PA193 - Secure coding principles and practices

LAB: Static analysis of source code

Petr Švenda *svenda@fi.muni.cz* *@rngsec*
Centre for Research on Cryptography and Security, Masaryk University

CROCS
Centre for Research on
Cryptography and Security

Disclaimer

- The slides for this seminar (and part of the lecture) are taken from newly created lecture for PV080.
  - (you will also see pv080 on screenshots)
- If you already absolved this course, try to enjoy it again ☺
- But new content was created only this autumn and most of you already absolved PV080 long before that (and GitHub introduced decent support for static analysis only in Summer 2020)
- (for next years, I will update accordingly)

Idea of the seminar

- Prepare repo with vulnerable code (IS->buggycode.zip)
- Enable automatic static analysis via GitHub Actions
  - Several providers of analysis environment (custom or standard tools)
- Trigger by commit, investigate warnings/errors found
- Fix it, review again
- (Analyze your own homework before submission)
- Warning: Code scanning Actions are relatively new to GitHub, they may be glitches, UI bugs and tool failures

Basic analysis of C/C++ source code with various tools

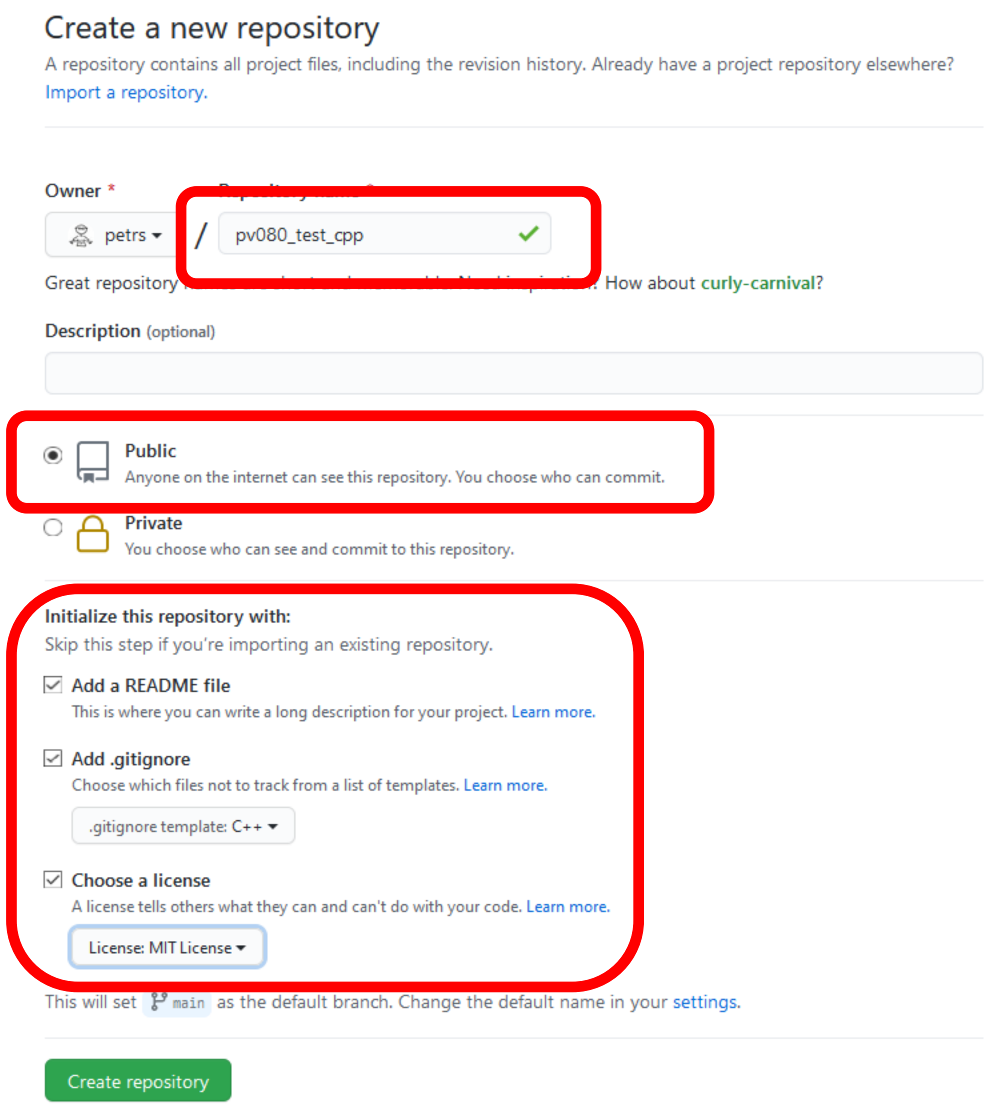
CODE SCANNING WITH GITHUB + ACTIONS + CODACY

Steps

1. Create repo on GitHub
2. Enable code analysis
3. Clone repo locally
4. Insert code with vulnerability, commit and push
5. Investigate results of analysis
6. Fix selected issue, rerun analysis
7. Repeat from step 5.

Create repo on GitHub

- Online at github.com
- Make repo public
  - GitHub Actions are free only for public ones
- Add readme, .gitignore, license
  - Generally good practice
- For start, don't mix languages
  - Put code of single lang in repo (e.g, c++)
  - Makes automatic analysis more difficult
  - E.g., 'pv080_test_cpp' & 'pv080_test_java'



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * Repository name *

petrs / pv080_test_cpp ✓

Great repository names are short and memorable. Need inspiration? How about curly-carnival?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☑ Add a README file
This is where you can write a long description for your project. Learn more.

☑ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

.gitignore template: C++

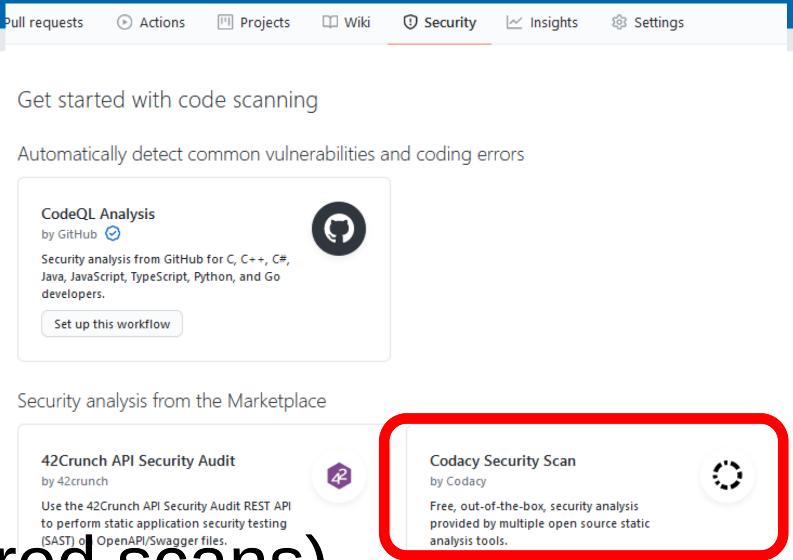
☑ Choose a license
A license tells others what they can and can't do with your code. Learn more.

License: MIT License

This will set main as the default branch. Change the default name in your settings.

Enable code scanning actions

- Online at github.com
- *Github→Repo*→Security→Set up code scanning
- Select Codacy Security Scan (scroll down in offered scans)
  - ‘Set up this workflow’ button



Get started with code scanning

Automatically detect common vulnerabilities and coding errors

CodeQL Analysis
by GitHub

Security analysis from GitHub for C, C++, C#, Java, JavaScript, TypeScript, Python, and Go developers.

Set up this workflow

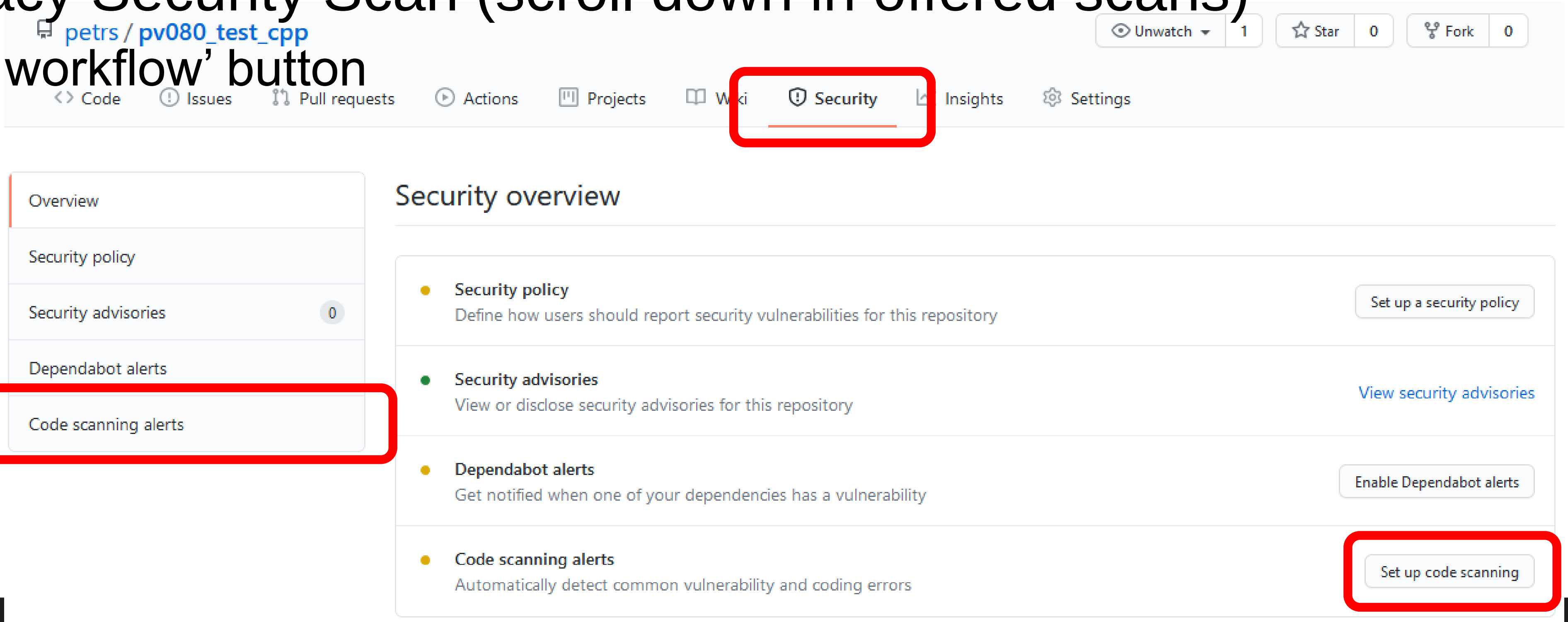
Security analysis from the Marketplace

42Crunch API Security Audit
by 42crunch

Use the 42Crunch API Security Audit REST API to perform static application security testing (SAST) of OpenAPI/Swagger files.

Codacy Security Scan
by Codacy

Free, out-of-the-box, security analysis provided by multiple open source static analysis tools.



Security overview

Security policy
Define how users should report security vulnerabilities for this repository
Set up a security policy

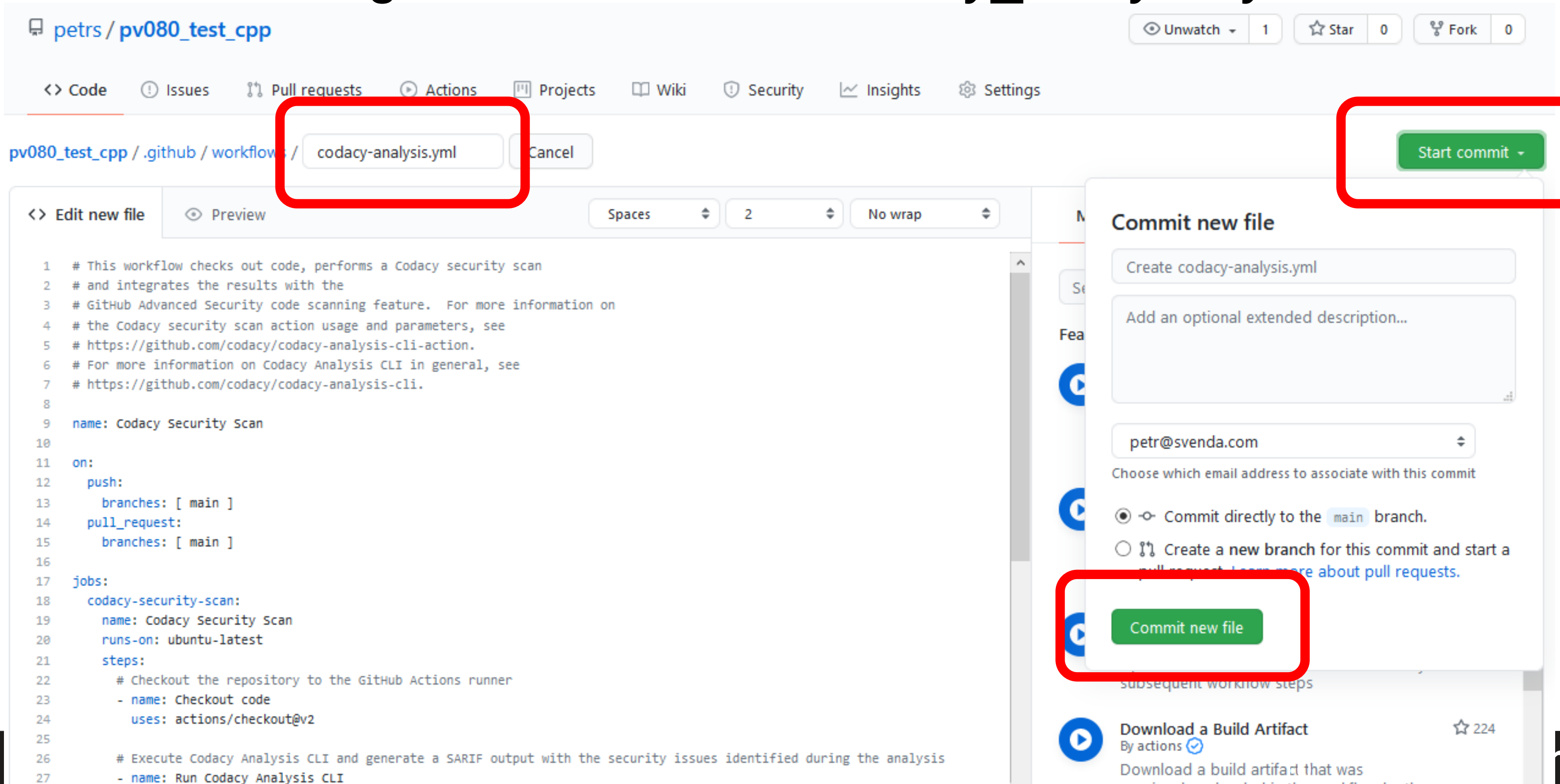
Security advisories
View or disclose security advisories for this repository
View security advisories

Dependabot alerts
Get notified when one of your dependencies has a vulnerability
Enable Dependabot alerts

Code scanning alerts
Automatically detect common vulnerability and coding errors
Set up code scanning

Commit configuration file for Codacy scan

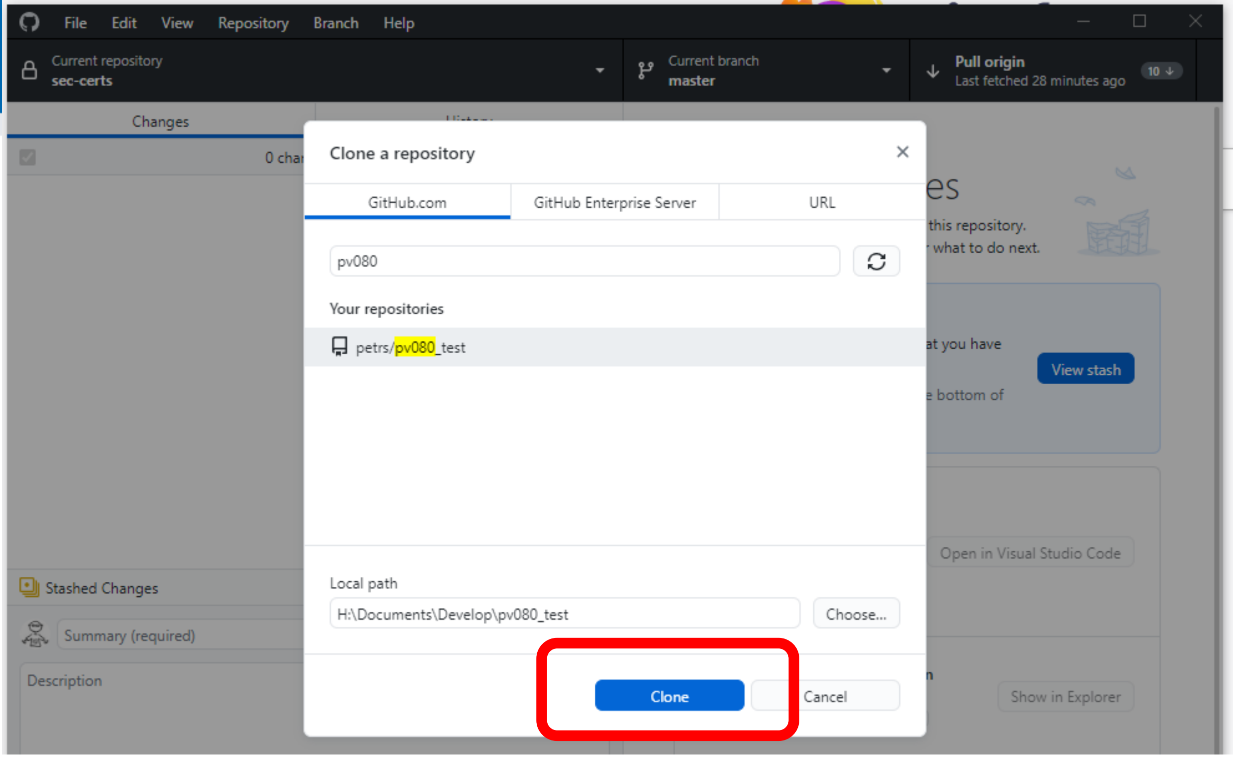
- No changes required to codacy_analysis.yml
  - Start commit →Commit new file
  - Can be found at /.github/workflows/ codacy_analysis.yml for later edits


petrs / pv080_test_cpp
Unwatch 1
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
pv080_test_cpp / .github / workflows / codacy-analysis.yml
Cancel
Start commit
Edit new file
Preview
Spaces
2
No wrap
Commit new file
Create codacy-analysis.yml
Add an optional extended description...
petr@svenda.com
Choose which email address to associate with this commit
Commit directly to the main branch.
Create a new branch for this commit and start a pull request. Learn more about pull requests.
Commit new file
Download a Build Artifact
By actions
224
Download a build artifact that was

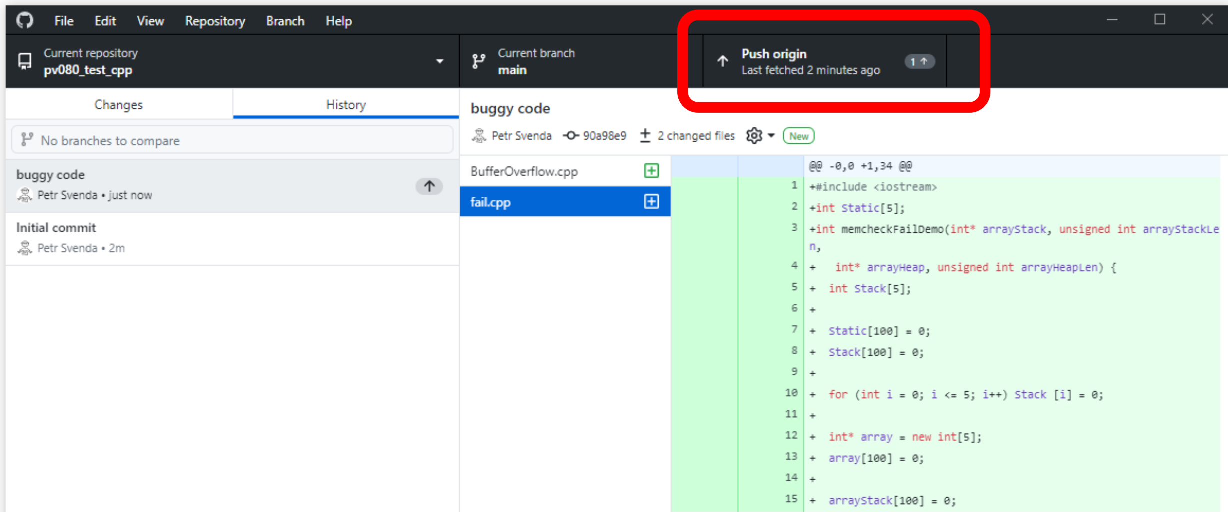
```
# This workflow checks out code, performs a Codacy security scan
# and integrates the results with the
# GitHub Advanced Security code scanning feature.  For more information on
# the Codacy security scan action usage and parameters, see
# https://github.com/codacy/codacy-analysis-cli-action.
# For more information on Codacy Analysis CLI in general, see
# https://github.com/codacy/codacy-analysis-cli.

name: Codacy Security Scan

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  codacy-security-scan:
    name: Codacy Security Scan
    runs-on: ubuntu-latest
    steps:
      # Checkout the repository to the GitHub Actions runner
      - name: Checkout code
        uses: actions/checkout@v2

      # Execute Codacy Analysis CLI and generate a SARIF output with the security issues identified during the analysis
      - name: Run Codacy Analysis CLI
```

Prepare repo content

- Locally on your PC
- Clone repository on your PC
  - GitHub Desktop File→Clone
  - git checkout your_repository.git



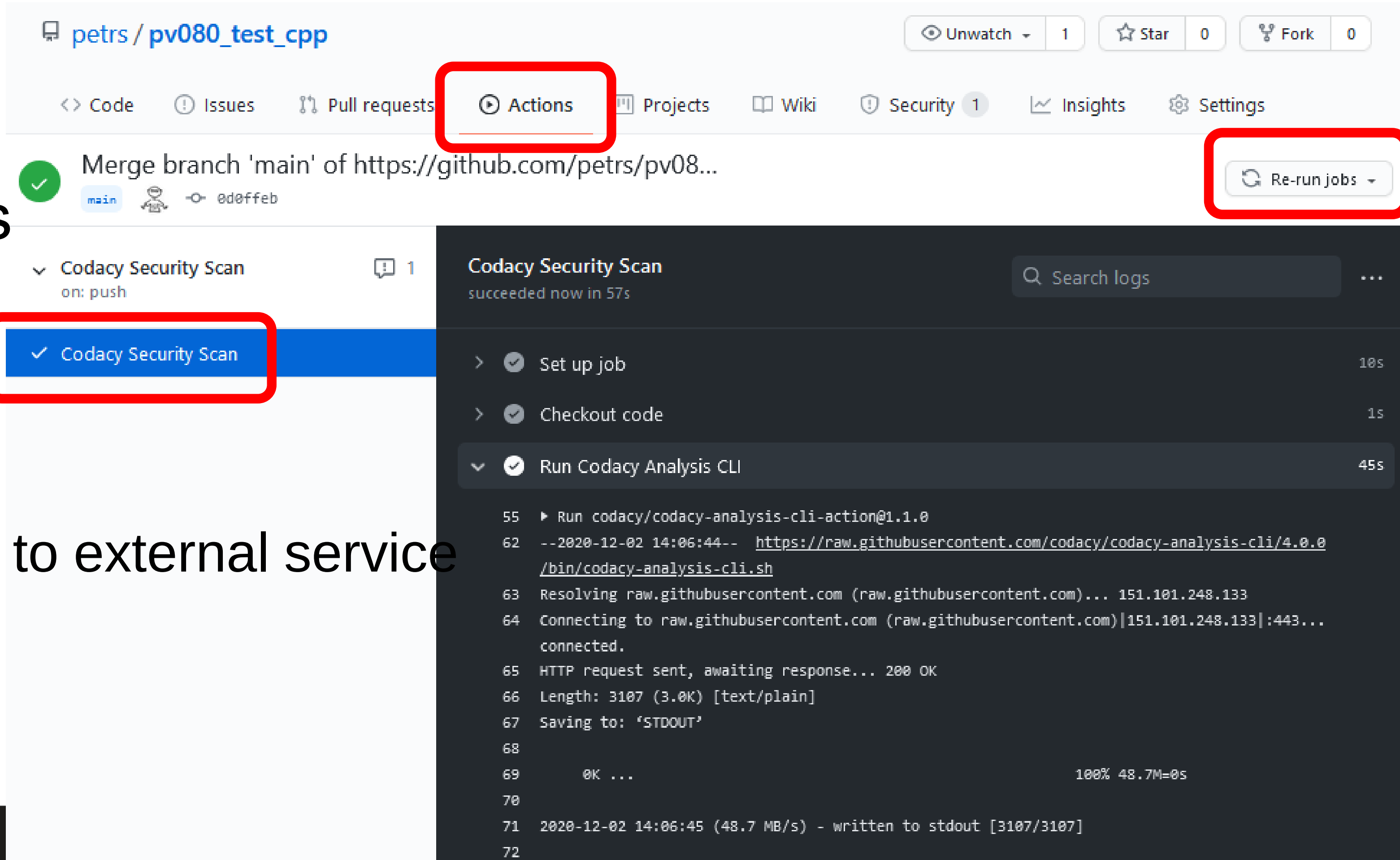
- Copy example buggy code into your repo and commit
  - IS → Study materials, buggycode.zip
  - Commit new files, push to repo (Push origin)



```
@@ -0,0 +1,34 @@
+#include <iostream>
+int Static[5];
+int memcheckFailDemo(int* arrayStack, unsigned int arrayStackLen,
+   int* arrayHeap, unsigned int arrayHeapLen) {
+  int Stack[5];
+
+  Static[100] = 0;
+  Stack[100] = 0;
+
+  for (int i = 0; i <= 5; i++) Stack [i] = 0;
+
+  int* array = new int[5];
+  array[100] = 0;
+
+  arrayStack[100] = 0;
```

Analyze results I.

- Observe scheduled, running and finished actions
- Online at github.com
- *Github→Repo*→Actions
- Re-run jobs if desired
  - Done on same commit!
  - Useful if Action failed due to external service



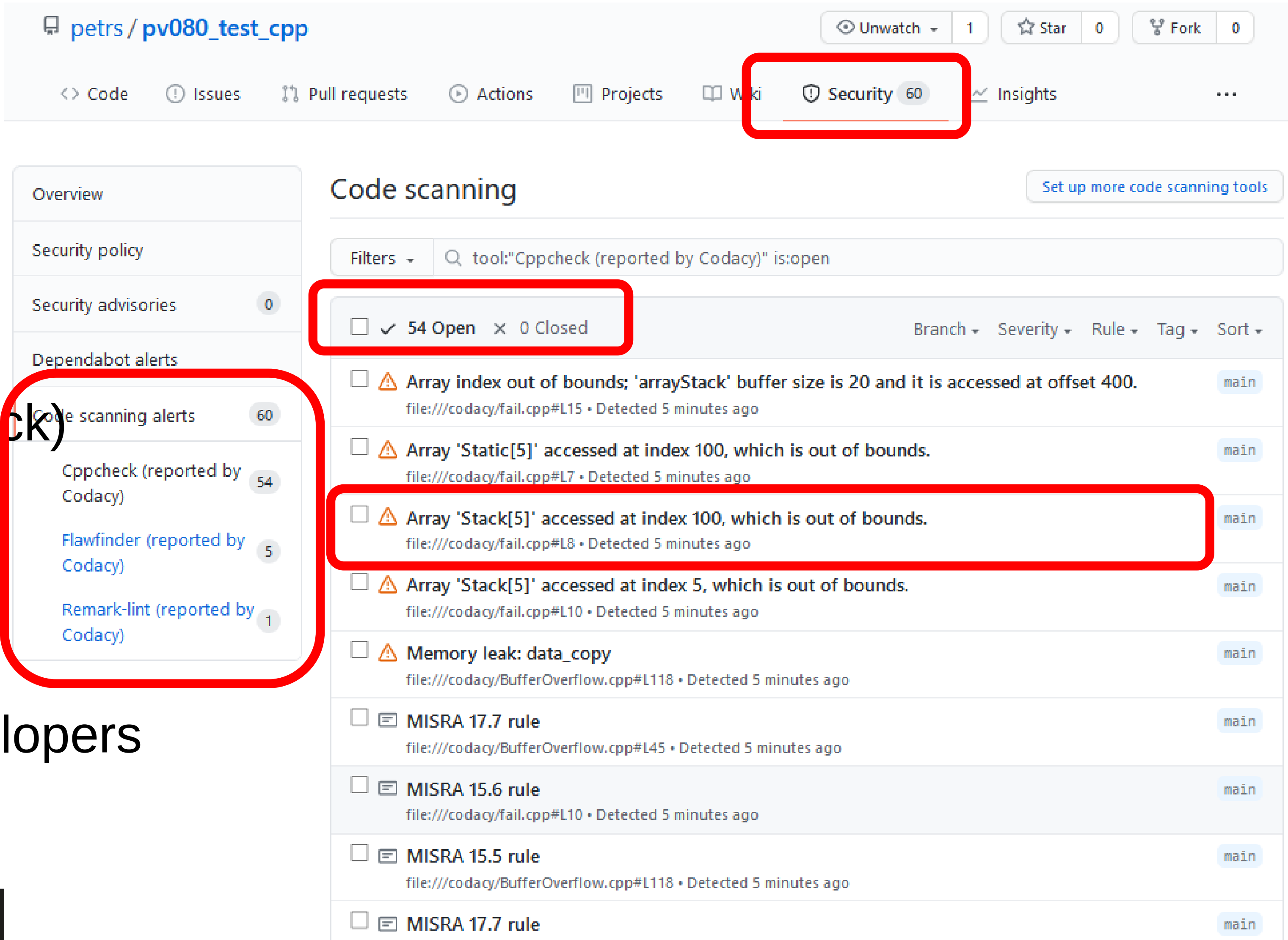
Merge branch 'main' of https://github.com/petrs/pv08...

Codacy Security Scan
succeeded now in 57s

```
▶ Run codacy/codacy-analysis-cli-action@1.1.0
--2020-12-02 14:06:44--  https://raw.githubusercontent.com/codacy/codacy-analysis-cli/4.0.0
/bin/codacy-analysis-cli.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.248.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.248.133|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 3107 (3.0K) [text/plain]
Saving to: 'STDOUT'

     0K ...                                                   100% 48.7M=0s

2020-12-02 14:06:45 (48.7 MB/s) - written to stdout [3107/3107]
```

Analyze results II.

- Online at github.com
- *Github→Repo*→Security
  - When actions are finished
- Code scanning alerts
  - Sorted by tool (e.g., Cppcheck)
- Shown similarly to Issues
  - Open, Closed
  - Can be filtered (severity…)
  - But visible only to repo developers


petrs / pv080_test_cpp

Unwatch 1 · Star 0 · Fork 0

Code · Issues · Pull requests · Actions · Projects · Wiki · Security 60 · Insights

Overview
Security policy
Security advisories 0
Dependabot alerts
Code scanning alerts 60
Cppcheck (reported by Codacy) 54
Flawfinder (reported by Codacy) 5
Remark-lint (reported by Codacy) 1

Code scanning

Set up more code scanning tools

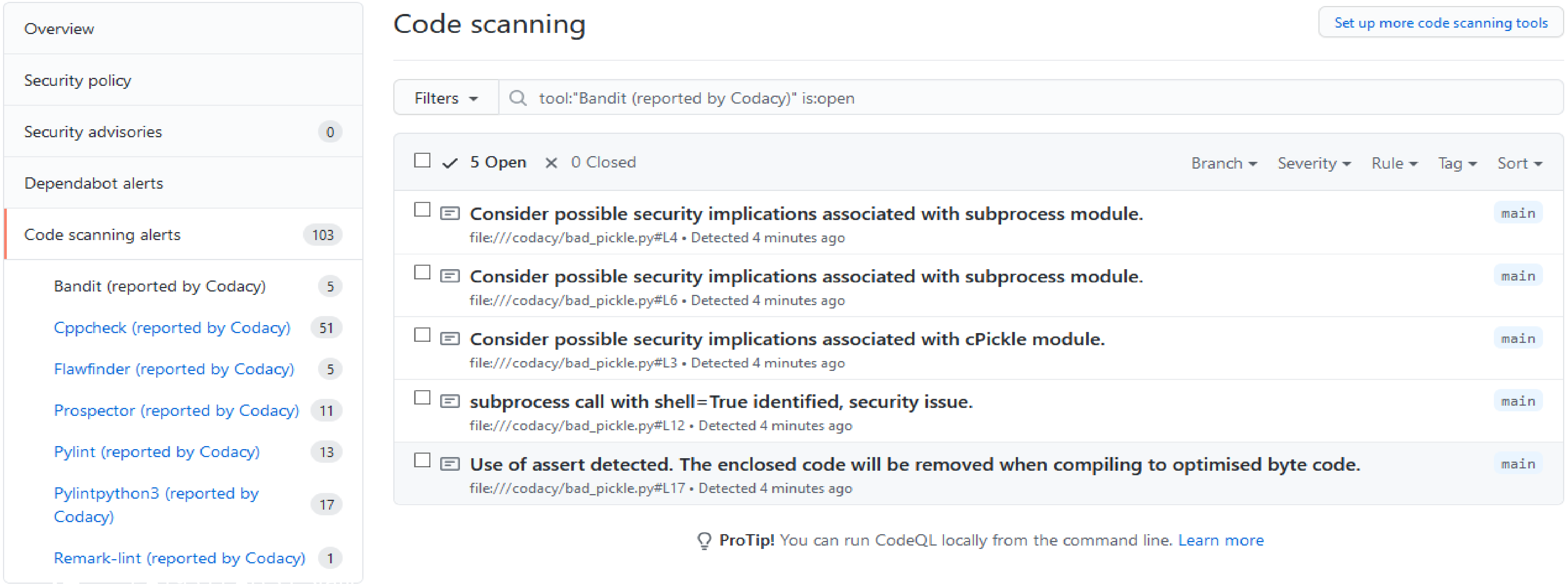
Filters · tool:"Cppcheck (reported by Codacy)" is:open

54 Open · 0 Closed · Branch · Severity · Rule · Tag · Sort

- Array index out of bounds; 'arrayStack' buffer size is 20 and it is accessed at offset 400. — file:///codacy/fail.cpp#L15 • Detected 5 minutes ago — main
- Array 'Static[5]' accessed at index 100, which is out of bounds. — file:///codacy/fail.cpp#L7 • Detected 5 minutes ago — main
- Array 'Stack[5]' accessed at index 100, which is out of bounds. — file:///codacy/fail.cpp#L8 • Detected 5 minutes ago — main
- Array 'Stack[5]' accessed at index 5, which is out of bounds. — file:///codacy/fail.cpp#L10 • Detected 5 minutes ago — main
- Memory leak: data_copy — file:///codacy/BufferOverflow.cpp#L118 • Detected 5 minutes ago — main
- MISRA 17.7 rule — file:///codacy/BufferOverflow.cpp#L45 • Detected 5 minutes ago — main
- MISRA 15.6 rule — file:///codacy/fail.cpp#L10 • Detected 5 minutes ago — main
- MISRA 15.5 rule — file:///codacy/BufferOverflow.cpp#L118 • Detected 5 minutes ago — main
- MISRA 17.7 rule — main

Update: 8.3.2021

- More bugs are reported currently (8.3.2021) than end of last year
  - More analysis tools were added in meantime (e.g., Bandit)



Overview	
Security policy	
Security advisories	0
Dependabot alerts	
Code scanning alerts	103
Bandit (reported by Codacy)	5
Cppcheck (reported by Codacy)	51
Flawfinder (reported by Codacy)	5
Prospector (reported by Codacy)	11
Pylint (reported by Codacy)	13
Pylintpython3 (reported by Codacy)	17
Remark-lint (reported by Codacy)	1

Code scanning

tool:"Bandit (reported by Codacy)" is:open

5 Open ✕ 0 Closed

- Consider possible security implications associated with subprocess module. `main`
  file:///codacy/bad_pickle.py#L4 • Detected 4 minutes ago
- Consider possible security implications associated with subprocess module. `main`
  file:///codacy/bad_pickle.py#L6 • Detected 4 minutes ago
- Consider possible security implications associated with cPickle module. `main`
  file:///codacy/bad_pickle.py#L3 • Detected 4 minutes ago
- subprocess call with shell=True identified, security issue. `main`
  file:///codacy/bad_pickle.py#L12 • Detected 4 minutes ago
- Use of assert detected. The enclosed code will be removed when compiling to optimised byte code. `main`
  file:///codacy/bad_pickle.py#L17 • Detected 4 minutes ago

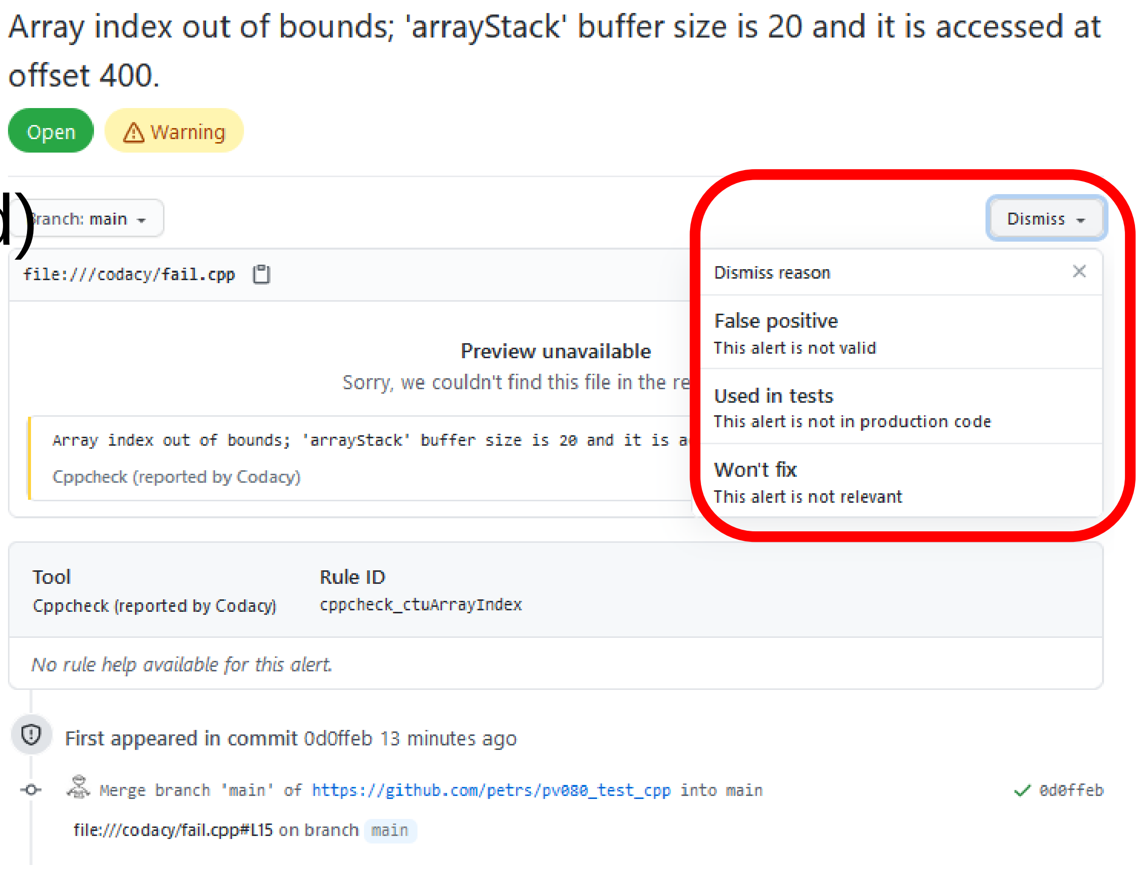
ProTip! You can run CodeQL locally from the command line. Learn more

Notes

- Standard Issues are used to report bugs or ask for / plan enhancements and new features (usually opened manually)
- Code scanning alerts are similarly treated, but opened automatically, visible only to developers
- Results from tool(s) are transformed to standardized ‘OASIS Static Analysis Results Interchange Format (SARIF) TC’, which GitHub can process, and display issues based on it

Analyze results III.

- Bug triage
  - atm, bug properties cannot be changed
  - (expect UI change in future)
- Can be dismissed (=> will not be fixed)
  - E.g., if False positive, not relevant…
  - Severity is set by original tools
    - Expect unification in future
  - Dismiss only bugs you are sure about!



Array index out of bounds; 'arrayStack' buffer size is 20 and it is accessed at offset 400.

file:///codacy/fail.cpp

Preview unavailable

Sorry, we couldn't find this file in the re

```
Array index out of bounds; 'arrayStack' buffer size is 20 and it is a
```

Cppcheck (reported by Codacy)

Dismiss reason

False positive
This alert is not valid

Used in tests
This alert is not in production code

Won't fix
This alert is not relevant

Tool	Rule ID
Cppcheck (reported by Codacy)	cppcheck_ctuArrayIndex

No rule help available for this alert.

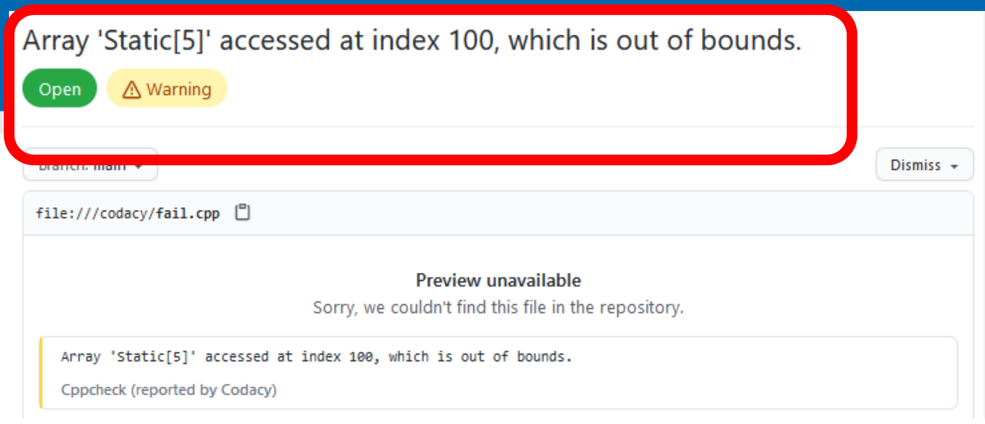
First appeared in commit 0d0ffeb 13 minutes ago

Merge branch 'main' of https://github.com/petrs/pv080_test_cpp into main ✓ 0d0ffeb

file:///codacy/fail.cpp#L15 on branch main

Fix bug(s)

- Locate reported bug in source code
  - (Note: for the moment, bug preview at Github is not working)
  - Use file and line number to locate (e.g., fail.cpp#L7 => line 7 in fail.cpp)
- Fix bug
  - E.g., Static[5]; → Static[101];
  - (Note: not proper fix, check length instead)
- Commit, Push
  - Will trigger analysis again
- Fixed issues are now in 'Closed' category
  - Introducing and fixing commit is visible in history



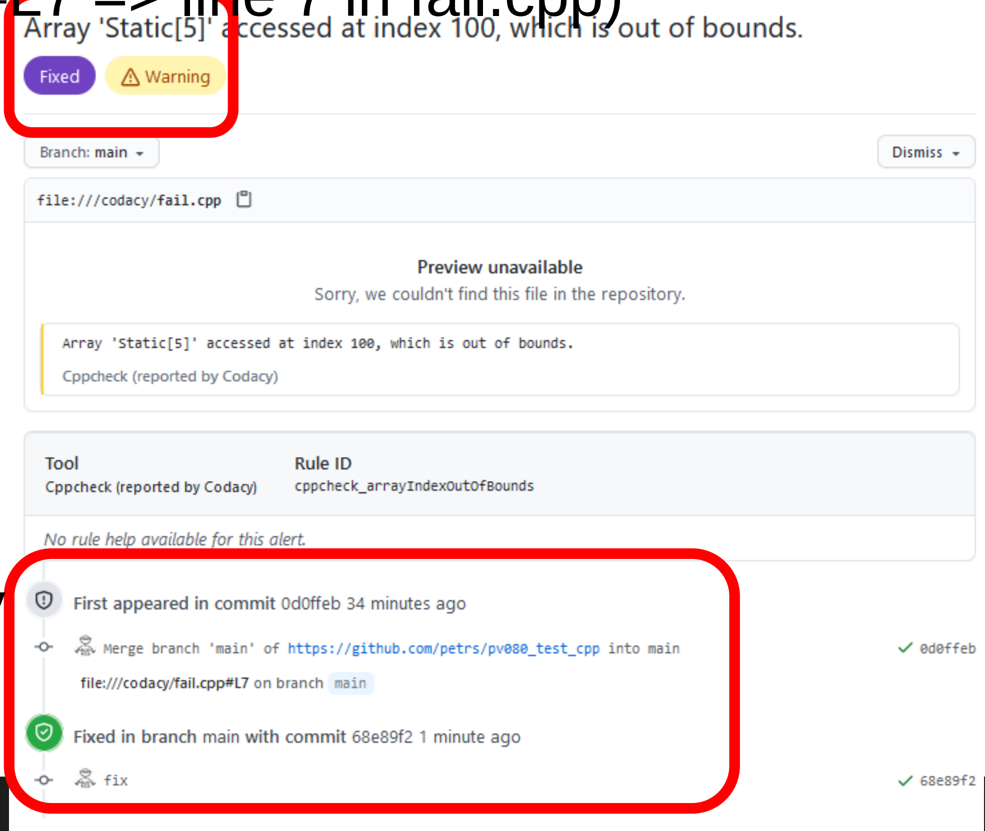
Array 'Static[5]' accessed at index 100, which is out of bounds.

file:///codacy/fail.cpp

Preview unavailable
Sorry, we couldn't find this file in the repository.

```
Array 'Static[5]' accessed at index 100, which is out of bounds.
```

Cppcheck (reported by Codacy)



Array 'Static[5]' accessed at index 100, which is out of bounds.

file:///codacy/fail.cpp

Preview unavailable
Sorry, we couldn't find this file in the repository.

```
Array 'Static[5]' accessed at index 100, which is out of bounds.
```

Cppcheck (reported by Codacy)

Tool	Rule ID
Cppcheck (reported by Codacy)	cppcheck_arrayIndexOutOfBounds

No rule help available for this alert.

First appeared in commit 0d0ffeb 34 minutes ago

Merge branch 'main' of https://github.com/petrs/pv080_test_cpp into main ✓ 0d0ffeb

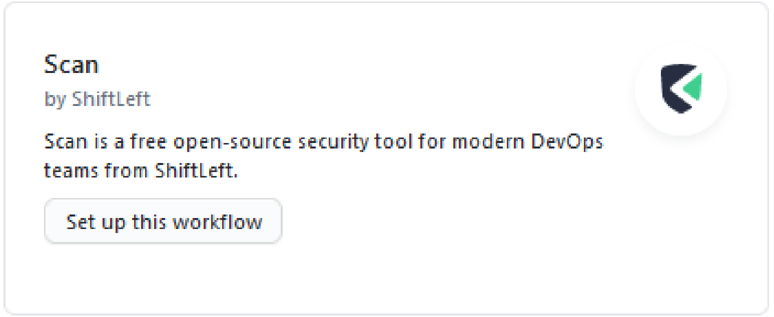
file:///codacy/fail.cpp#L7 on branch main

Fixed in branch main with commit 68e89f2 1 minute ago

fix ✓ 68e89f2

Scanning of python source code with

ADDITIONAL SCANNING OF PYTHON CODE WITH SHIFTLEFT SCANNER

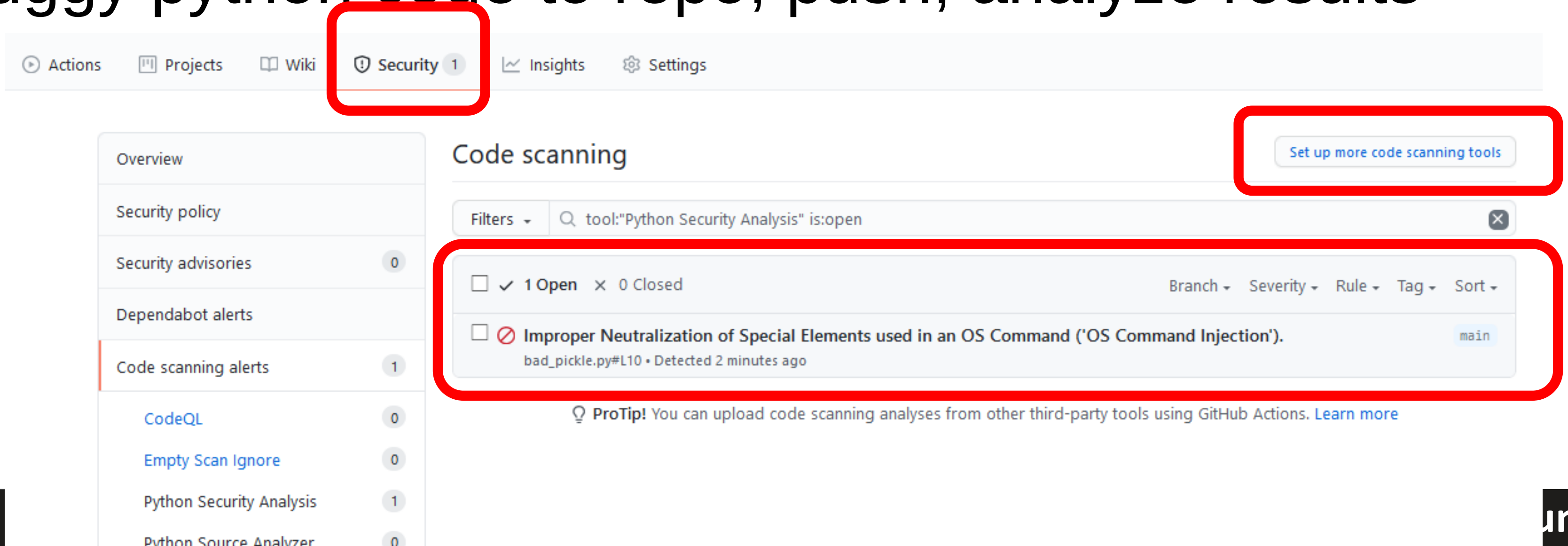


Scan
by ShiftLeft
Scan is a free open-source security tool for modern DevOps teams from ShiftLeft.

Set up this workflow

Setup ShiftLeft actions on repo

- Create new repository (or reuse previous), clone locally
- Enable code scanning actions
  – Set up more scanning tools – pick ‘Scan by ShiftLeft‘
- No need to change shiftleft-analysis.yml before commit
- Copy buggy python code to repo, push, analyze results



Code scanning

tool:"Python Security Analysis" is:open

✓ 1 Open ✕ 0 Closed

Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection'). main
bad_pickle.py#L10 • Detected 2 minutes ago

ProTip! You can upload code scanning analyses from other third-party tools using GitHub Actions. Learn more

Notes

- ShiftLeft scan requires no special configuration (same as Codacy)
- Will find additional bug in Python code
- Provides better explanation of bug (results from tools are likely to improve in future)

Bit more advanced setup, CodeQL code analysis, configurable build steps

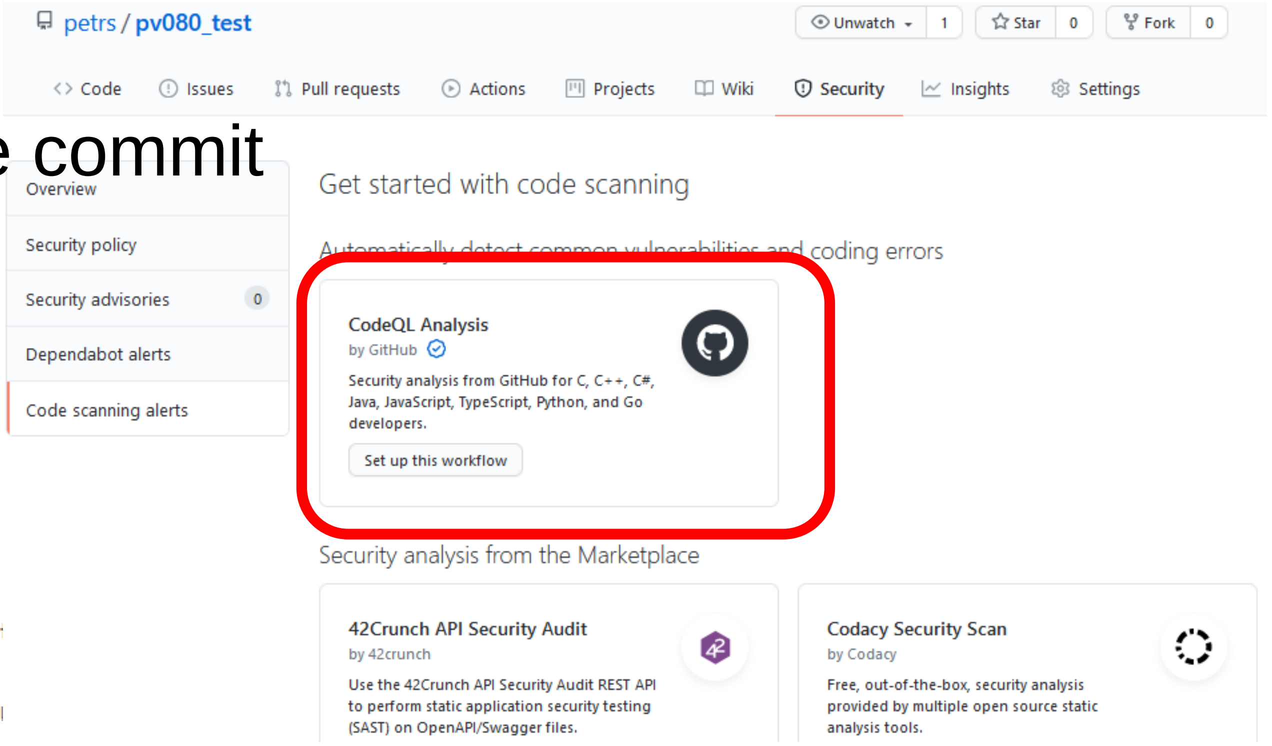
CODE SCANNING WITH GITHUB + ACTIONS + CODEQL

CodeQL basics

- Your source code → CodeQL code → rules executed on that canonical code
  - Adding support for new language (e.g., Go) => just convert Go source code to CodeQL canonical form and then use all already existing rules
- CodeQL uses own language to write analysis rules
  - Many existing security rules are already written, you don’t need to learn this language or write own rules to use it
- CodeQL is integrated in GitHub Actions or can be run for external CI
  - We will use integrated option
  - https://docs.github.com/en/free-pro-team@latest/github/finding-security-vulnerabilities-and-errors-in-your-code/enabling-code-scanning-for-a-repository
- Note: difference between dedicated tool (e.g., cppcheck) and CodeQL
  - Single tool for single language – detection rules must be written again for new lang
  - CodeQL – detection rules are written for canonical code, new lang requires only to write conversion between lang code and canonical code

Setup CodeQL actions on repository

- Create new repository (e.g., pv080_test_python), clone locally
- Enable code scanning actions
  - Pick CodeQL (instead of Codacy)
- Check codeql-analysis.yml before commit
  - Modify set of target languages
    - language: ['cpp']
- Copy buggy code to repo, push



petrs / pv080_test

Unwatch 1 | Star 0 | Fork 0

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

Overview
Security policy
Security advisories 0
Dependabot alerts
Code scanning alerts

Get started with code scanning

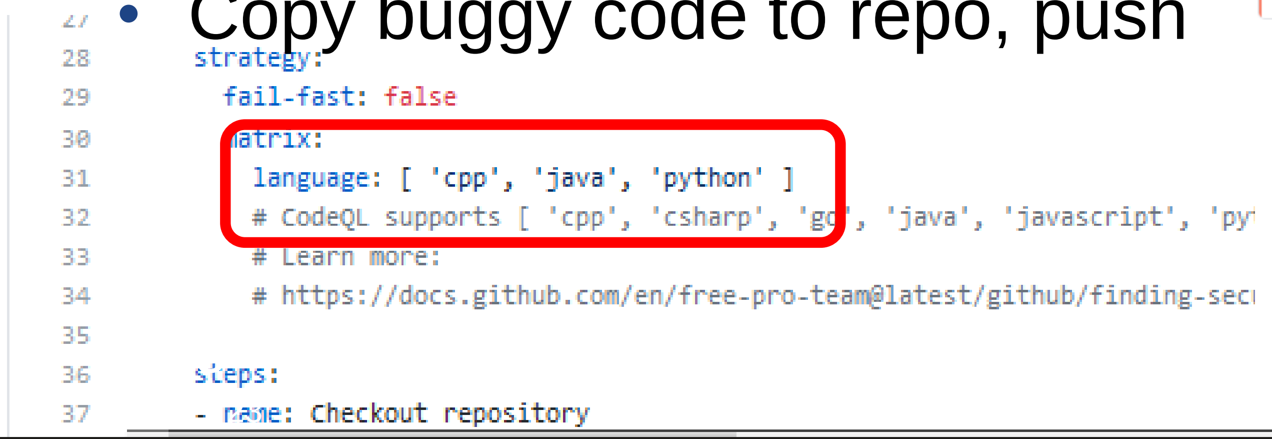
Automatically detect common vulnerabilities and coding errors

CodeQL Analysis
by GitHub
Security analysis from GitHub for C, C++, C#, Java, JavaScript, TypeScript, Python, and Go developers.
Set up this workflow

Security analysis from the Marketplace

42Crunch API Security Audit
by 42crunch
Use the 42Crunch API Security Audit REST API to perform static application security testing (SAST) on OpenAPI/Swagger files.

Codacy Security Scan
by Codacy
Free, out-of-the-box, security analysis provided by multiple open source static analysis tools.

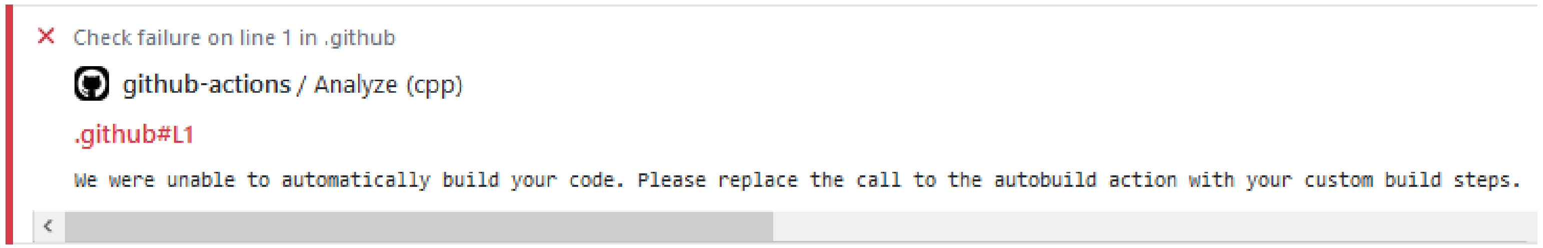


```
    strategy:
      fail-fast: false
      matrix:
        language: [ 'cpp', 'java', 'python' ]
        # CodeQL supports [ 'cpp', 'csharp', 'go', 'java', 'javascript', 'py
        # Learn more:
        # https://docs.github.com/en/free-pro-team@latest/github/finding-sec

    steps:
    - name: Checkout repository
```

CROCS

Fixing build for CodeQL

- CodeQL Action may fail with:



✕ Check failure on line 1 in .github

github-actions / Analyze (cpp)

.github#L1

```
We were unable to automatically build your code. Please replace the call to the autobuild action with your custom build steps.
```

- Reason
  - Analysis for some languages works on the compiled code/bytecode (e.g., Java)
  - Static analysis generally runs on unfinished code, but not always
  - One shall not commit broken code to repo anyway
- Fix: tell CodeQL how to build

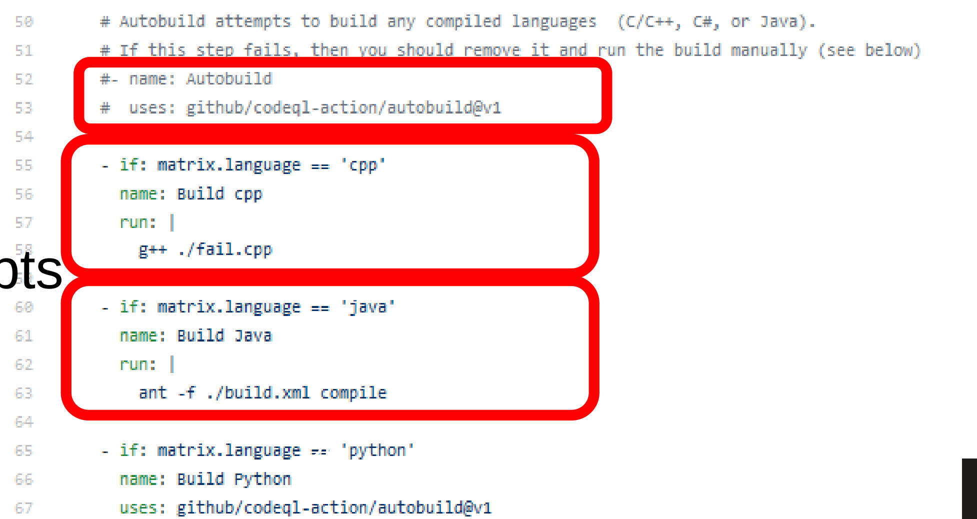
Fixing build for CodeQL I.

- GitHub CodeQL tries to compile your code
  - But how it knows how to compile your project?
- Autobuild feature is only heuristic (=> can be wrong, can fail)
  - https://docs.github.com/en/free-pro-team@latest/github/finding-security-vulnerabilities-and-errors-in-your-code/configuring-the-codeql-workflow-for-compiled-languages
  - Depends on CI operating system
  - Search for .sln or .vcxproj (MS Visual Studio), then call MSBuild.exe
  - Search for build.bat, build.cmd, and build.exe, then run it
  - Search for Makefile, then call make
  - Starts in repo root, then try in subdirectories…
- Tip: Start with simplest example, make it work, then make more complicated

Fixing build for CodeQL II.

- The solution depends on build system for your project
  - Make, gradle, ant, maven…
  - We will only discuss simple direct build with g++ and makefile
- Option 1: Makefile into repo root (g++ fail.cpp)
  - Feel free to use improved makefile scripts
  - Generally better solution than option 2
- Option 2: Direct specification in codeql-analysis.yml
  - Disable autobuild by commenting it out with #
  - Insert conditional statement based on language
    - Example here for cpp and java
    - Python is left with autobuild
  - More flexibility in configuration, more changes to scripts

```
main:
      g++ ./fail.cpp
```



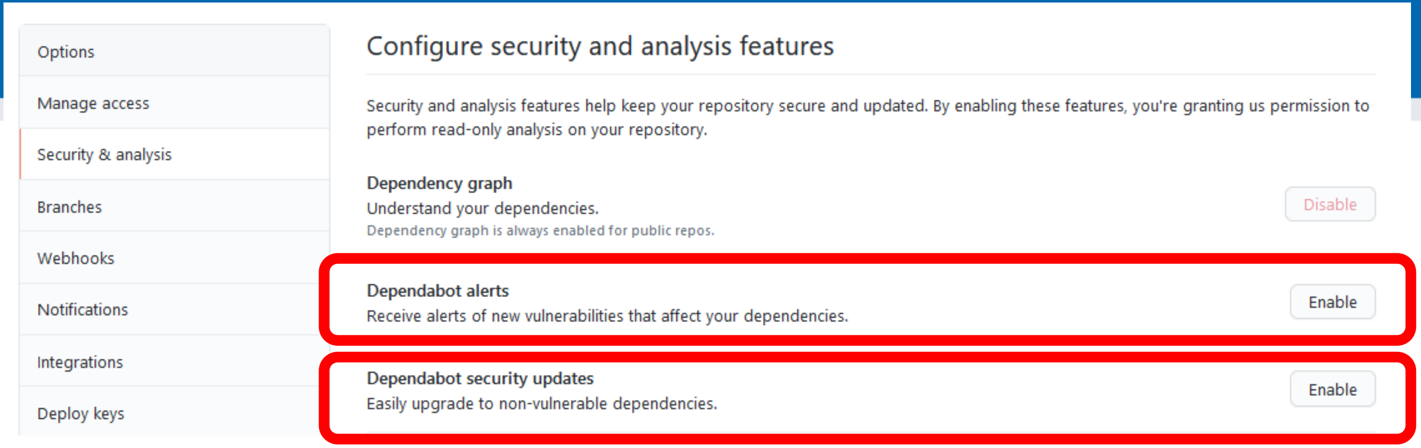
```
    # Autobuild attempts to build any compiled languages  (C/C++, C#, or Java).
    # If this step fails, then you should remove it and run the build manually (see below)
    #- name: Autobuild
    #  uses: github/codeql-action/autobuild@v1

    - if: matrix.language == 'cpp'
      name: Build cpp
      run: |
        g++ ./fail.cpp

    - if: matrix.language == 'java'
      name: Build Java
      run: |
        ant -f ./build.xml compile

    - if: matrix.language == 'python'
      name: Build Python
      uses: github/codeql-action/autobuild@v1
```

Setup Action to observe new vulnerabilities in your dependencies, notify you and even propose automatic patch

CHECKING SECURITY OF DEPENDENCIES GITHUB + DEPENDABOT

Enable dependabot

- Enable Dependabot alerts
  - You will receive notification about vulnerable dependency
- Enable Dependabot security updates
  - You will receive automatic pull requests fixing vulnerable dependency
  - Always analyze automatic pull requests for correctness



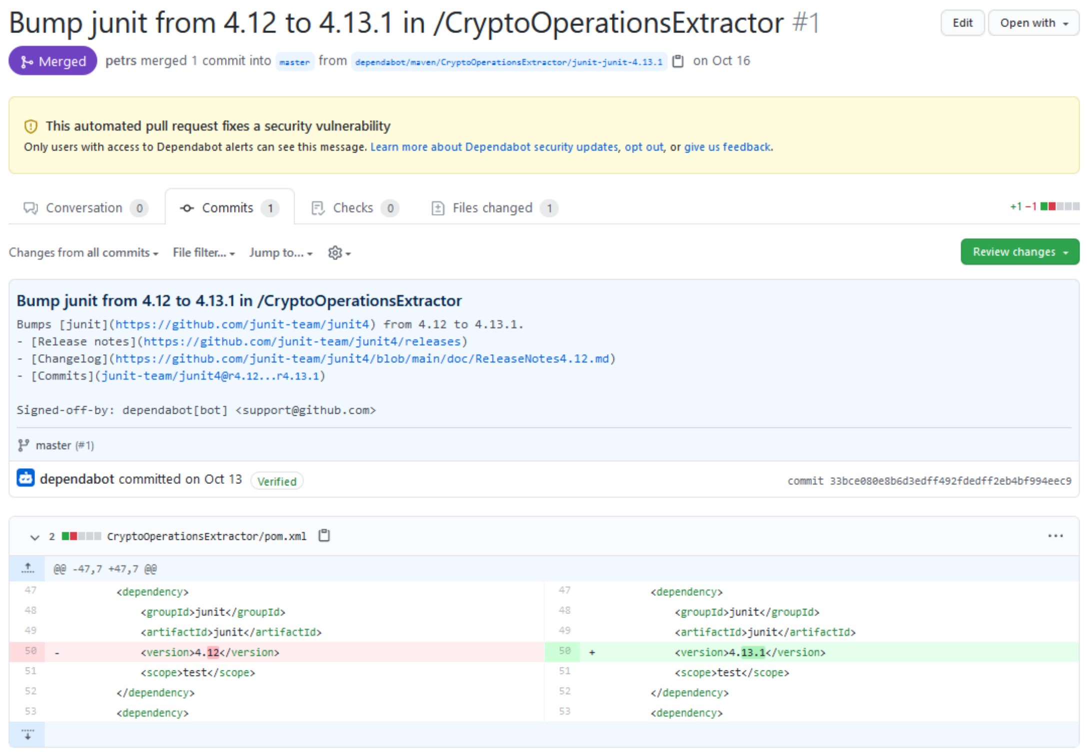
Configure security and analysis features

Security and analysis features help keep your repository secure and updated. By enabling these features, you're granting us permission to perform read-only analysis on your repository.

Dependency graph
Understand your dependencies.
Dependency graph is always enabled for public repos.

Dependabot alerts
Receive alerts of new vulnerabilities that affect your dependencies.

Dependabot security updates
Easily upgrade to non-vulnerable dependencies.



Bump junit from 4.12 to 4.13.1 in /CryptoOperationsExtractor #1

petrs merged 1 commit into `master` from `dependabot/maven/CryptoOperationsExtractor/junit-junit-4.13.1` on Oct 16

This automated pull request fixes a security vulnerability
Only users with access to Dependabot alerts can see this message. Learn more about Dependabot security updates, opt out, or give us feedback.

Bump junit from 4.12 to 4.13.1 in /CryptoOperationsExtractor

```
Bumps [junit](https://github.com/junit-team/junit4) from 4.12 to 4.13.1.
- [Release notes](https://github.com/junit-team/junit4/releases)
- [Changelog](https://github.com/junit-team/junit4/blob/main/doc/ReleaseNotes4.12.md)
- [Commits](junit-team/junit4@r4.12...r4.13.1)

Signed-off-by: dependabot[bot] <support@github.com>
```

master (#1)

dependabot committed on Oct 13 — commit 33bce080e8b6d3edff492fdedff2eb4bf994eec9

2 CryptoOperationsExtractor/pom.xml

```
@@ -47,7 +47,7 @@
         <dependency>
             <groupId>junit</groupId>
             <artifactId>junit</artifactId>
-            <version>4.12</version>
+            <version>4.13.1</version>
             <scope>test</scope>
         </dependency>
         <dependency>
```

Notes

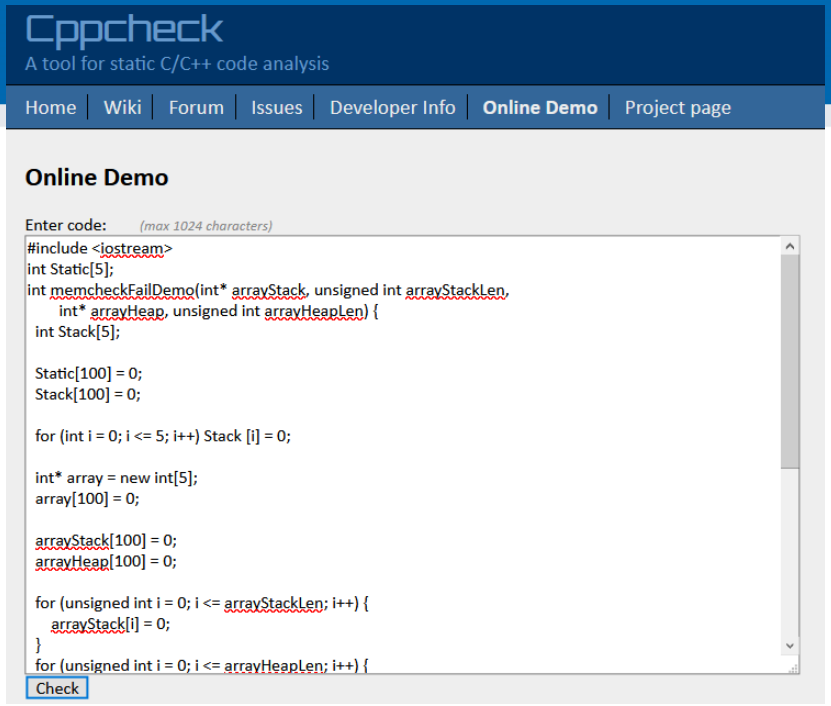
- Dependabot is well established feature of GitHub
- GitHub checks for vulnerabilities in major libraries (dependencies) and notify you if tour repo use it

Run cppcheck locally without Github Actions. Suitable for projects with proprietary code, troubleshooting, execution with non-standard parameters etc.

RUNNING TOOL(S) LOCALLY

Cppcheck for C++ files

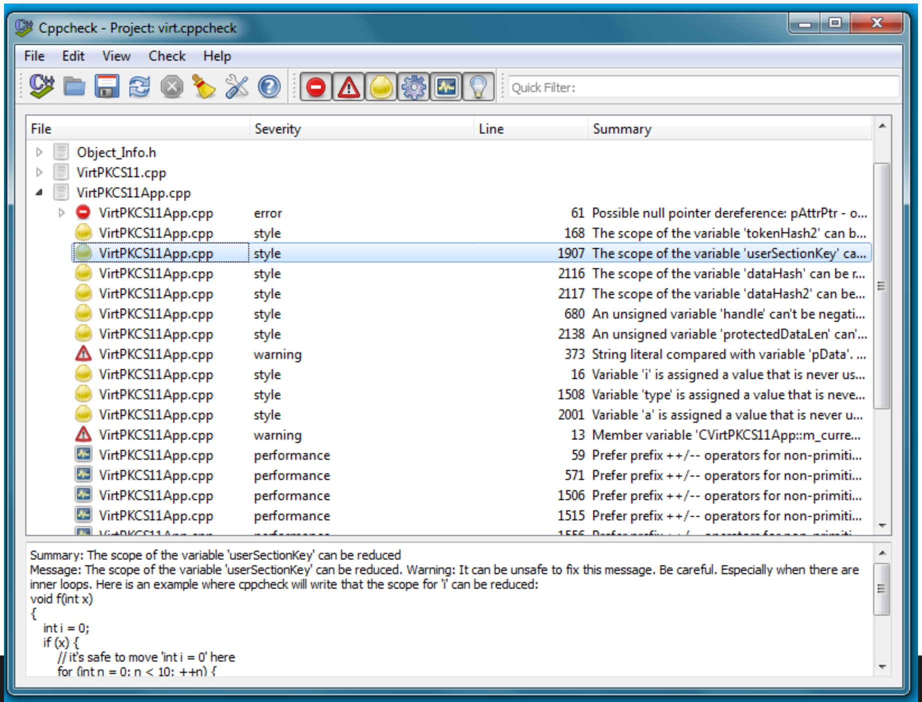
- For small files, you may try cppcheck online
  - https://cppcheck.sourceforge.net/demo/
  - Paste fail.cpp into browser and Check
  - Compare with errors as reported by Codacy
- Run cppcheck from command line
  - Get latest release
    - https://github.com/danmar/cppcheck/releases
  - Run **`cppcheck --enable=all fail.cpp`**
- Run cppcheck via GUI
  - Allows for analysis of folders, sorting by severity…


Cppcheck
A tool for static C/C++ code analysis
Home | Wiki | Forum | Issues | Developer Info | Online Demo | Project page

Online Demo

Enter code: *(max 1024 characters)*

```
#include <iostream>
int Static[5];
int memcheckFailDemo(int* arrayStack, unsigned int arrayStackLen,
    int* arrayHeap, unsigned int arrayHeapLen) {
 int Stack[5];

 Static[100] = 0;
 Stack[100] = 0;

 for (int i = 0; i <= 5; i++) Stack [i] = 0;

 int* array = new int[5];
 array[100] = 0;

 arrayStack[100] = 0;
 arrayHeap[100] = 0;

 for (unsigned int i = 0; i <= arrayStackLen; i++) {
   arrayStack[i] = 0;
 }
 for (unsigned int i = 0; i <= arrayHeapLen; i++) {
```



File	Severity	Line	Summary
Object_Info.h			
VirtPKCS11.cpp			
VirtPKCS11App.cpp			
VirtPKCS11App.cpp	error	61	Possible null pointer dereference: pAttrPtr - o...
VirtPKCS11App.cpp	style	168	The scope of the variable 'tokenHash2' can b...
VirtPKCS11App.cpp	style	1907	The scope of the variable 'userSectionKey' ca...
VirtPKCS11App.cpp	style	2116	The scope of the variable 'dataHash' can be r...
VirtPKCS11App.cpp	style	2117	The scope of the variable 'dataHash2' can be...
VirtPKCS11App.cpp	style	680	An unsigned variable 'handle' can't be negati...
VirtPKCS11App.cpp	style	2138	An unsigned variable 'protectedDataLen' can'...
VirtPKCS11App.cpp	warning	373	String literal compared with variable 'pData'. ...
VirtPKCS11App.cpp	style	16	Variable 'i' is assigned a value that is never us...
VirtPKCS11App.cpp	style	1508	Variable 'type' is assigned a value that is neve...
VirtPKCS11App.cpp	style	2001	Variable 'a' is assigned a value that is never u...
VirtPKCS11App.cpp	warning	13	Member variable 'CVirtPKCS11App::m_curre...
VirtPKCS11App.cpp	performance	59	Prefer prefix ++/-- operators for non-primiti...
VirtPKCS11App.cpp	performance	571	Prefer prefix ++/-- operators for non-primiti...
VirtPKCS11App.cpp	performance	1506	Prefer prefix ++/-- operators for non-primiti...
VirtPKCS11App.cpp	performance	1515	Prefer prefix ++/-- operators for non-primiti...

```
Summary: The scope of the variable 'userSectionKey' can be reduced
Message: The scope of the variable 'userSectionKey' can be reduced. Warning: It can be unsafe to fix this message. Be careful. Especially when there are
inner loops. Here is an example where cppcheck will write that the scope for 'i' can be reduced:
void f(int x)
{
  int i = 0;
  if (x) {
    // it's safe to move 'int i = 0' here
    for (int n = 0; n < 10; ++n) {
```

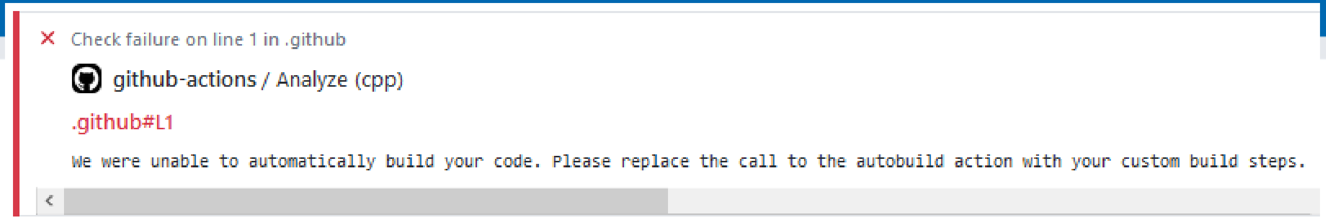
Other tools

- There are many tools for different languages
- Codacy Action is “just” running preconfigured tools
- Try it!

STILL WANT MORE? TRY ON YOUR OLD PB071 HOMEWORK ☺

Some hints on common issues

TROUBLESHOOTING


× Check failure on line 1 in .github

github-actions / Analyze (cpp)

.github#L1

```
We were unable to automatically build your code. Please replace the call to the autobuild action with your custom build steps.
```

Troubleshooting

- Analysis is not finished yet
  - Wait an hour, try to make another bogus commit (update file)
- Start from small working examples, then extend to larger project
  - E.g., simple main.java, only later large java project via ant
- Analyze failed to start for specific language
  - GitHub Actions usually requires code to be compilable
    - Analysis for some languages works on the compiled code/bytecode (e.g., Java)
    - (static analysis runs on unfinished code, but one shall not commit broken code to repo)
  - Github will invoke autobuild feature
    - Tries to build various languages as defined here
      - **https://docs.github.com/en/free-pro-team@latest/github/finding-security-vulnerabilities-and-errors-in-your-code/configuring-the-codeql-workflow-for-compiled-languages**
- Paths case sensitivity
  - Linux is case-sensitive for path names while Windows isn’t
    - /java/ and /Java/ are the same on Windows, but not on Linux
- Clicking on log of ‘Perform Code QL Analysis’ shows nothing
  - Likely GitHub bug, click left on the Analyze (language), then again on ‘Perform Code QL Analysis’
- Makefile requires tabs, not spaces

Some tips

- Setup scanning tools at the beginning of new project
  - And make sure all bugs are always fixed (similar to “compile cleanly” mantra)
- Look at the text logs produced by actions (click on named Action)
  - What tool was executed, what configuration…
- Tools will improve over the time (detected bugs, visualization on GitHub side)

NO HOMEWORK ASSIGNMENT THIS WEEK ☺

CHECK-OUT

Checkout

- Which of the seminar parts you enjoyed most?
- Write three items you liked (ideally, single word)
- Write to sli.do when displayed

slido

Write a topic from the seminar you enjoyed most

ⓘ Start presenting to display the poll results on this slide.

THANK YOU FOR COMING, SEE YOU NEXT WEEK