Polyspace® Bug Finder™ Access™ Release Notes

MATLAB®

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® Bug Finder™ Access™ Release Notes

Contents

R2019a

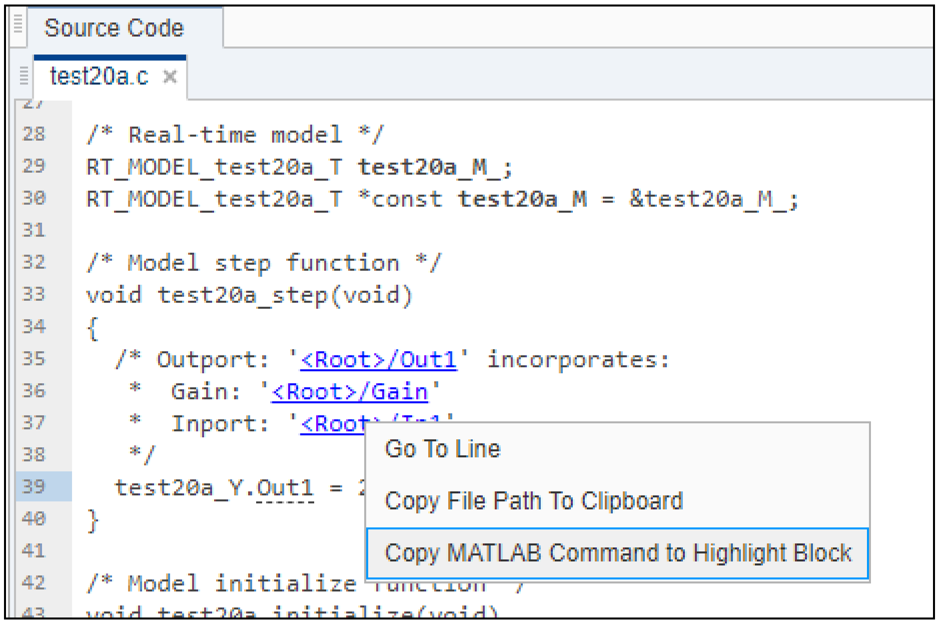
R2020a

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**.


Source Code
test20a.c
Go To Line
Copy File Path To Clipboard
Copy MATLAB Command to Highlight Block

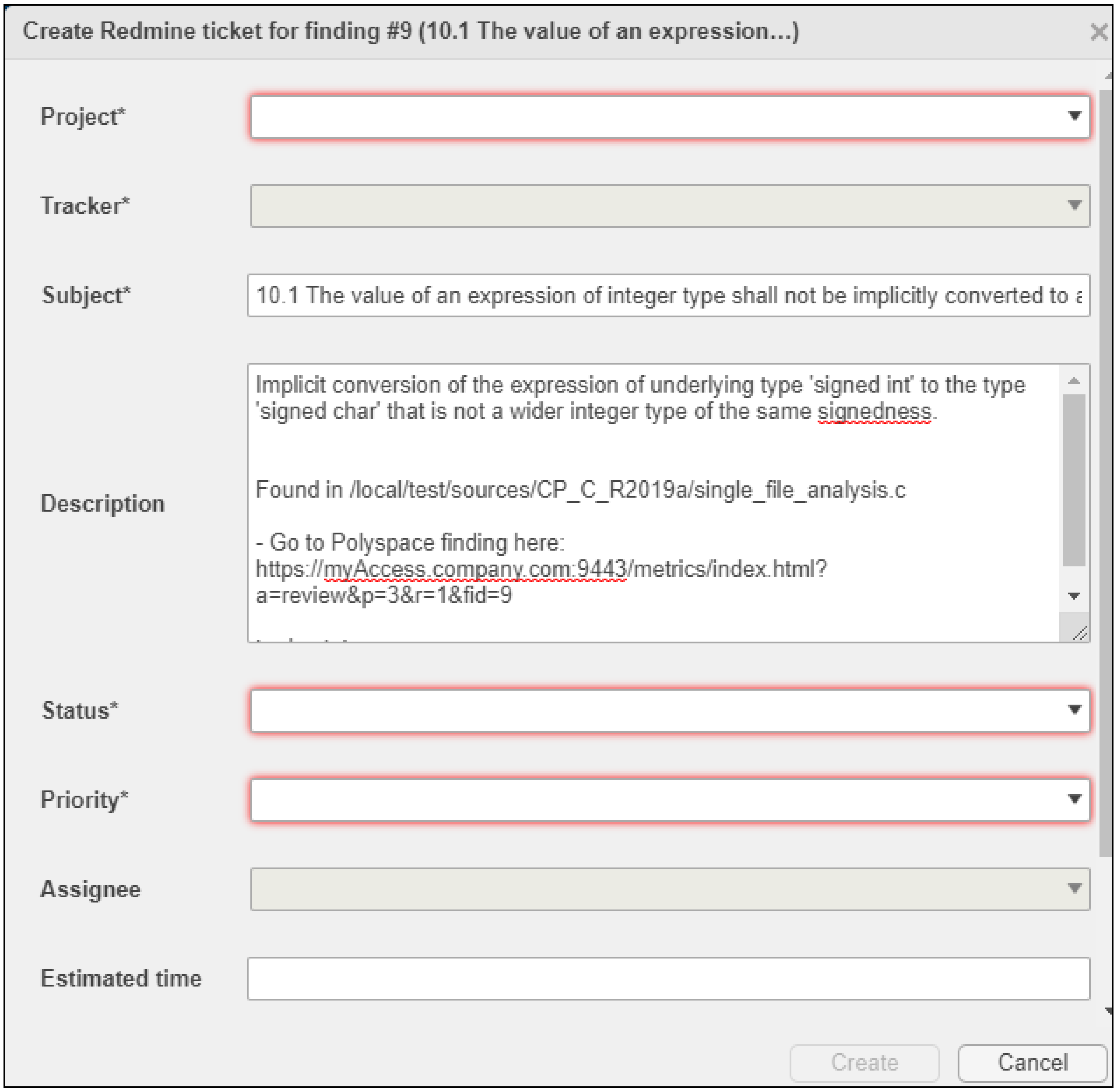
```
/* Real-time model */
RT_MODEL_test20a_T test20a_M_;
RT_MODEL_test20a_T *const test20a_M = &test20a_M_;

/* Model step function */
void test20a_step(void)
{
  /* Outport: '<Root>/Out1' incorporates:
   *  Gain: '<Root>/Gain'
   *  Inport: '<Root
   */
  test20a_Y.Out1 = 2
}

/* Model initialize
void test20a_initialize(void)
```

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.

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.


Create Redmine ticket for finding #9 (10.1 The value of an expression...)

Project*	
Tracker*	
Subject*	10.1 The value of an expression of integer type shall not be implicitly converted to a
Description	Implicit conversion of the expression of underlying type 'signed int' to the type 'signed char' that is not a wider integer type of the same signedness. Found in /local/test/sources/CP_C_R2019a/single_file_analysis.c - Go to Polyspace finding here: https://myAccess.company.com:9443/metrics/index.html?a=review&p=3&r=1&fid=9
Status*	
Priority*	
Assignee	
Estimated time	

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”.

Bug Tracking Tool Support: Manage tickets for multiple findings

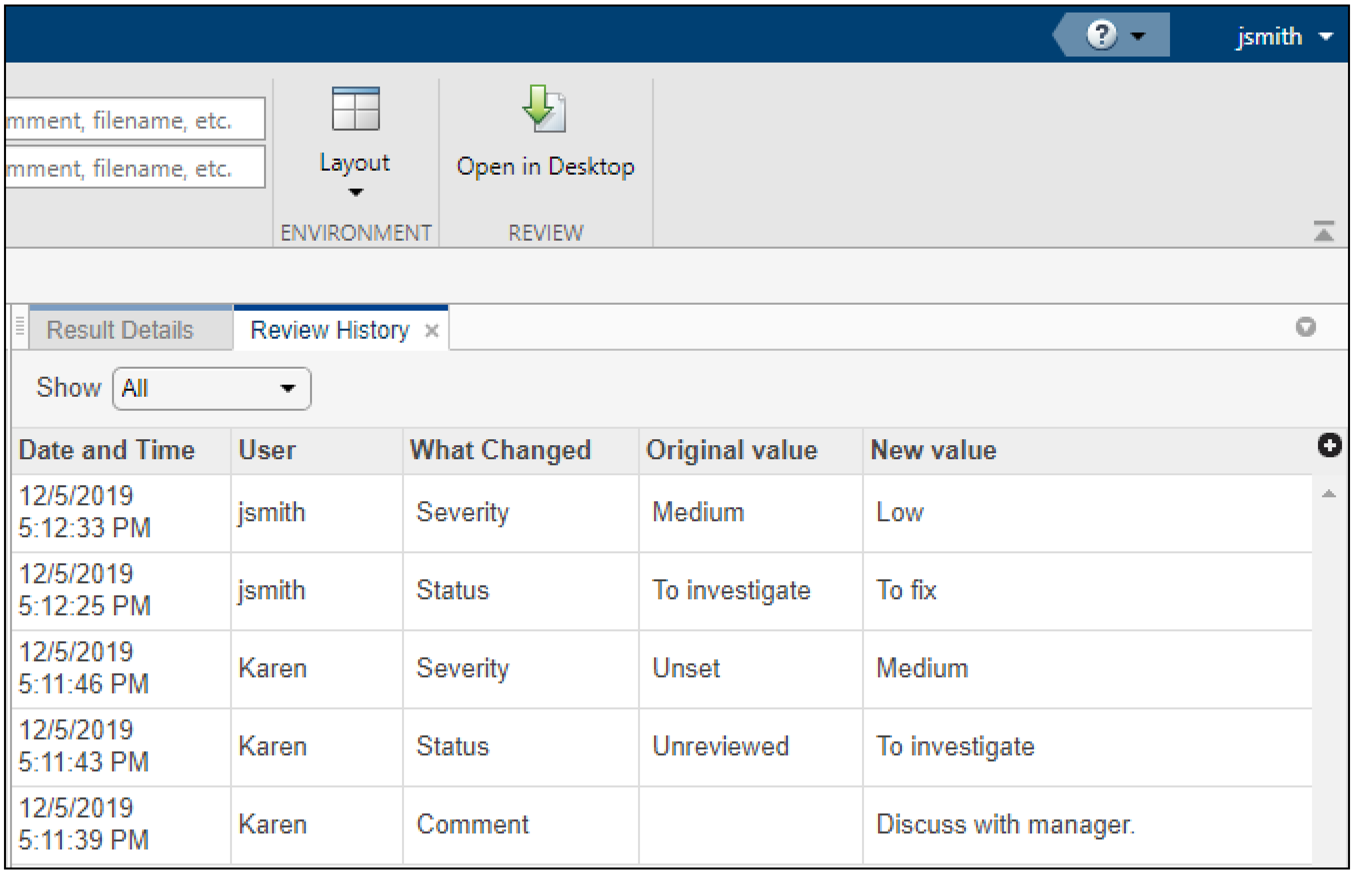
In R2020a, if you create a bug tracking tool ticket in Polyspace Access, you can select multiple findings that you associate with the ticket. If a ticket already exists, you can add that ticket to additional findings or you can detach the ticket from findings that are associated with the ticket.

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: 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**.



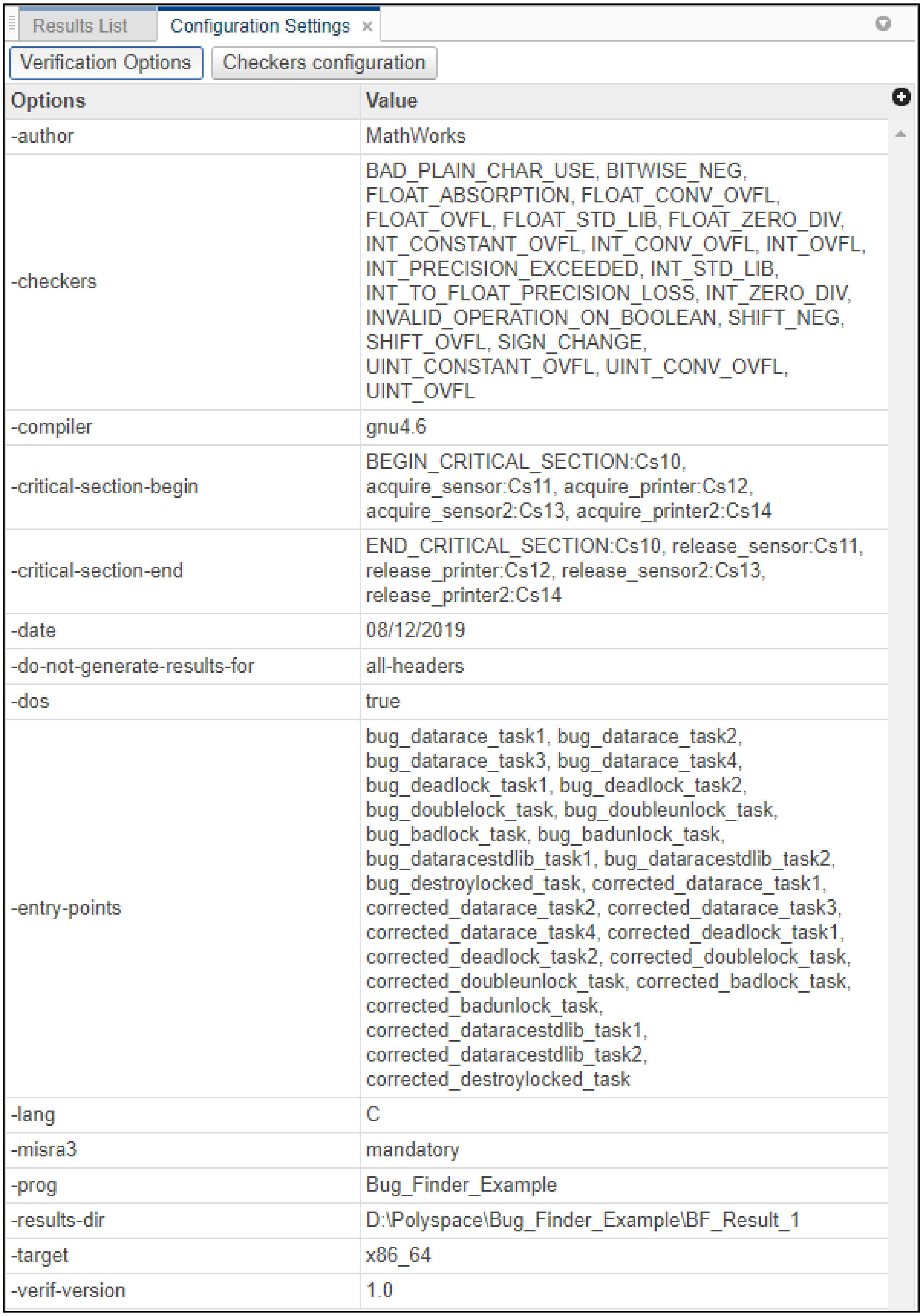
Date and Time	User	What Changed	Original value	New value
12/5/2019 5:12:33 PM	jsmith	Severity	Medium	Low
12/5/2019 5:12:25 PM	jsmith	Status	To investigate	To fix
12/5/2019 5:11:46 PM	Karen	Severity	Unset	Medium
12/5/2019 5:11:43 PM	Karen	Status	Unreviewed	To investigate
12/5/2019 5:11:39 PM	Karen	Comment		Discuss with manager.

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**.



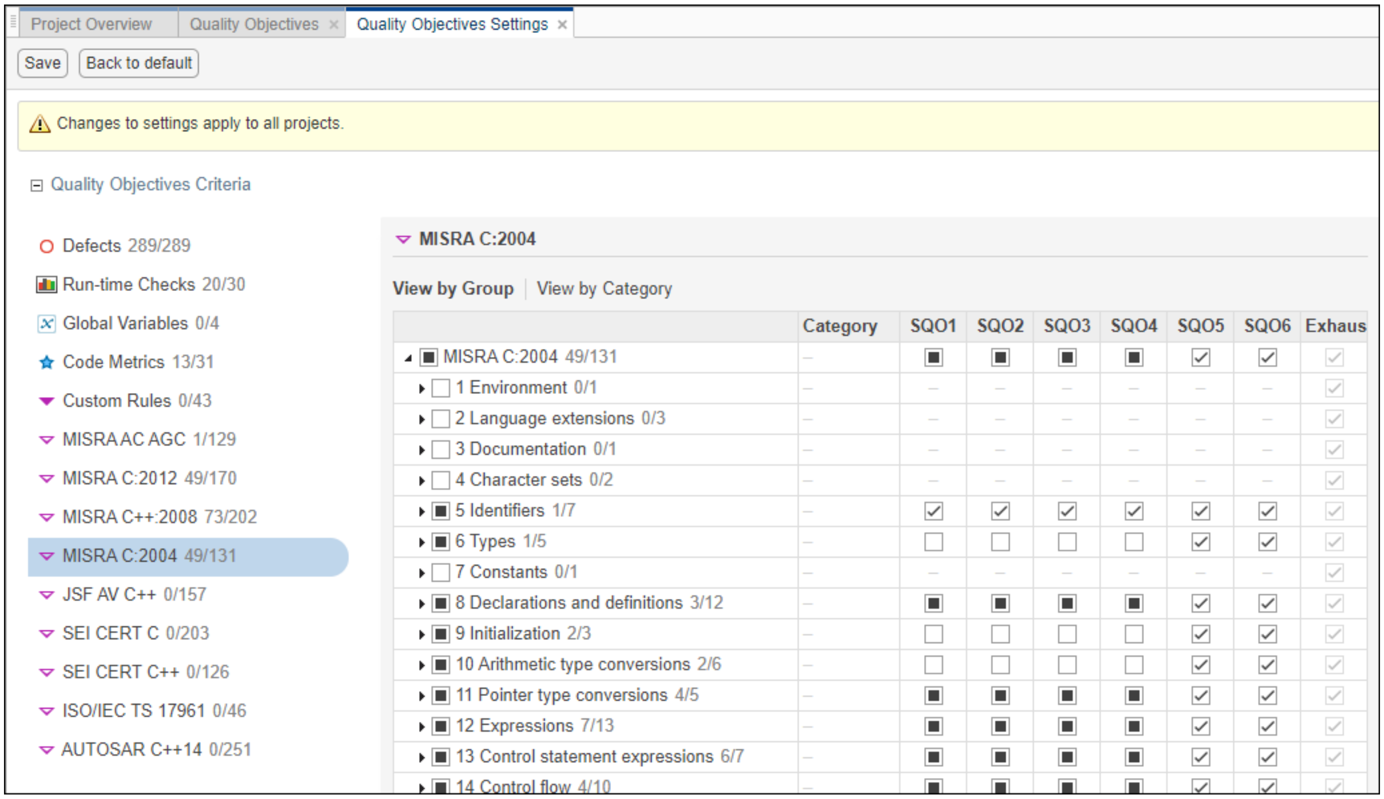
Options	Value
-author	MathWorks
-checkers	BAD_PLAIN_CHAR_USE, BITWISE_NEG, FLOAT_ABSORPTION, FLOAT_CONV_OVFL, FLOAT_OVFL, FLOAT_STD_LIB, FLOAT_ZERO_DIV, INT_CONSTANT_OVFL, INT_CONV_OVFL, INT_OVFL, INT_PRECISION_EXCEEDED, INT_STD_LIB, INT_TO_FLOAT_PRECISION_LOSS, INT_ZERO_DIV, INVALID_OPERATION_ON_BOOLEAN, 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.



Changes to settings apply to all projects.

Quality Objectives Criteria

- Defects 289/289
- Run-time Checks 20/30
- Global Variables 0/4
- Code Metrics 13/31
- Custom Rules 0/43
- MISRA AC AGC 1/129
- MISRA C:2012 49/170
- MISRA C++:2008 73/202
- MISRA C:2004 49/131
- JSF AV C++ 0/157
- SEI CERT C 0/203
- SEI CERT C++ 0/126
- ISO/IEC TS 17961 0/46
- AUTOSAR C++14 0/251

MISRA C:2004

View by Group | View by Category

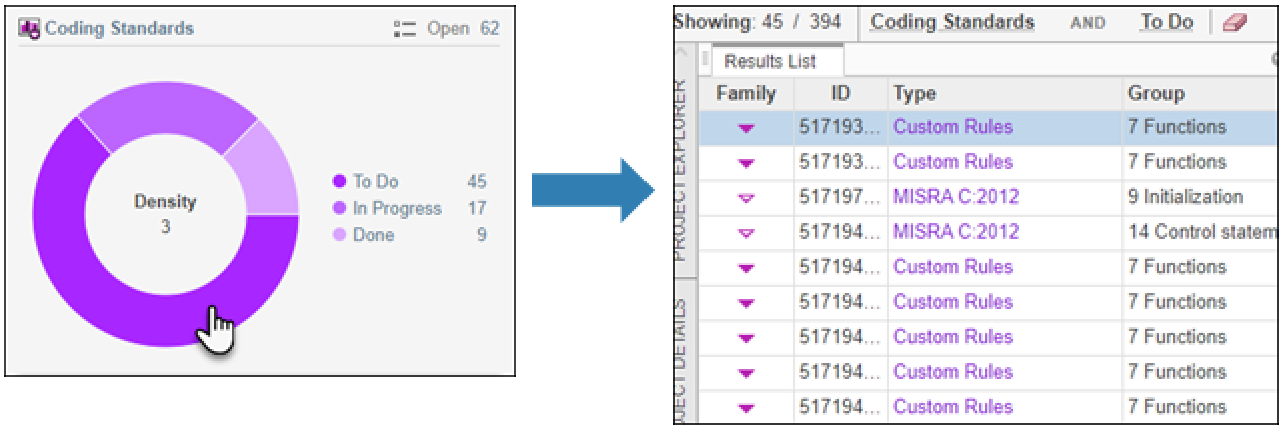
	Category	SQO1	SQO2	SQO3	SQO4	SQO5	SQO6	Exhaus
■ MISRA C:2004 49/131	–	■	■	■	■	✓	✓	✓
□ 1 Environment 0/1	–	–	–	–	–	–	–	✓
□ 2 Language extensions 0/3	–	–	–	–	–	–	–	✓
□ 3 Documentation 0/1	–	–	–	–	–	–	–	✓
□ 4 Character sets 0/2	–	–	–	–	–	–	–	✓
■ 5 Identifiers 1/7	–	✓	✓	✓	✓	✓	✓	✓
■ 6 Types 1/5	–	□	□	□	□	✓	✓	✓
□ 7 Constants 0/1	–	–	–	–	–	–	–	✓
■ 8 Declarations and definitions 3/12	–	■	■	■	■	✓	✓	✓
■ 9 Initialization 2/3	–	□	□	□	□	✓	✓	✓
■ 10 Arithmetic type conversions 2/6	–	□	□	□	□	✓	✓	✓
■ 11 Pointer type conversions 4/5	–	■	■	■	■	✓	✓	✓
■ 12 Expressions 7/13	–	■	■	■	■	✓	✓	✓
■ 13 Control statement expressions 6/7	–	■	■	■	■	✓	✓	✓
■ 14 Control flow 4/10	–	■	■	■	■	✓	✓	✓

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

Previously, you could not see quality objective statistics for Bug Finder results. 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.



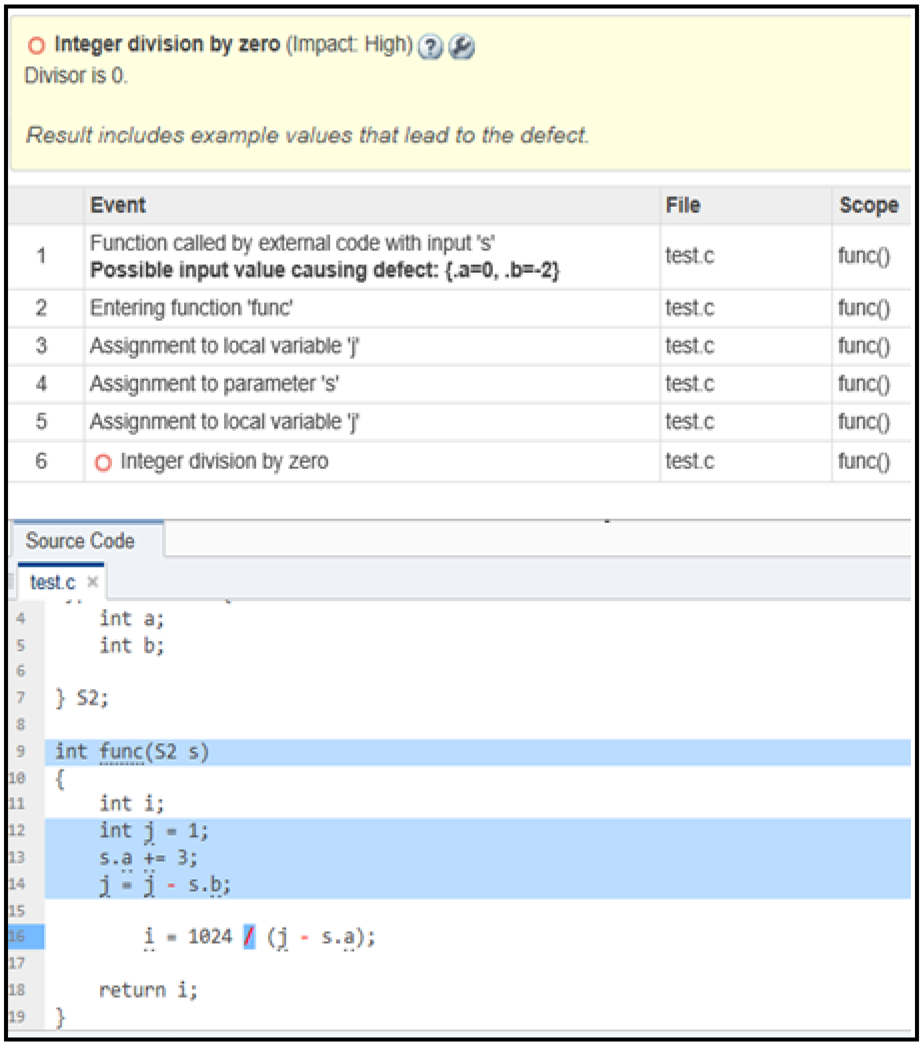
Showing: 45 / 394 | Coding Standards AND To Do

Results List

Family	ID	Type	Group
▼	517193...	Custom Rules	7 Functions
▼	517193...	Custom Rules	7 Functions
▽	517197...	MISRA C:2012	9 Initialization
▽	517194...	MISRA C:2012	14 Control statem
▼	517194...	Custom Rules	7 Functions
▼	517194...	Custom Rules	7 Functions
▼	517194...	Custom Rules	7 Functions
▼	517194...	Custom Rules	7 Functions
▼	517194...	Custom Rules	7 Functions

Extending Checkers: See example value for defect found with stricter analysis

Summary: In R2020a, if the analysis option **Run stricter checks considering all values of system inputs (-checks-using-system-input-values)** is enabled, for a subset of numerical and static memory defects, you can see an example of values that lead to the detected defect in the **Results Details**.



Integer division by zero (Impact: High)
Divisor is 0.

Result includes example values that lead to the defect.

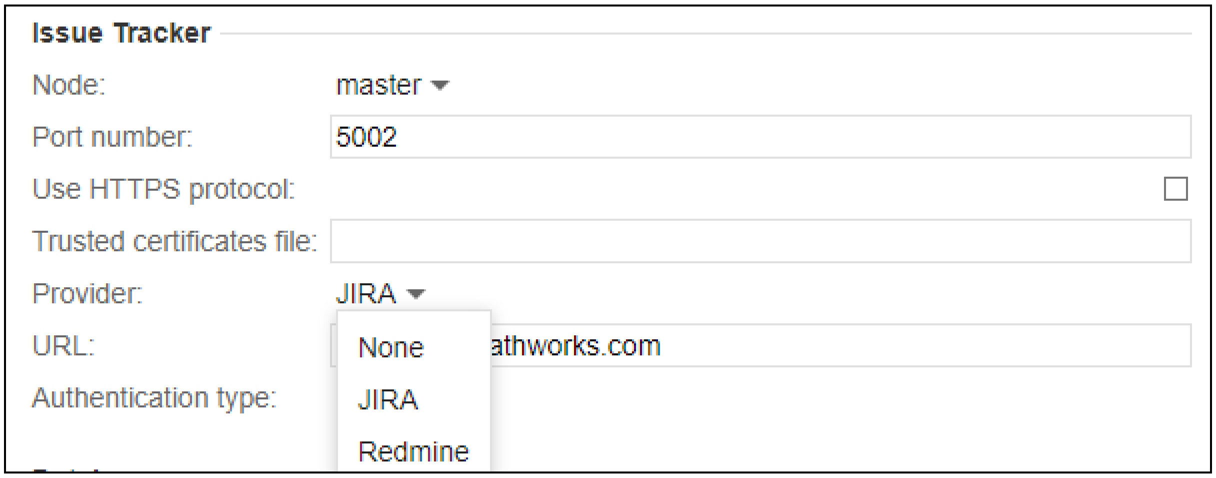
	Event	File	Scope
1	Function called by external code with input 's' **Possible input value causing defect: {.a=0, .b=-2}**	test.c	func()
2	Entering function 'func'	test.c	func()
3	Assignment to local variable 'j'	test.c	func()
4	Assignment to parameter 's'	test.c	func()
5	Assignment to local variable 'j'	test.c	func()
6	Integer division by zero	test.c	func()

```
    int a;
    int b;

} S2;

int func(S2 s)
{
    int i;
    int j = 1;
    s.a += 3;
    j = j - s.b;

        i = 1024 / (j - s.a);

    return i;
}
```

You can use the example values to fix defects in your code that are due to specific system input values.

Installation and Configuration: New 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.



Issue Tracker

Node:	master
Port number:	5002
Use HTTPS protocol:	☐
Trusted certificates file:	
Provider:	JIRA
URL:	athworks.com
Authentication type:	

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 `polypace` 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

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 Bug Finder™ Access™. Previously, you loaded all LDAP users listed under the **LDAP base** that you specified when you started Polyspace Bug Finder 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.

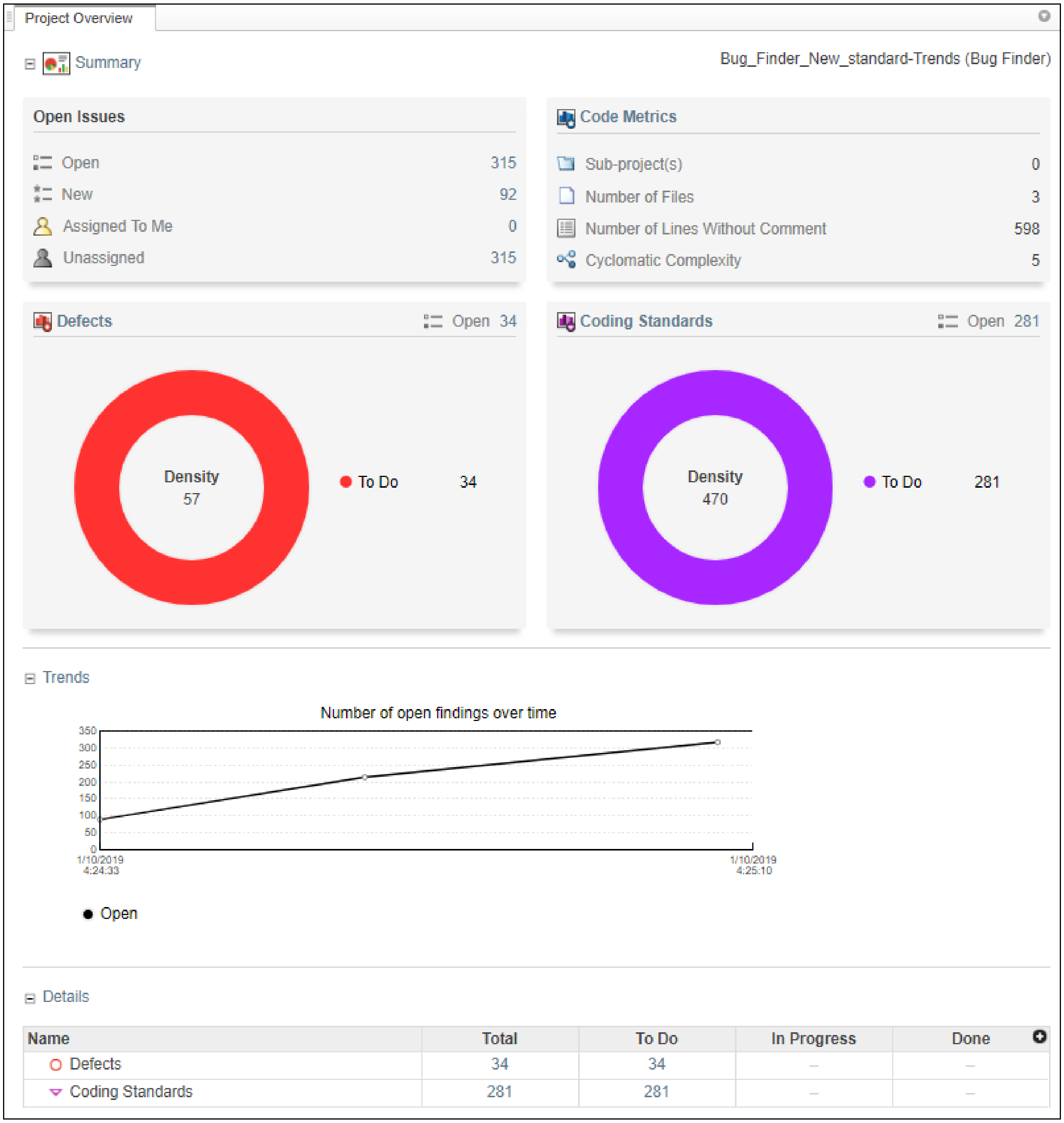
R2019a

Version: 2.0

New Features

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 Bug Finder Access **DASHBOARD**. When an analysis run is uploaded to the Polyspace Access database, the dashboard updates to give a snapshot of the findings, including a progress trend for number of findings compared to previous runs.


Project Overview

Summary

Bug_Finder_New_standard-Trends (Bug Finder)

Open Issues

Open	315
New	92
Assigned To Me	0
Unassigned	315

Code Metrics

Sub-project(s)	0
Number of Files	3
Number of Lines Without Comment	598
Cyclomatic Complexity	5

Defects — Open 34

Density 57 — ● To Do 34

Coding Standards — Open 281

Density 470 — ● To Do 281

Trends

Details

Name	Total	To Do	In Progress	Done
Defects	34	34	–	–
Coding Standards	281	281	–	–

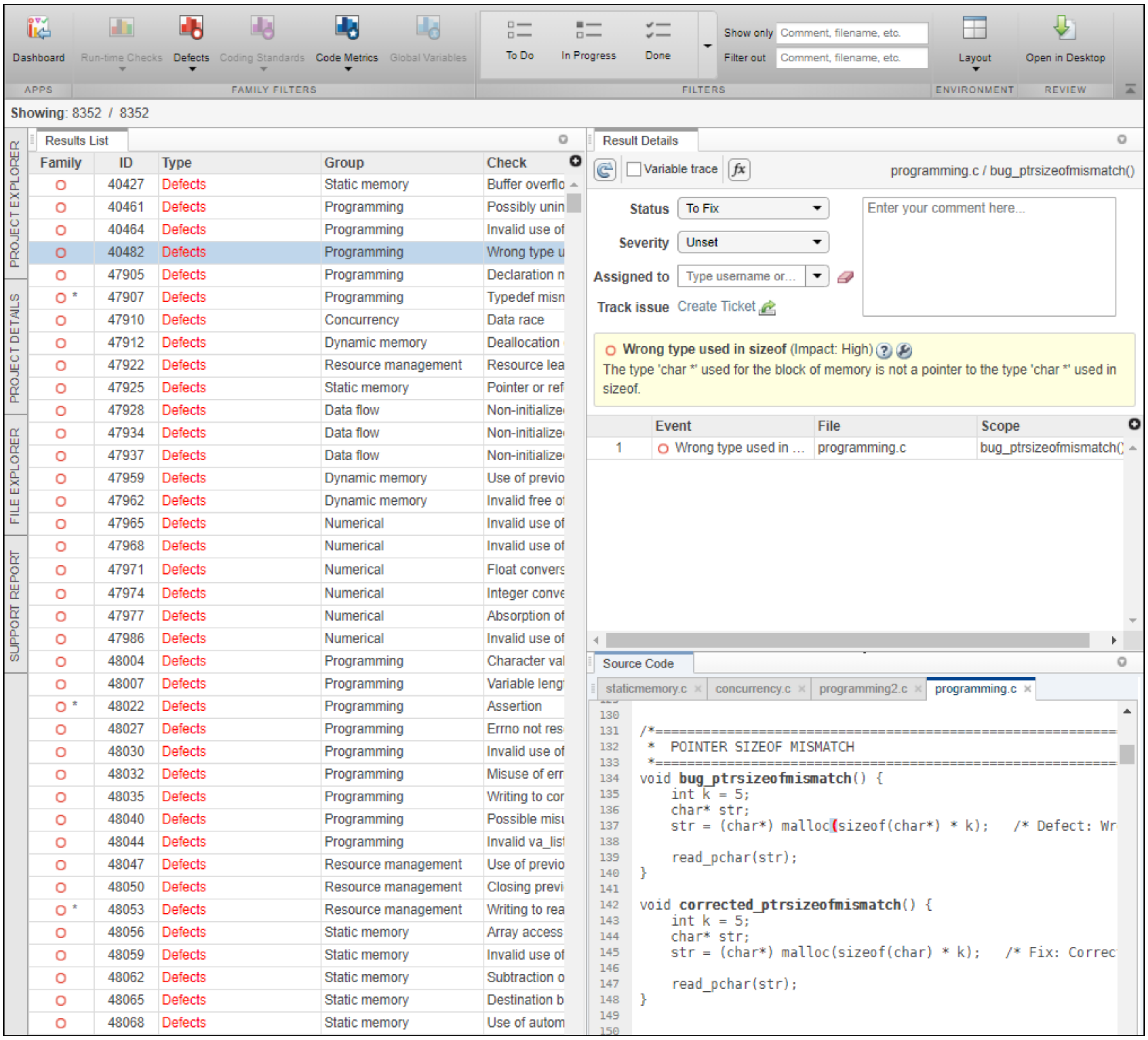
Additional Benefits:

- *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, move all the projects 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 Bug Finder Access license and the appropriate credentials can view the dashboard from their web browser.

Collaborative Review Support: Review Polyspace Bug Finder 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 Bug Finder Access **REVIEW** web interface. You do not need to install a Polyspace product on your machine to open and review analysis results.



Show only: Comment, filename, etc.
Filter out: Comment, filename, etc.

Showing: 8352 / 8352

Family	ID	Type	Group	Check
○	40427	Defects	Static memory	Buffer overflo
○	40461	Defects	Programming	Possibly unin
○	40464	Defects	Programming	Invalid use of
○	40482	Defects	Programming	Wrong type u
○	47905	Defects	Programming	Declaration n
○ *	47907	Defects	Programming	Typedef misn
○	47910	Defects	Concurrency	Data race
○	47912	Defects	Dynamic memory	Deallocation
○	47922	Defects	Resource management	Resource lea
○	47925	Defects	Static memory	Pointer or ref
○	47928	Defects	Data flow	Non-initialize
○	47934	Defects	Data flow	Non-initialize
○	47937	Defects	Data flow	Non-initialize
○	47959	Defects	Dynamic memory	Use of previo
○	47962	Defects	Dynamic memory	Invalid free o
○	47965	Defects	Numerical	Invalid use of
○	47968	Defects	Numerical	Invalid use of
○	47971	Defects	Numerical	Float convers
○	47974	Defects	Numerical	Integer conve
○	47977	Defects	Numerical	Absorption of
○	47986	Defects	Numerical	Invalid use of
○	48004	Defects	Programming	Character val
○	48007	Defects	Programming	Variable leng
○ *	48022	Defects	Programming	Assertion
○	48027	Defects	Programming	Errno not res
○	48030	Defects	Programming	Invalid use of
○	48032	Defects	Programming	Misuse of err
○	48035	Defects	Programming	Writing to cor
○	48040	Defects	Programming	Possible misu
○	48044	Defects	Programming	Invalid va_list
○	48047	Defects	Resource management	Use of previo
○	48050	Defects	Resource management	Closing previ
○ *	48053	Defects	Resource management	Writing to rea
○	48056	Defects	Static memory	Array access
○	48059	Defects	Static memory	Invalid use of
○	48062	Defects	Static memory	Subtraction o
○	48065	Defects	Static memory	Destination b
○	48068	Defects	Static memory	Use of autom

programming.c / bug_ptrsizeofmismatch()

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

Enter your comment here...

○ **Wrong type used in sizeof** (Impact: High)
The type 'char *' used for the block of memory is not a pointer to the type 'char *' used in sizeof.

	Event	File	Scope
1	○ Wrong type used in ...	programming.c	bug_ptrsizeofmismatch()

staticmemory.c × | concurrency.c × | programming2.c × | programming.c ×

```
/*============================================================
 *  POINTER SIZEOF MISMATCH
 *============================================================
void bug_ptrsizeofmismatch() {
    int k = 5;
    char* str;
    str = (char*) malloc(sizeof(char*) * k);   /* Defect: Wr

    read_pchar(str);
}

void corrected_ptrsizeofmismatch() {
    int k = 5;
    char* str;
    str = (char*) malloc(sizeof(char) * k);   /* Fix: Correc

    read_pchar(str);
}
```

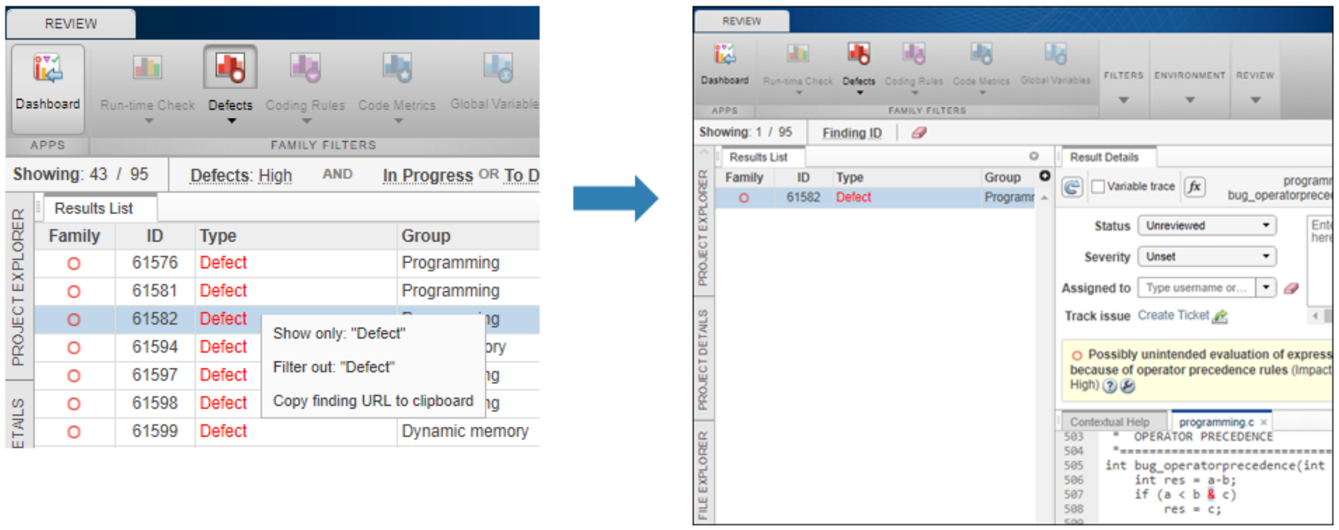
Additional Benefits:

- *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 Bug Finder Access license and the appropriate credentials can view the results from their web browser.

Collaborative Review Support: Share Polyspace Bug Finder results using web links

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



Family	ID	Type	Group
○	61576	Defect	Programming
○	61581	Defect	Programming
○	61582	Defect	ng
○	61594	Defect	ory
○	61597	Defect	ng
○	61598	Defect	ng
○	61599	Defect	Dynamic memory

Family	ID	Type	Group
○	61582	Defect	Programr

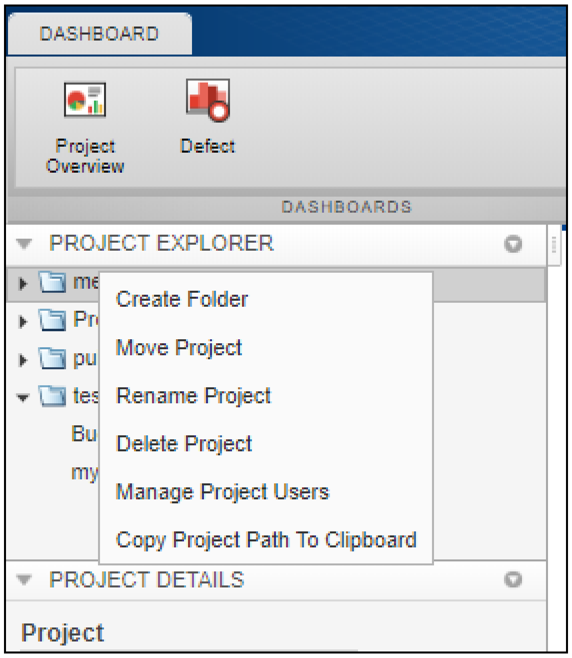
Status: Unreviewed
Severity: Unset
Assigned to: Type username or...
Track issue: Create Ticket

○ Possibly unintended evaluation of express because of operator precedence rules (Impact High)

```
 *   OPERATOR PRECEDENCE
 *==============================
int bug_operatorprecedence(int
    int res = a-b;
    if (a < b & c)
        res = c;
```

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

Summary:In R2019a, you can manage project users in Polyspace Bug Finder 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.

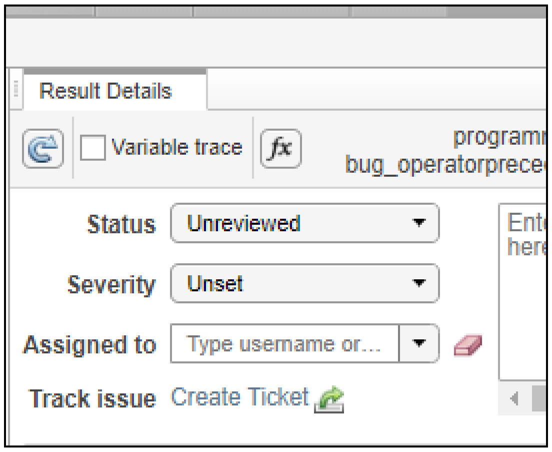


Additional Benefits:

- *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 Bug Finder results

Summary: In R2019a, Polyspace Bug Finder Access supports integration with the JIRA software. If you have an instance of the JIRA software, after you configure Polyspace Bug Finder 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. You can add the ticket to any existing JIRA project.



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