Cppcheck Status

SHARMAZANASHVILI Alexander
Georgian Technical University

TODUA Luka
Georgian Technical University

Current Status

- Our latest Cppcheck Scan report this week – 01.14.2020/01.21.2020

- We found 2 Defects

- From:
  - Peter Onyisi,
  - Apostolos Tsirigotis

STATUS	ID	SCAN DATE	FILE	MR DATE	LINE	DEFECT MESSAGE	AUTHOR	MAIL
	01212020001	01-21-2020	athena\InnerDetector\InDetMonitoring\TRTMonitoringRun3\src\TRTMonitoringRun3_Alg.cxx	01-20-2020	851	Array 'trackfound[2][64]' accessed at index trackfound[*][-9999], which is out of bounds.	Peter Onyisi	ponyisi@utexas.edu
	01212020002	01-21-2020	athena\MuonSpectrometer\MuonDetDescr\MuonTrackingGeometry\src\MuonInertMaterialBuilder.cxx	01-15-2020	802, 806, 818	Using iterator to local container 'constituents' that may be invalid.	Apostolos Tsirigotis	apostolos.tsirigotis@cern.ch

- We generated 2 Jira tickets

On Jira we have feedbacks from authors. For example,

on Jira ticket ATLASSQ-171 which was assigned

To Ruth Pottgen, he didn't know how this defect occurred so,

we had to add author of original Merge

Request Benedict Tobias Winter.

There were discussions between them to solve this issue.

Issue is still in progress.

Choosing Checking Platform

- Currently we are doing Cppcheck Scan on windows 10. because windows 10 always has latest updates of Cppcheck.

- On different operating systems such as linux based or windows. Some of them don't get latest updates as fast as windows 10.

- For the moment:
  - Latest ubuntu 19.10 has Cppcheck version 1.88;
  - CentOS 7 has Cppcheck version 1.88 and 1.90(only accessable via testing repository package - unstable);
  - On Windows we have Cppcheck version 1.89(stable) and latest 1.90;

- Difference between version 1.88 and 1.89/1.90 is that, 1.88 is using C++14 for checking, Where 1.89/1.90 is using C++17.

- Now we are using Cppcheck version 1.89 on Windows 10. With version 1.90 there was some problems.

Automatization Status

- In past we created automatization steps for Cppcheck scan report.

- This is steps are operated by one shell file,

- which is capable of:
  - cloning/pulling Athena repository from Gitlab
  - Scaning this repository with Cppcheck
  - Generating Defects in xml format
  - In defects file adding information such as:
    - Authors name
    - Authors email
    - Marge request date

- This steps are done without user interactions.

- For the moment:
  - we are working on transforming xml file into .html table.

- For the future:
  - Automatic creation of Jira Tickets with this .html table.