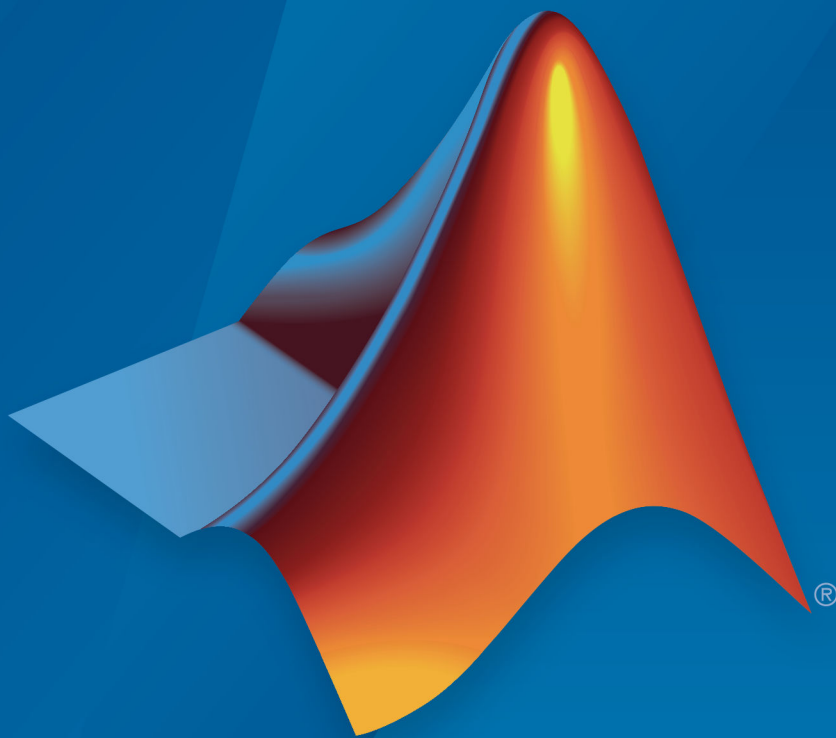


Polyspace[®] Code Prover[™] Access[™] Release Notes



MATLAB[®]

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 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.

R2019a

Project Dashboard: Track progress of code quality via Polyspace results	1-2
Project Dashboard: Compare Polyspace Code Prover results against Software Quality Objectives	1-4
Collaborative Review Support: Review Polyspace Code Prover results and source code in web browser	1-5
Collaborative Review Support: Share Polyspace Code Prover results using web links	1-7
Project Authorization Management: Create and enforce authorization policies for access to project	1-8
Bug Tracking Tool Support: Create JIRA issues for Polyspace Code Prover results and assign to developer	1-9

R2019a

Version: 2.0

New Features

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.

Summary

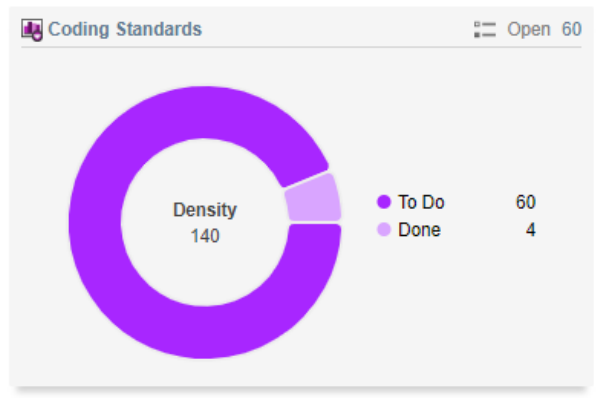
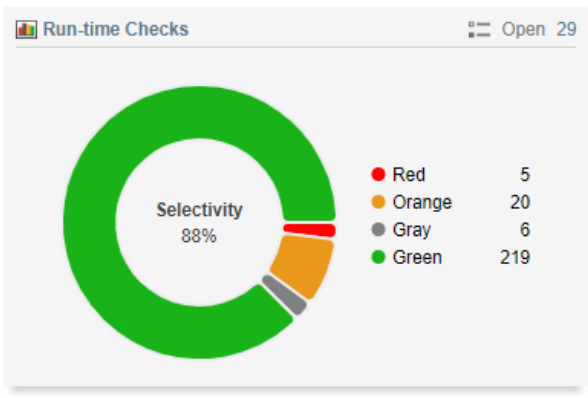
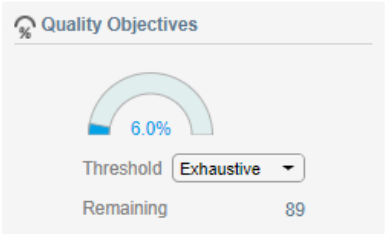
Code-Prover_Example-Trends_pre (Code Prover)

Open Issues

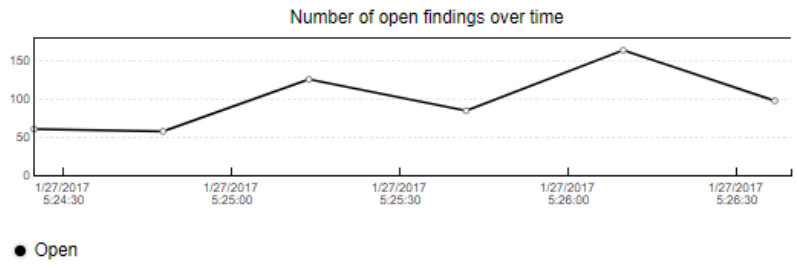
Open	97
New	7
Assigned To Me	0
Unassigned	97

Code Metrics

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



Trends



Details

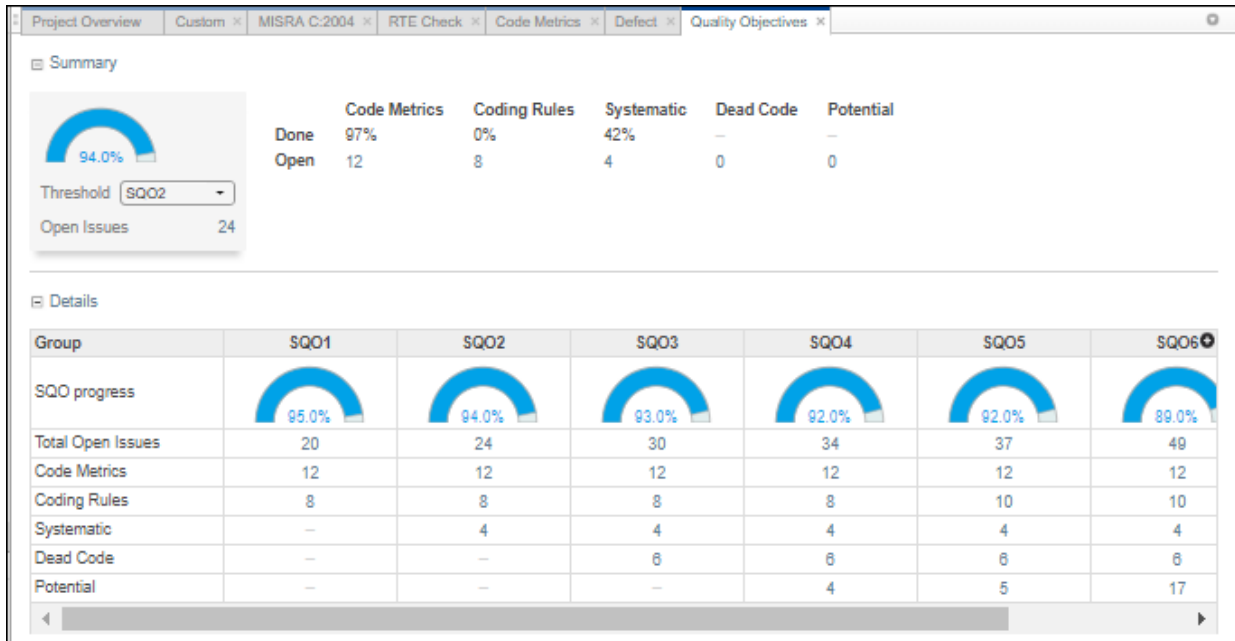
Name	Total	To Do	In Progress	Done
● Red	5	4	–	1
× Gray	6	6	–	–
? Orange	20	19	–	1
✓ Green	219	–	–	–
▽ Coding Standards	64	60	–	4
⊗ Global Variables	23	8	–	–

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.

The screenshot displays a static analysis tool interface with a top navigation bar and a main workspace. The workspace is divided into a 'Results List' table on the left and a 'Result Details' panel on the right.

Results List Table:

Family	ID	Type	Group	Check
● *	58538	Red Check	Static memory	Illegally deref
● *	58603	Red Check	Other	Invalid use of
● *	58686	Red Check	Control flow	Non-terminat
● *	58701	Red Check	Static memory	Out of bound
● *	58845	Red Check	Control flow	Non-terminat
× *	58534	Gray Check	Data flow	Unreachable
× *	58627	Gray Check	Data flow	Unreachable
× *	58681	Gray Check	Data flow	Unreachable
× *	58725	Gray Check	Data flow	Unreachable
× *	58767	Gray Check	Data flow	Unreachable
× *	58847	Gray Check	Data flow	Unreachable
? *	58543	Orange Check	Static memory	Illegally deref
? *	58570	Orange Check	Numerical	Division by ze
? *	58582	Orange Check	Numerical	Overflow
? *	58585	Orange Check	Numerical	Overflow
? *	58589	Orange Check	Numerical	Overflow
? *	58597	Orange Check	Numerical	Overflow
? *	58599	Orange Check	Data flow	Non-initialize
? *	58601	Orange Check	Other	User assertio
? *	58626	Orange Check	Data flow	Non-initialize
? *	58674	Orange Check	Data flow	Non-initialize
? *	58675	Orange Check	Data flow	Non-initialize
? *	58676	Orange Check	Static memory	Illegally deref
? *	58707	Orange Check	Data flow	Non-initialize
? *	58712	Orange Check	Other	User assertio
? *	58766	Orange Check	Numerical	Overflow
? *	58773	Orange Check	Static memory	Out of bound
? *	58778	Orange Check	Data flow	Non-initialize
? *	58783	Orange Check	Other	User assertio
? *	58785	Orange Check	Data flow	Non-initialize
? *	58790	Orange Check	Other	User assertio
? *	58818	Orange Check	Numerical	Overflow
? *	58833	Orange Check	Numerical	Overflow
▼ *	58879	MISRA C:2012	9 Initialization	9.1 The value
▼ *	58880	MISRA C:2012	9 Initialization	9.1 The value

Result Details Panel (example.c / Pointer_Arithmetic()):

Status: Unreviewed
Severity: Unset
Assigned to: Type username or...
Track issue: Create Ticket

Illegally dereferenced pointer
Error: pointer is outside its bounds
Dereference of local pointer 'p' (pointer to int 32, size: 32 bits):
Pointer is not null.
Points to 4 bytes at offset 400 in buffer of 400 bytes, so is outside bounds.
Pointer may point to variable or field of variable:
'array', local to function 'Pointer_Arithmetic'.

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

Source Code:

```

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

```

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.

The image shows two screenshots of the Polyspace Code Prover interface. The left screenshot shows a table of findings, and the right screenshot shows a detailed view of a specific finding.

Table of Findings (Left Screenshot):

Family	ID	Type	Group	Check
●	549792	Red Check	Static memory	Illegally
●	549935	Red Check	Other	Invalid
●	550104	Red Check	Control flow	Non-ter
●	550134	Red Check	Static memory	Out of b
●	550320	Red Check	Control flow	Non-ter
X	549784	Gr	low	Unread
X	549983	Gr	low	Unread
X	550094	Gr	low	Unread
X	550188	Gray Check	Data flow	Unread

Detailed View (Right Screenshot):

Showing 1 / 1090 Finding ID

Family	ID	Type	Group
●	550320	Red Check	Control flow

Result Details

Status: Unreviewed

Severity: Unset

Assigned to: Type username or...

Track issue: Create Ticket

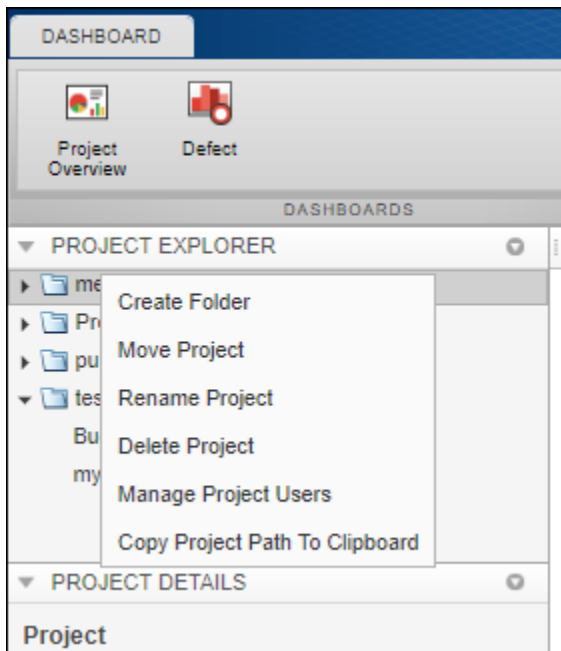
● Non-terminating call
The called function example Recursion (in th... contains an error or does not terminate.

Source Code

```
example c
148
141     Recursion(depth);
142 }
143
144
145 static void Recursion_caller
146 {
147     int x = random_int();
148
149
150     if ((x > -4) && (x <= -1))
151         Recursion(6x);
152 }
```

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 **PROJECT EXPLORER** and assigning roles to member of your team. The roles authorize or forbid users from viewing projects.

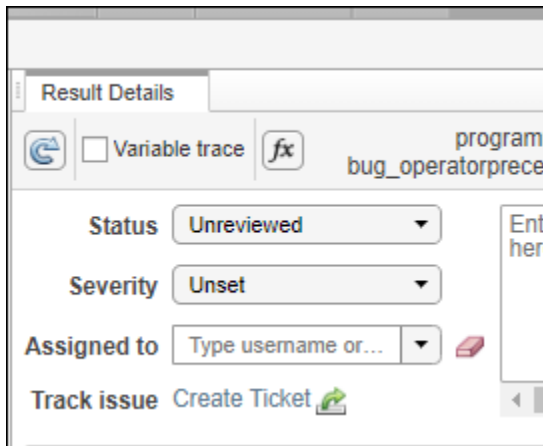


Additional Benefits:

- *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.

