Seamless Static Analysis with Cppcheck

From IDE to CI and Code Review

DANIEL MARJAMÄKI

Agenda

- Motivation to integrate static analysis
  - Static analysis is not used enough
  - Use Cppcheck to avoid serious vulnerabilities
- Cppcheck Integration
  - IDE
  - CI + locally
  - Other tools, i.e. code review
- Cppcheck + Clang-tidy

Motivation

Not enough people use static analysis. Why?

Cppcheck Debian results

- We run open source cppcheck on debian source code
  - 1000's of warnings related to undefined behavior
  - Proof that many projects are not using cppcheck
  - It is getting better

Why don’t people use static analysis?

- Usability
  - Configuration takes a bit of effort
  - Many warnings

Unchecked legacy code - lots of warnings

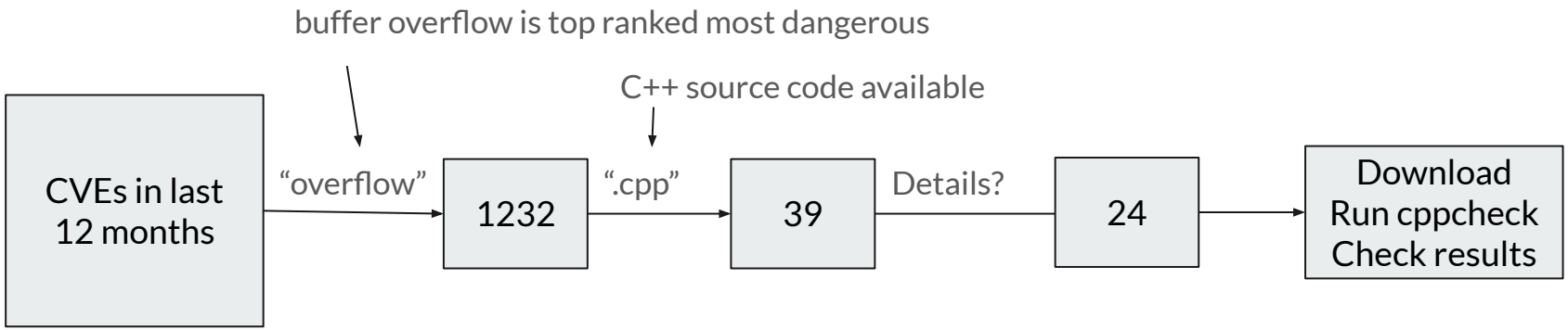
- If you turn on all checkers at once => lots of warnings
  - Many are not serious bugs
  - There is danger to fix them
- I suggest that you only turn on a few checkers
  - Only the most critical - definite UB

Why should you use Cppcheck

A look at security vulnerabilities

Cppcheck CVE coverage

- Investigation made in march this year.
  - How many (percentage) vulnerabilities are found by Cppcheck



CVE coverage

CVE	Found	Kind	Info
CVE-2023-35949	Yes	format string	Found: invalidscanf
CVE-2023-35950	Yes	format string	Found: invalidscanf
CVE-2023-35951	Yes	format string	Found: invalidscanf
CVE-2023-35952	Yes	format string	Found: invalidscanf
CVE-2023-35953	Yes	format string	Found: invalidscanf
CVE-2024-0051	Premium	iterator	Found: bughuntingIteratorIncrement
CVE-2024-25580		integer overflow	integer overflow not found yet
CVE-2024-30806	Premium	pointer	Found: bughuntingBufferOverflow
CVE-2024-31002	Premium	pointer	Found: bughuntingBufferOverflow
CVE-2024-31003	Premium	pointer	Found: bughuntingBufferOverflow
CVE-2024-31580		container	Not found yet
CVE-2024-33781	Premium	pointer	Found: bughuntingBufferOverflow
CVE-2024-33782		string not zero terminated	Not found yet
CVE-2024-34408		container	Not found yet
CVE-2024-37310	Premium	integer overflow, pointer	integer overflow not found yet, dangerous pointer usage found
CVE-2024-38952	Yes	array	Found: invalidscanf
CVE-2024-39129	Yes	format string	Found: invalidscanf
CVE-2024-40658		pointer	Not found yet
CVE-2024-43091		integer overflow	integer overflow not found yet
CVE-2024-43097		integer overflow	integer overflow not found yet
CVE-2024-43767		pointer	Not found, solution: do not use raw pointer
CVE-2024-43768		integer overflow	integer overflow not found yet
CVE-2024-46426	Premium	integer overflow, pointer	integer overflow not found yet, dangerous pointer usage found
CVE-2025-25943		buffer	Not found yet

Kind: format string (6, 25% of 24)

```
char header[1000];
const std::string OFF("OFF");
const std::string NOFF("NOFF");
const std::string COFF("COFF");
if(fscanf(off_file,"%s\n",header)!=1
```

(there are 6169 more such warnings, we have 1 open FP ticket)

Integrating Cppcheck

Integrating static analysis

- IDE : The earlier you get warnings the faster and easier it is to fix
  - Visual Studio Code - many
  - Clion - cppcheck plugin
  - Eclipse - cppcheclipse
  - Visual Studio - yes
  - Etc
- CI, locally
  - Enforcing that cppcheck is happy
- Other tools, i.e. code review

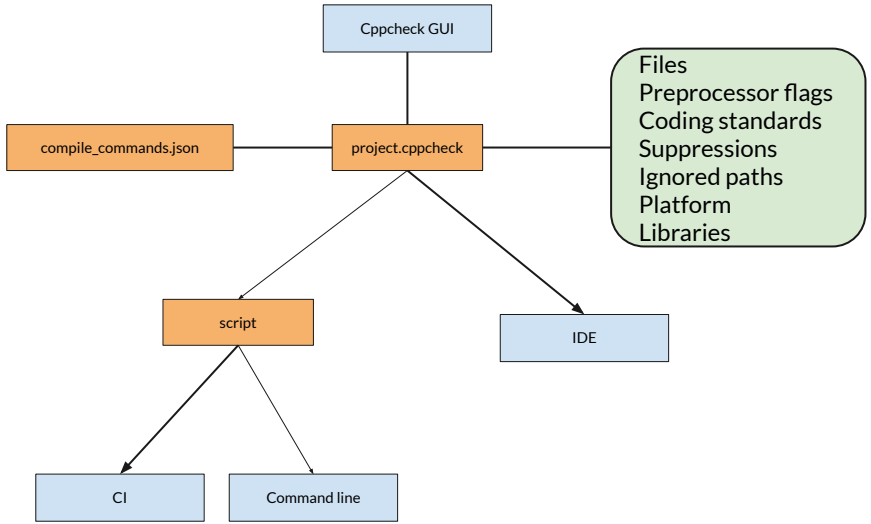
integration: we actively work on it

In last year:

- Added SARIF output for integration into various tools
  - Github
  - Gitlab
  - Sonarqube
  - etc
- Recently added --file-filter=+ option to simplify IDE integration
- Cppcheclipse
  - We are now maintaining this eclipse plugin
    - Thank you Konrad Windszus
  - Made it active and working again

More improvements to come..

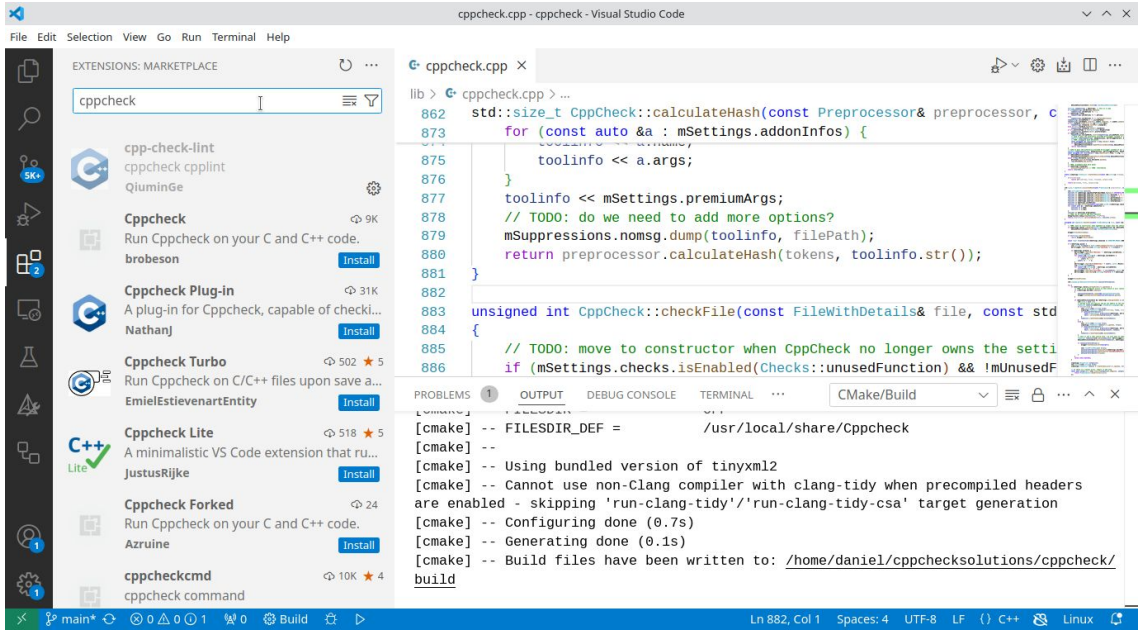
Using same configuration in IDE/CI/command line



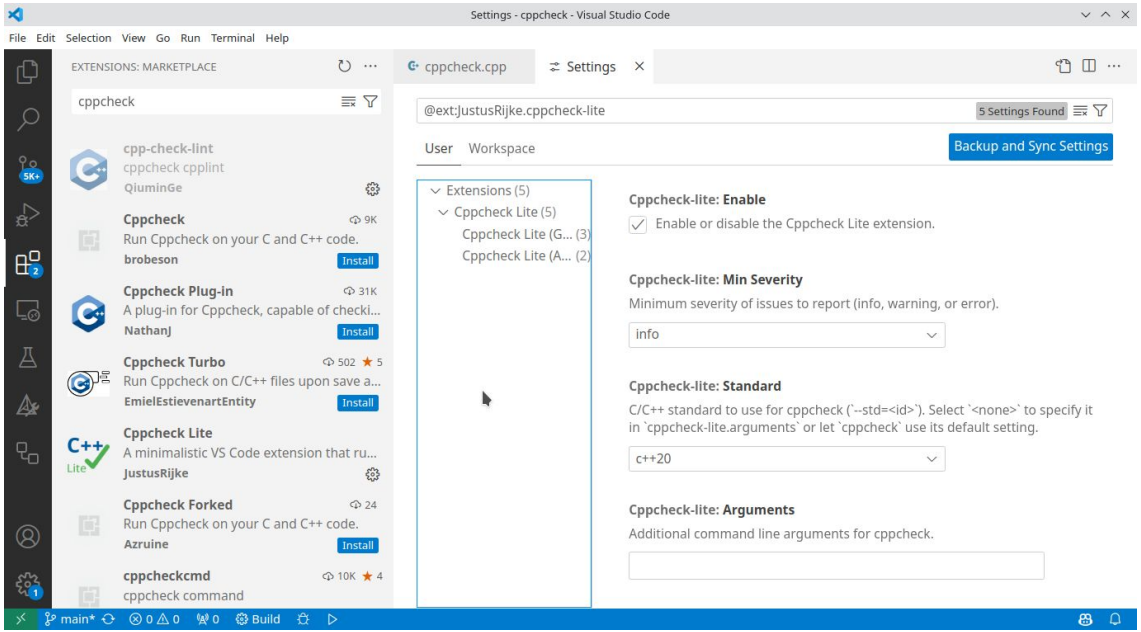
Using a cppcheck project file

- Command line + CI:
  - Straight forward: --project=project.cppcheck
- IDE: Many IDE plugins do not support project files directly
  - Configure extra arguments:
    - --project=project.cppcheck
    - --file-filter=+
    - =>plugin will execute such command: `cppcheck` --project=project.cppcheck --file-filter=+ `src/file.cpp`
      - project.cppcheck is loaded
      - src/file.cpp is checked using settings from project file, other files in the project are not checked

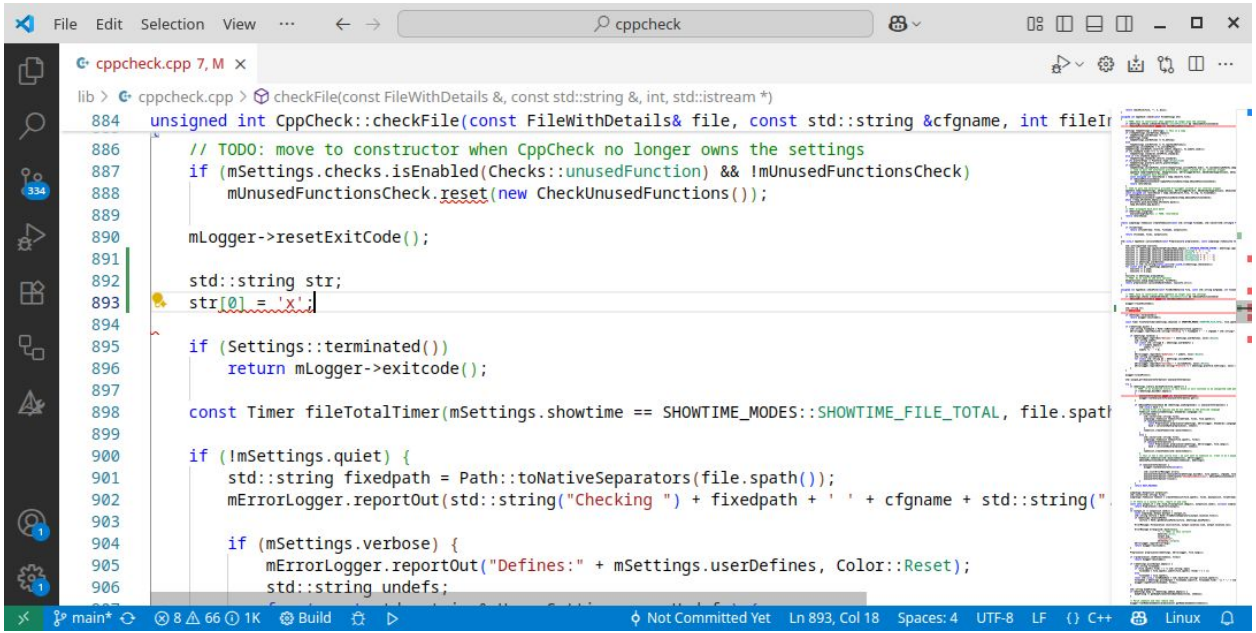
Vscode - install cppcheck plugin



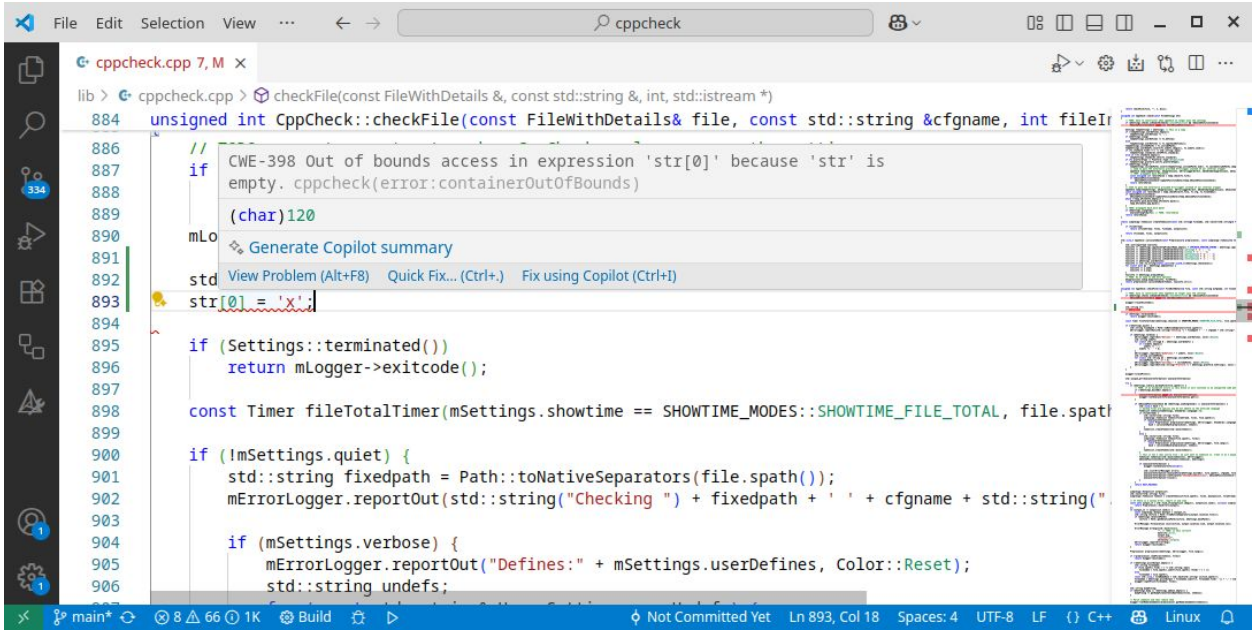
Vscode - configuration



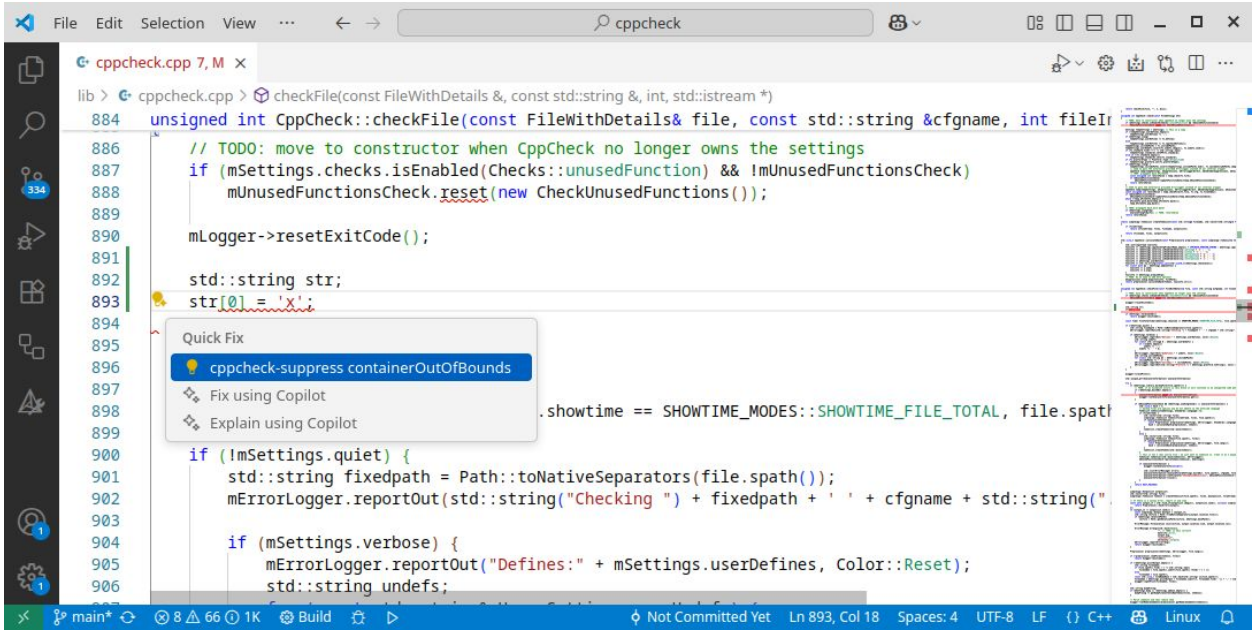
Vscode - results as you type or when you save



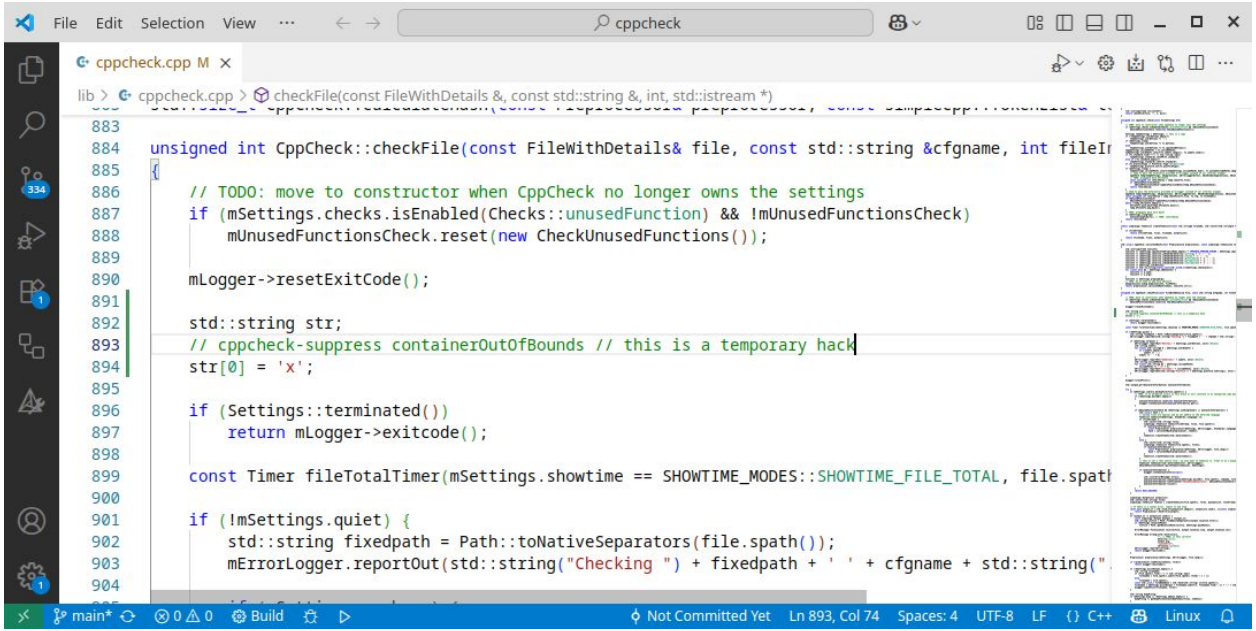
Vscode - hoover to see warning



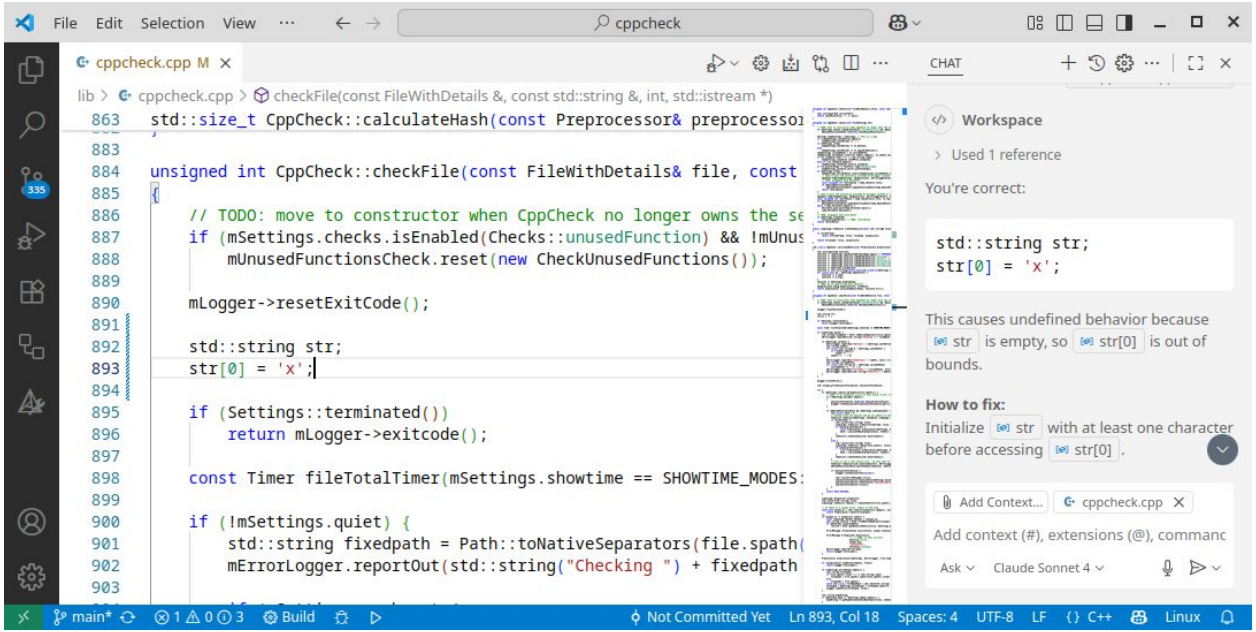
Vscode - lightbulb options



Vscode - suppression



Vscode - sometimes copilot can explain/fix



Integrate cppcheck results in various tools

- SARIF ; I think this is the “future” way to export static analysis results into various tools
  - Unified interface for all your static analyzers.

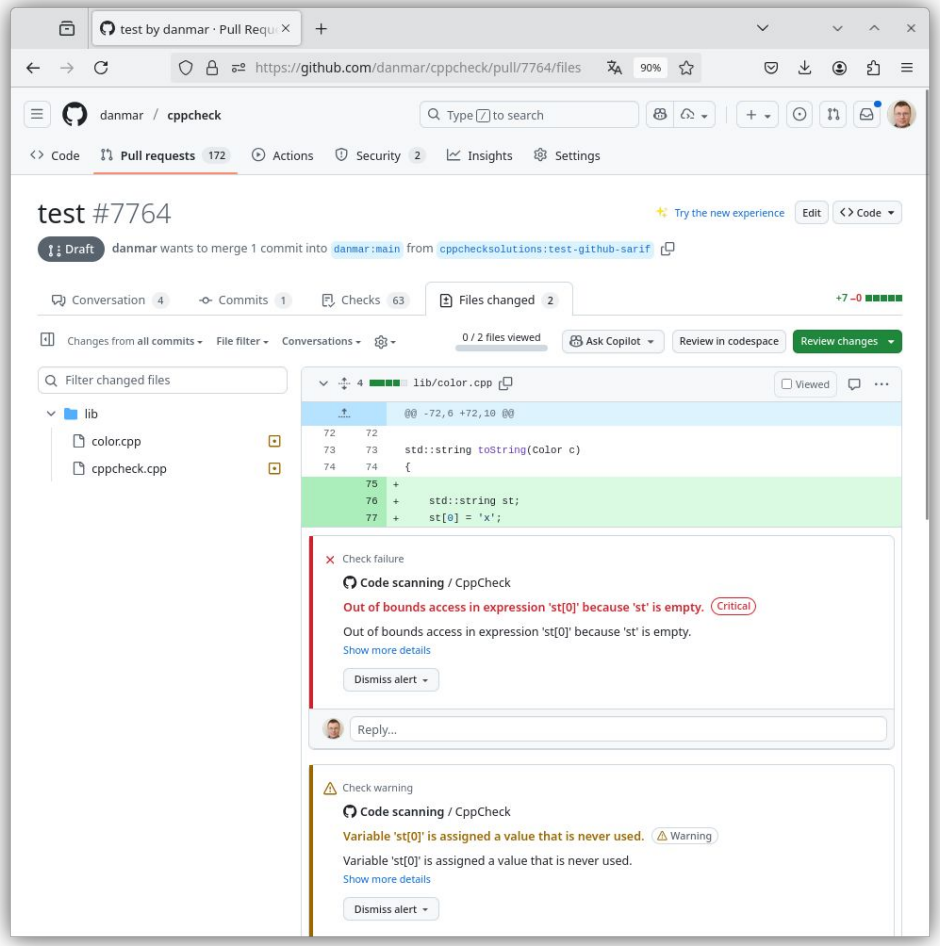
SARIF - github

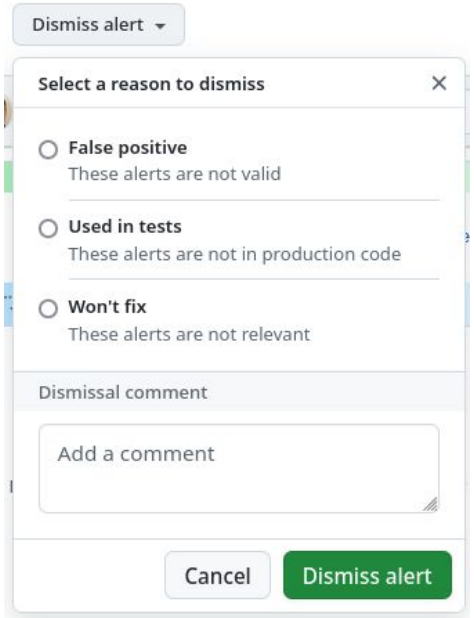
Simple github action snippet to run cppcheck on project and upload results:

```
- name: Run Cppcheck
  run: |
    cppcheck --project=test.cppcheck --output-format=sarif 2> results.sarif

- name: Upload report
  uses: github/codeql-action/upload-sarif@v3
  with:
    sarif_file: results.sarif
    category: cppcheck
```

SARIF - github





Cppcheck + clang-tidy

Use other tools also

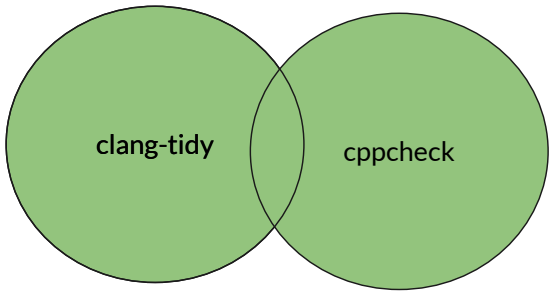
- It is a good idea to use several tools
- There is a common complaint to using several static analysis tools:
    - It means we must have several reports/plugins/scripts

Cppcheck and clang-tidy

I am a Cppcheck guy and I recommend that you use clang-tidy also

cppcheck and clang-tidy complement each other:

- Catch different bugs
- Different heuristics to catch a bug



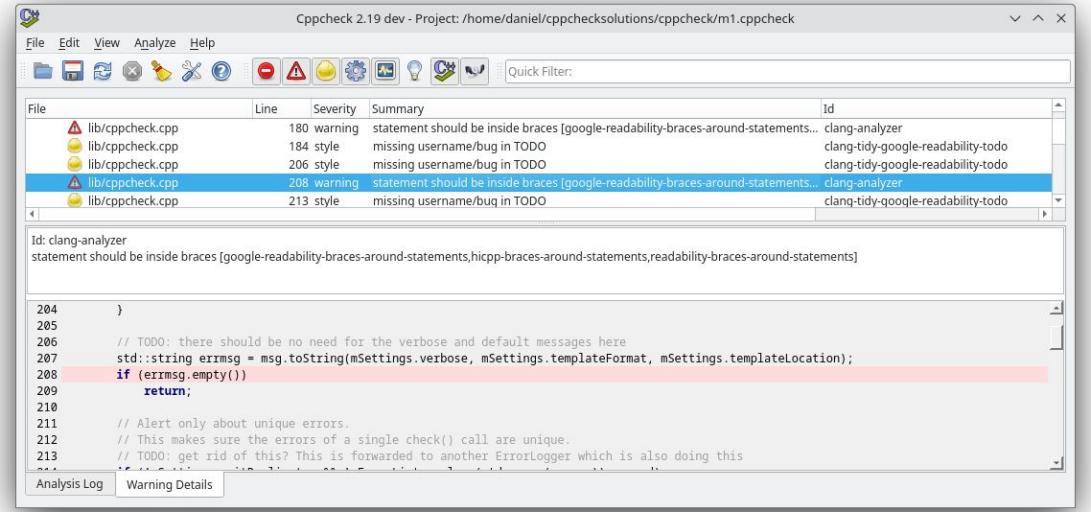
Configuration

- Provide a compile_commands.json
- Activate clang-tidy:
  - GUI: click on option “clang-tidy” in project settings
  - Command line: --clang-tidy

Cppcheck features for clang-tidy

- Output templates; reformatting warnings into for instance csv
- GUI
- SARIF
- Reuse same plugin/scripts
  - 1 IDE plugin
  - 1 CI plugin
  - 1 script to generate report
- Suppressions
- Multithreaded analysis
- Incremental analysis

Clang-tidy – GUI



Summary

We have talked about:

- Everybody should use static analysis
- How to integrate Cppcheck
- Cppcheck + clang-tidy