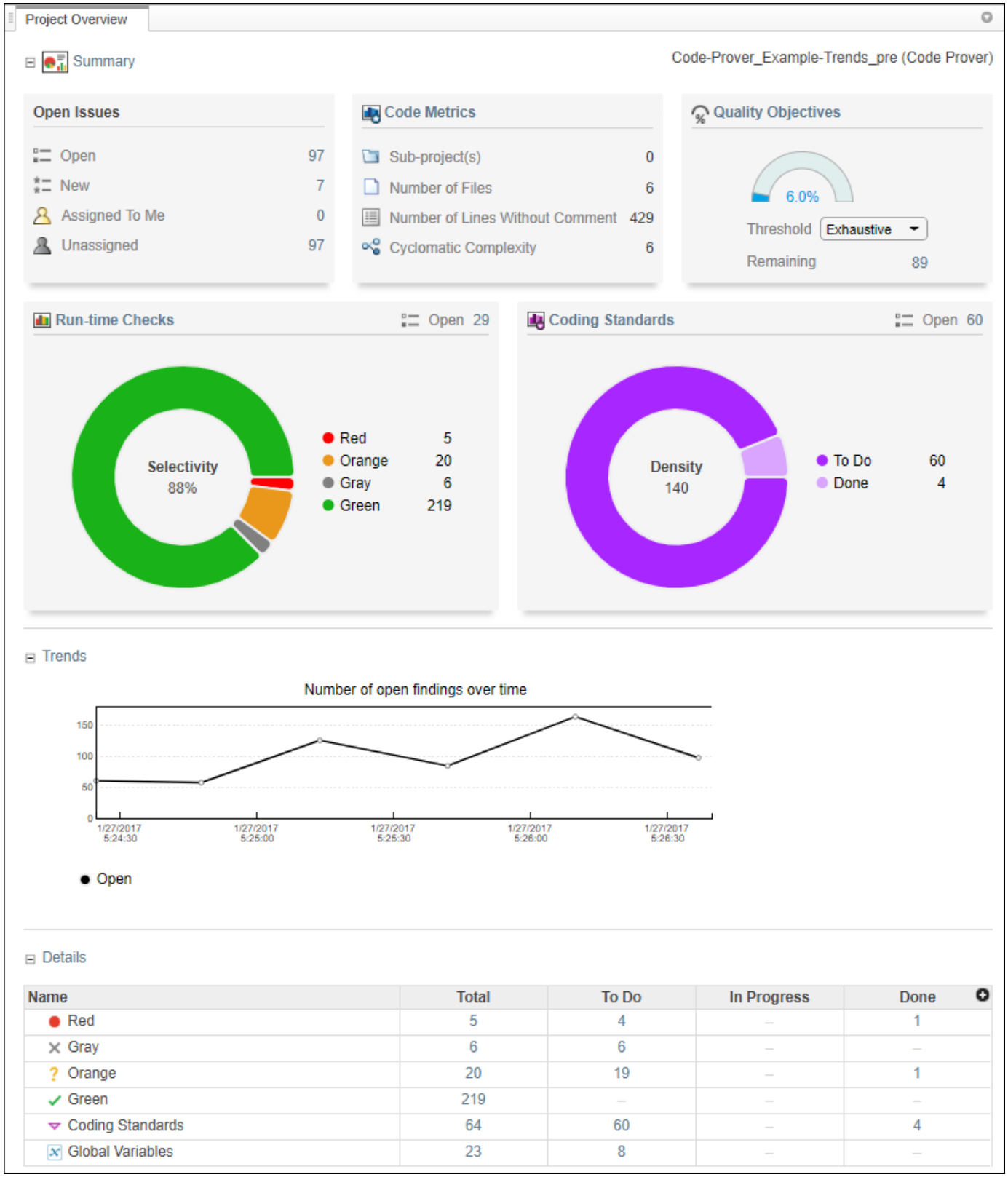
Version: 2.0

New Features

Dashboard and Review in Web Browser

Project Dashboard: Track progress of code quality via Polyspace results

Summary: In R2019a, you can track the progress of the code quality of your projects using the new intuitive Polyspace Code Prover Access **DASHBOARD**. When an analysis run is uploaded to the Polyspace Access database, the dashboard updates to give a snapshot of all the available findings, including a progress trend for number of findings compared to previous runs.

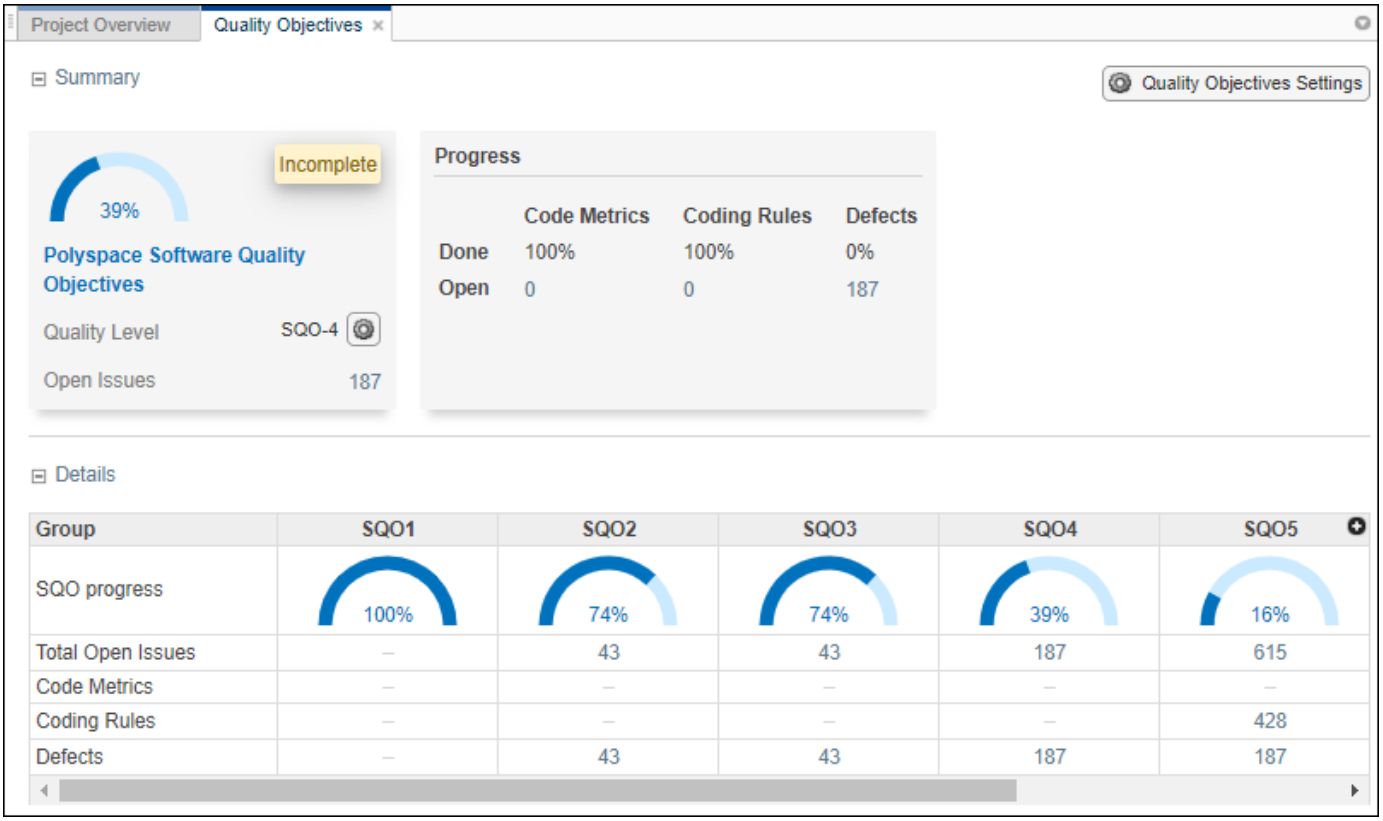

Project Overview
Summary
Code-Prover_Example-Trends_pre (Code Prover)
Open Issues
Open 97
New 7
Assigned To Me 0
Unassigned 97
Code Metrics
Sub-project(s) 0
Number of Files 6
Number of Lines Without Comment 429
Cyclomatic Complexity 6
Quality Objectives
6.0%
Threshold Exhaustive
Remaining 89
Run-time Checks
Open 29
Selectivity
88%
Red 5
Orange 20
Gray 6
Green 219
Coding Standards
Open 60
Density
140
To Do 60
Done 4
Trends
Number of open findings over time
150
100
50
0
1/27/2017 5:24:30
1/27/2017 5:25:00
1/27/2017 5:25:30
1/27/2017 5:26:00
1/27/2017 5:26:30
Open
Details
Name
Total
To Do
In Progress
Done
Red 5 4 – 1
Gray 6 6 – –
Orange 20 19 – 1
Green 219 – – –
Coding Standards 64 60 – 4
Global Variables 23 8 – –

Additional Benefits:

- *Prioritize reviews:* See new and open issues that have not been fixed or justified, then open a detailed results list for just those issues. You can drill down on a set of findings filtered by new, open, unassigned, by family of findings, or by file.
- *Aggregate results for multiple projects:* If your team works on multiple projects, you can move all of those under an umbrella project and view a snapshot of the code quality for all your team's projects.
- *Authenticate client access:* The web interface is behind a login. Only users with a Polyspace Code Prover Access license and the appropriate credentials can view the dashboard from their web browser.

Project Dashboard: Compare Polyspace Code Prover results against Software Quality Objectives

Summary: In R2019a, check the quality of your code against pre-defined quality objective thresholds using the new Polyspace Code Prover Access **Quality Objectives** dashboard web interface. Use the thresholds to establish PASS/FAIL criteria for the code quality of your projects. For instance, the dashboard displays the progress and remaining open issues across thresholds and categories of findings. Use the available dropdown menu to select a threshold and see a more detailed view of completion by category of finding.

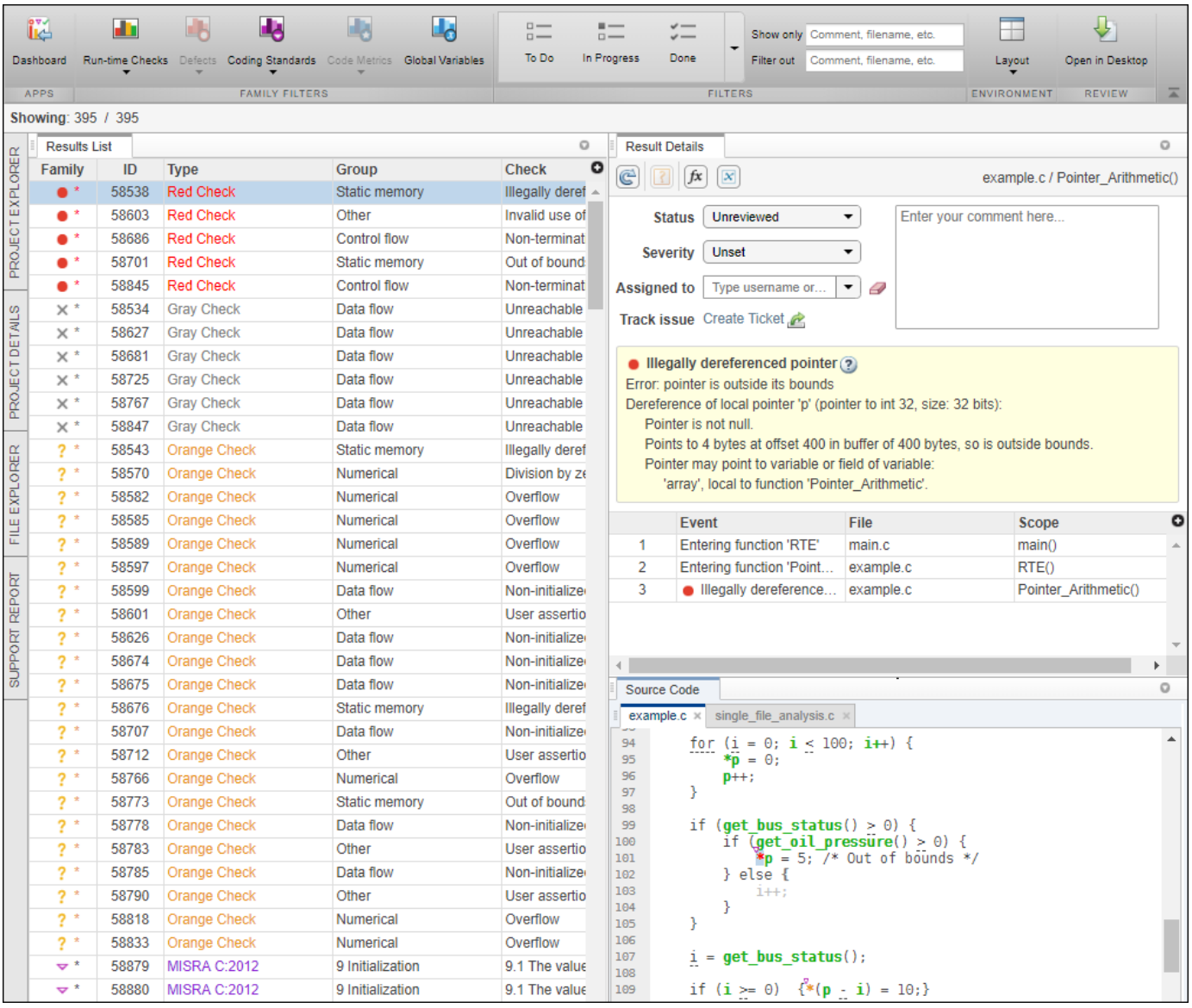


Additional Benefits:

- *Achieve better quality code:* The dashboard lets you drill down to categories of open issues for each threshold. Click a cell in the table to open a list of findings you need to address to pass a given quality objective threshold.

Collaborative Review Support: Review Polyspace Code Prover results and source code in web browser

Summary: In R2019a, review Polyspace analysis findings and view the findings in your source code using the new Polyspace Code Prover Access **REVIEW** web interface. You do not need to install a Polyspace product on your machine to open and review analysis results.



Additional Benefits:

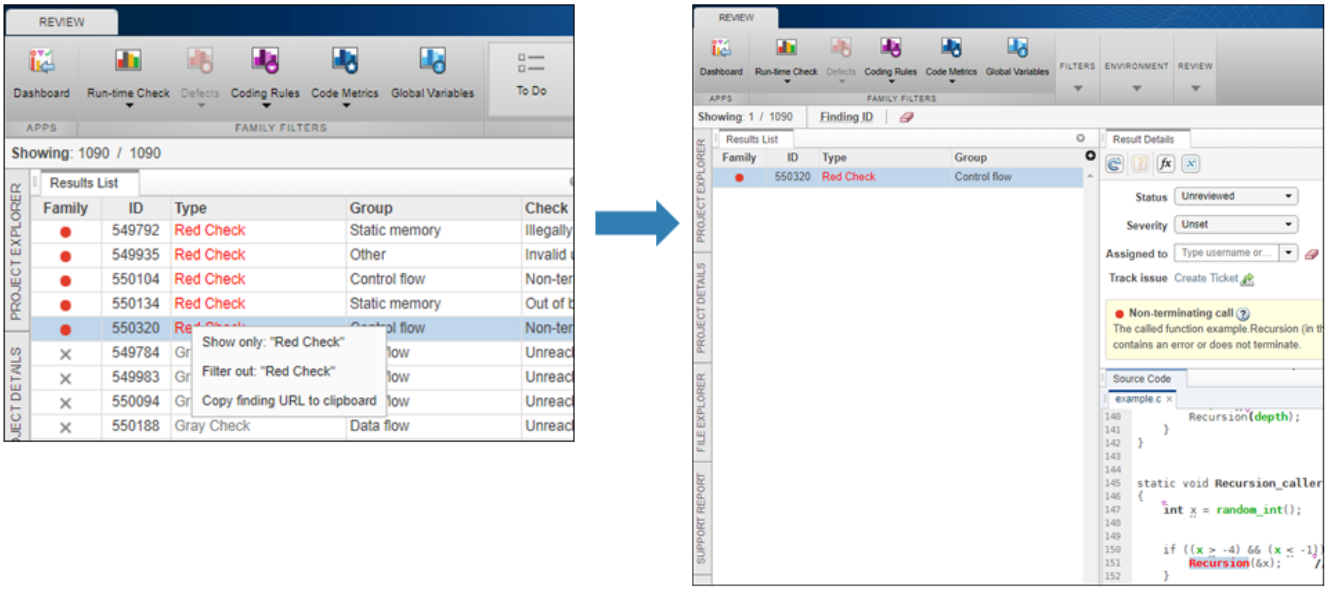
- *Facilitate collaborative review:* The web interface streamlines the review efforts of your team. For instance:

- During a team meeting, findings can be assessed and assigned to developers.
- Developers can log into the web interface to review findings assigned to them, and determine whether to justify the findings or fix them.
- A project manager can track the progress of the review by filtering the list of results for findings that are still open.

- *Authenticate client access:* The web interface is behind a login. Only users with a Polyspace Code Prover Access license and the appropriate credentials can view the results from their web browser.

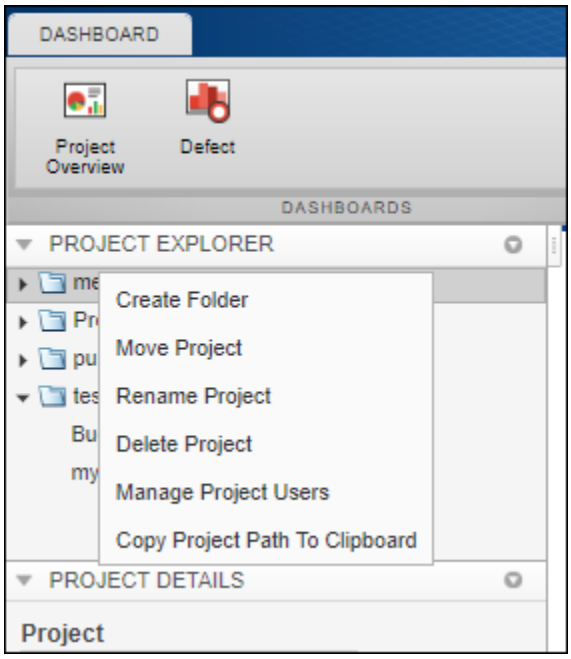
Collaborative Review Support: Share Polyspace Code Prover results using web links

Summary: In R2019a, you can right-click an analysis result in the Polyspace Code Prover Access interface to obtain a URL that you can share with other team members. The link that you provide opens the Polyspace Code Prover Access interface and displays the finding along with the corresponding source code.



Project Authorization Management: Create and enforce authorization policies for access to project

Summary:In R2019a, you can manage project users in Polyspace Code Prover Access by right-clicking a project in the **PROJET EXPLORER** and assigning roles to member of your team. The roles authorize or forbid users from viewing projects.

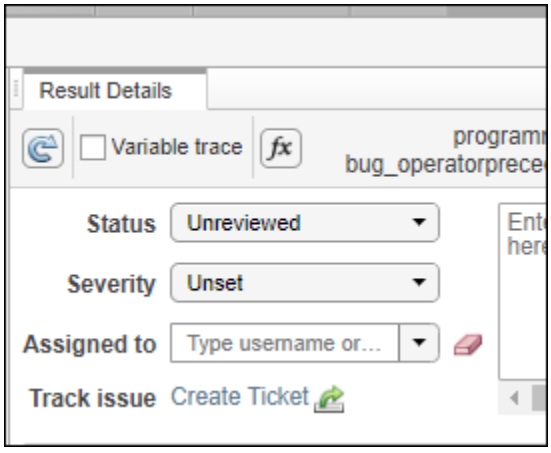


Additional Benefits:

- *Restrict access to your source code:* Use the authorization policy to restrict who can view the source code you upload with your analysis results.
- *Display relevant projects only:* When they log in to Polyspace Access, users can only see projects for which they are administrators, owners, or contributors. Use the authorization policy so that team members only see projects that they are working on.

Bug Tracking Tool Support: Create JIRA issues for Polyspace Code Prover results and assign to developer

Summary: In R2019a, Polyspace Code Prover supports integration with the JIRA software. If you have an instance of the JIRA software, after you configure Polyspace Code Prover, you can create a JIRA ticket to track Polyspace findings. The ticket is populated with details of the finding and a link to open that finding in Polyspace Access. You can add the ticket to any existing JIRA project.



Once you create a ticket, the **Result Details** pane in the Polyspace Code Prover web interface displays a link to the corresponding JIRA issue.