Testing C

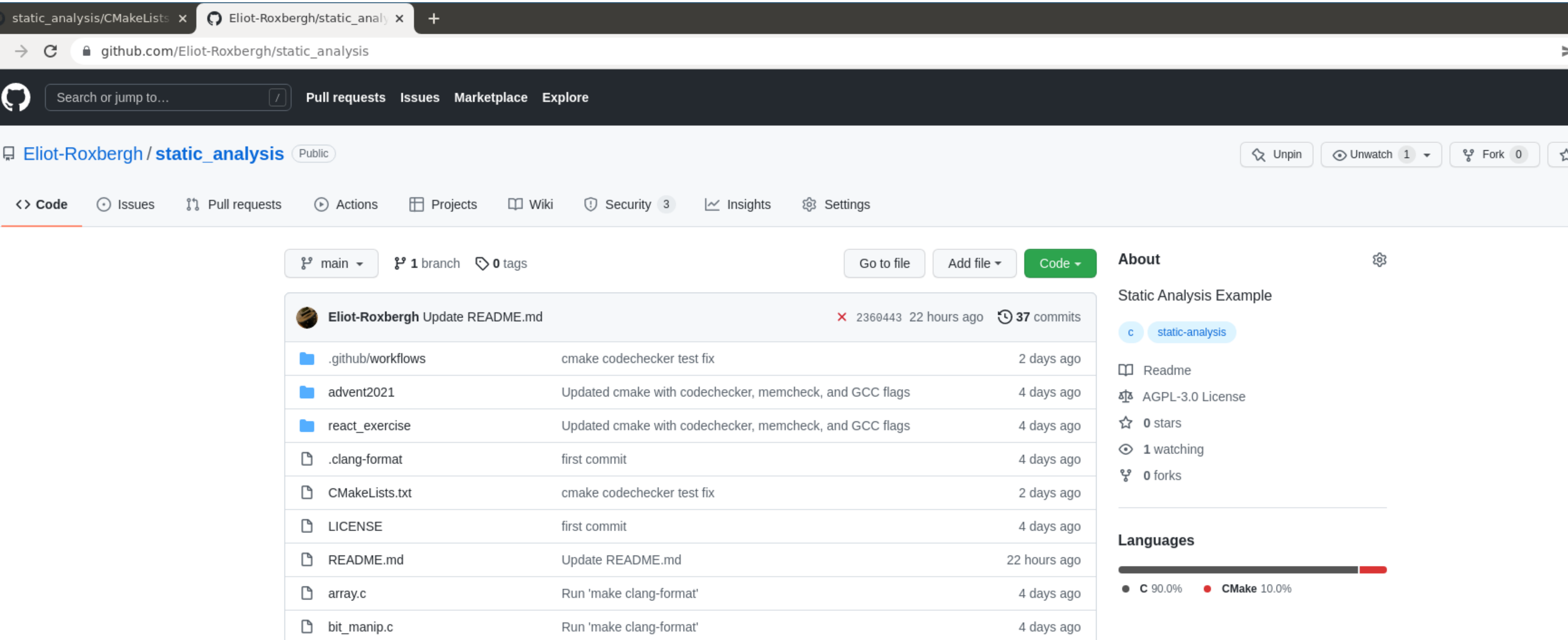
Short on testing C code

Eliot Roxbergh

Testing C

- Compiler / Linker
- Dynamic analysis
- Static analysis
- Unit tests, code coverage, CI/CD ...

The Code



static_analysis/CMakeLists
Eliot-Roxbergh/static_analy
github.com/Eliot-Roxbergh/static_analysis

Eliot-Roxbergh / static_analysis Public

Eliot-Roxbergh Update README.md ✕ 2360443 22 hours ago 37 commits

.github/workflows	cmake codechecker test fix	2 days ago
advent2021	Updated cmake with codechecker, memcheck, and GCC flags	4 days ago
react_exercise	Updated cmake with codechecker, memcheck, and GCC flags	4 days ago
.clang-format	first commit	4 days ago
CMakeLists.txt	cmake codechecker test fix	2 days ago
LICENSE	first commit	4 days ago
README.md	Update README.md	22 hours ago
array.c	Run 'make clang-format'	4 days ago
bit_manip.c	Run 'make clang-format'	4 days ago

About

Static Analysis Example

c static-analysis

Readme
AGPL-3.0 License
0 stars
1 watching
0 forks

Languages

● C 90.0% ● CMake 10.0%

Compiler / Linker

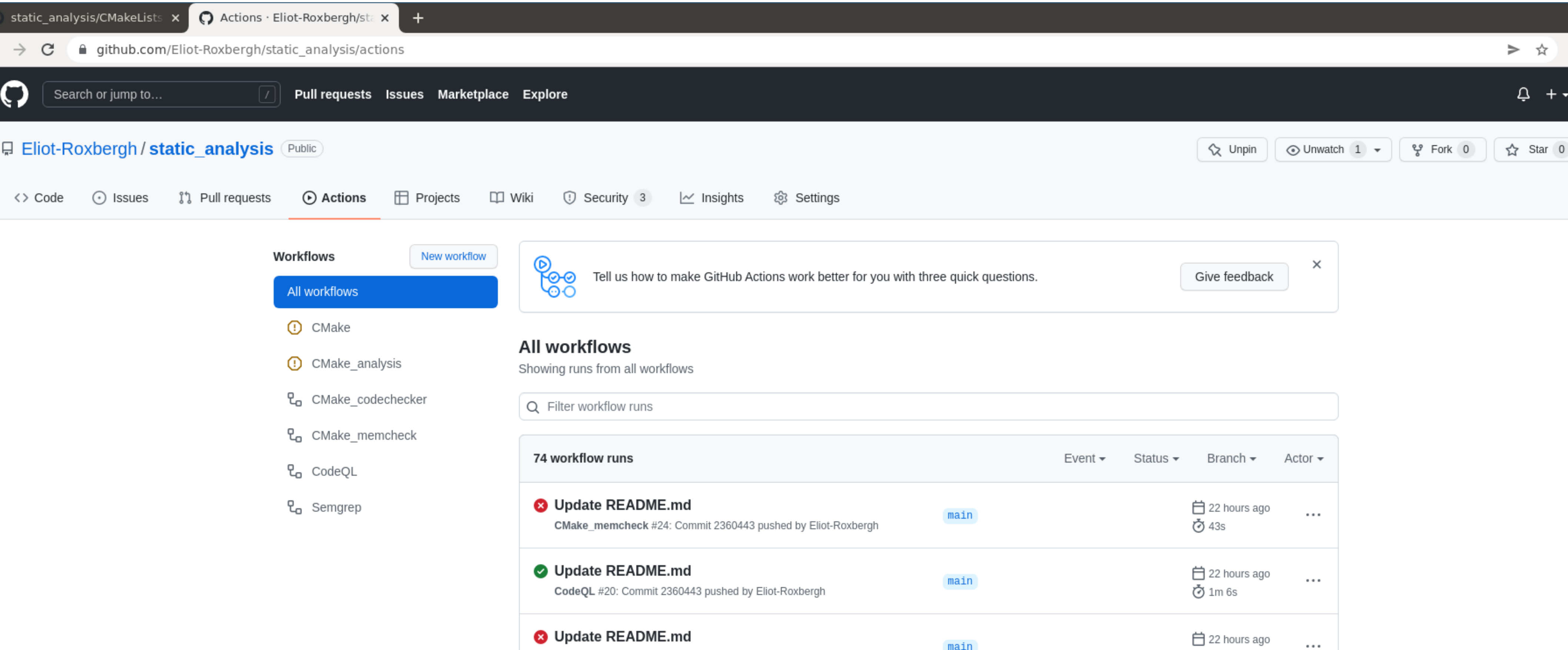
set(C_FLAGS_WARNINGS "**-Wall -Wextra -pedantic -Werror -Wformat=2 -Wconversion**")

set(C_FLAGS_SECURITY "-D_FORTIFY_SOURCE=2 -D_GLIBCXX_ASSERTIONS -fstack-protector-strong -Wl,-z,noexecstack -Wl,-z,now -Wl,-z,relro -Wl,-z,defs")

set(C_FLAGS_SECURITY_EXEC "-fpie -Wl,-pie")

set(C_FLAGS_SECURITY_LIB "-fpic")

CI Pipeline



github.com/Eliot-Roxbergh/static_analysis/actions

Eliot-Roxbergh / static_analysis Public

Workflows

- All workflows
- CMake
- CMake_analysis
- CMake_codechecker
- CMake_memcheck
- CodeQL
- Semgrep

Tell us how to make GitHub Actions work better for you with three quick questions.

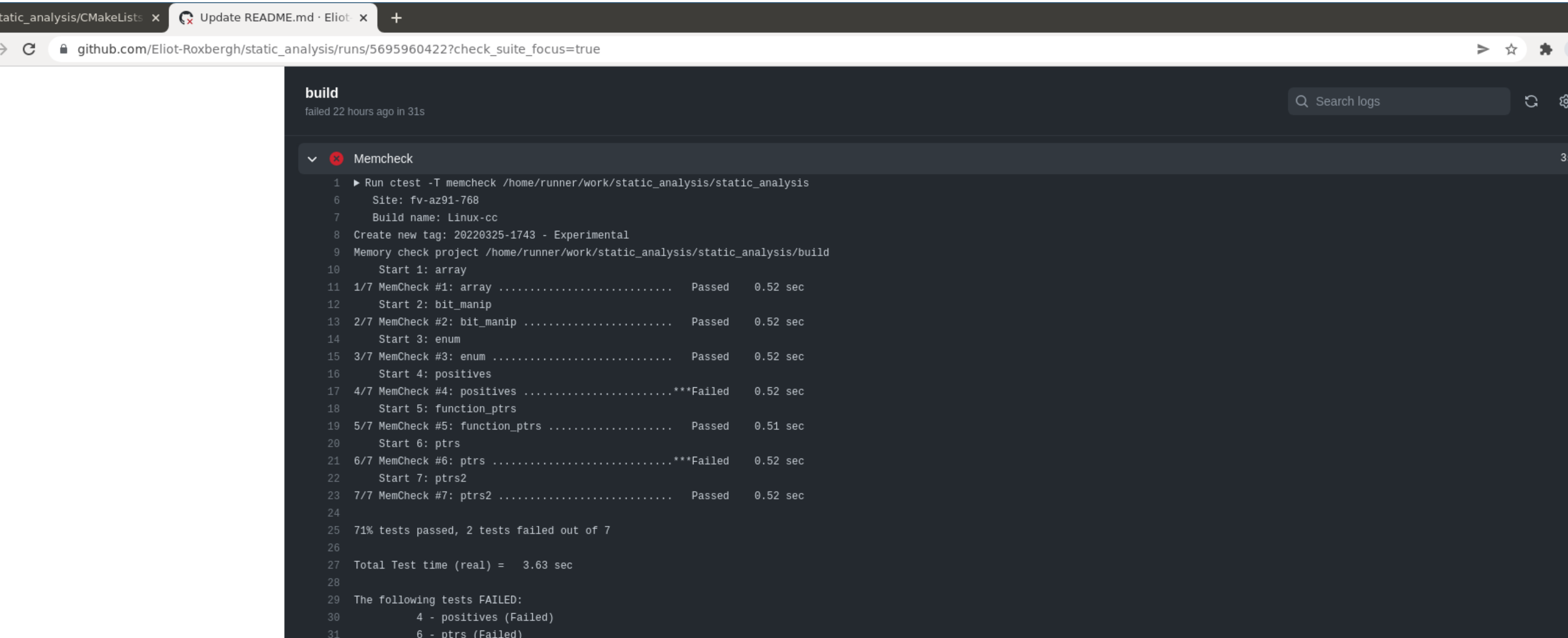
All workflows

Showing runs from all workflows

74 workflow runs

Status	Run	Branch	Time
✗	**Update README.md** **CMake_memcheck** #24: Commit 2360443 pushed by Eliot-Roxbergh	main	22 hours ago, 43s
✓	**Update README.md** **CodeQL** #20: Commit 2360443 pushed by Eliot-Roxbergh	main	22 hours ago, 1m 6s
✗	**Update README.md**	main	22 hours ago

Memcheck (Valgrind)


tatic_analysis/CMakeLists | Update README.md · Eliot

github.com/Eliot-Roxbergh/static_analysis/runs/5695960422?check_suite_focus=true

build
failed 22 hours ago in 31s

Memcheck

```
▶ Run ctest -T memcheck /home/runner/work/static_analysis/static_analysis
  Site: fv-az91-768
  Build name: Linux-cc
Create new tag: 20220325-1743 - Experimental
Memory check project /home/runner/work/static_analysis/static_analysis/build
    Start 1: array
1/7 MemCheck #1: array ............................   Passed    0.52 sec
    Start 2: bit_manip
2/7 MemCheck #2: bit_manip ........................   Passed    0.52 sec
    Start 3: enum
3/7 MemCheck #3: enum .............................   Passed    0.52 sec
    Start 4: positives
4/7 MemCheck #4: positives ........................***Failed    0.52 sec
    Start 5: function_ptrs
5/7 MemCheck #5: function_ptrs ....................   Passed    0.51 sec
    Start 6: ptrs
6/7 MemCheck #6: ptrs .............................***Failed    0.52 sec
    Start 7: ptrs2
7/7 MemCheck #7: ptrs2 ............................   Passed    0.52 sec

71% tests passed, 2 tests failed out of 7

Total Test time (real) =   3.63 sec

The following tests FAILED:
          4 - positives (Failed)
          6 - ptrs (Failed)
```

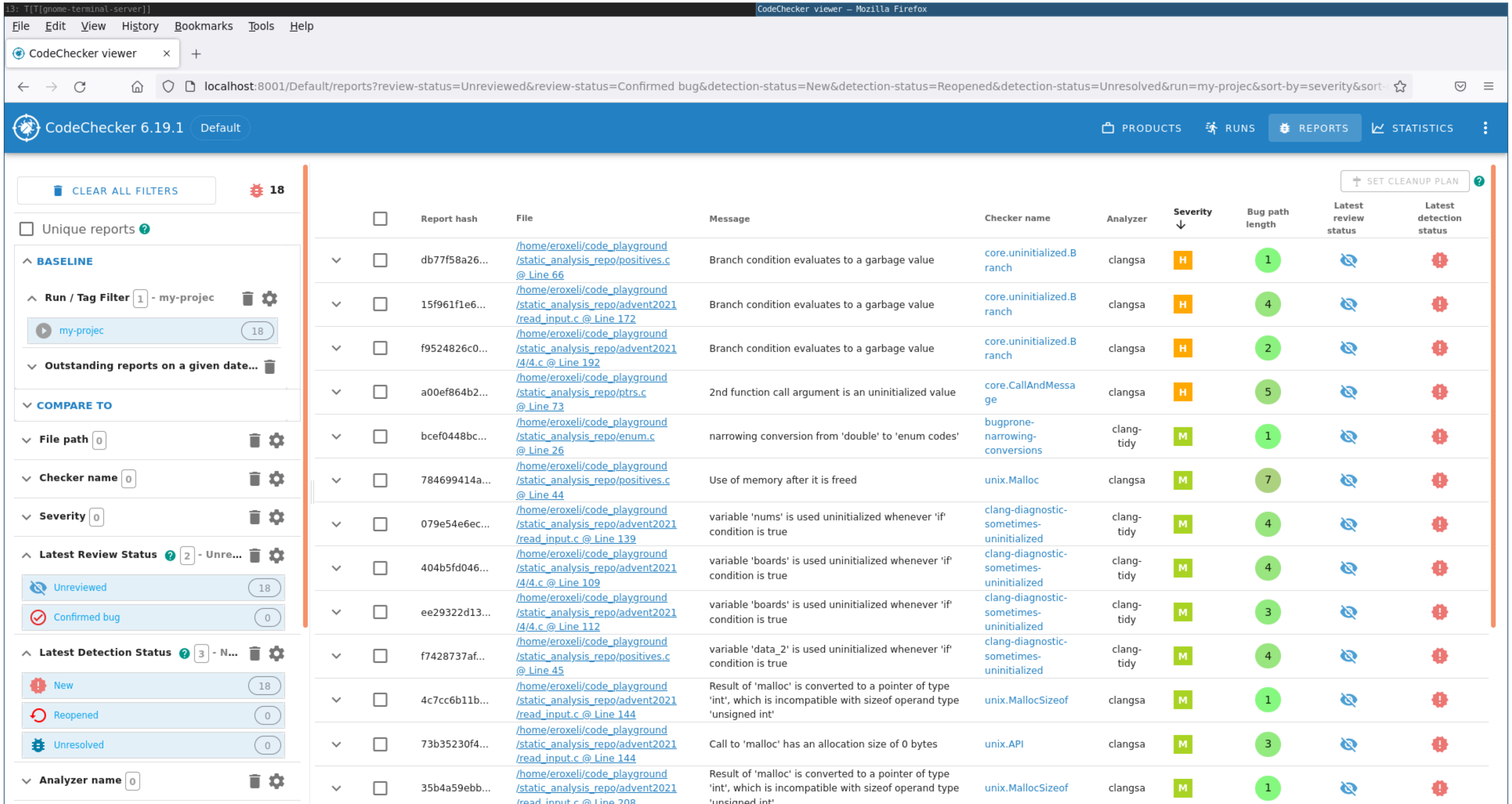
Static analysis

- Clang-tidy (/w CodeChecker front-end)
- Semgrep
- CodeQL

Static analysis

- **Clang-tidy** (/w CodeChecker front-end)
  - 12 TPs, 8 FPs, +2 duplicates
- **Semgrep**
  - 2 TPs, 11 FPs, +3 duplicates
- **CodeQL**
  - 2 TPs, 0 FPs, +1 duplicates
- **In total**
  19 FPs
  16 Tps
  +6 duplicate TPs (very little overlap!)

Clang-tidy


i3: T[T[gnome-terminal-server]]
CodeChecker viewer — Mozilla Firefox
File Edit View History Bookmarks Tools Help
CodeChecker viewer
CodeChecker 6.19.1 Default
PRODUCTS RUNS REPORTS STATISTICS
CLEAR ALL FILTERS
SET CLEANUP PLAN

localhost:8001/Default/reports?review-status=Unreviewed&review-status=Confirmed bug&detection-status=New&detection-status=Reopened&detection-status=Unresolved&run=my-projec&sort-by=severity&sort-

18

Unique reports

BASELINE

Run / Tag Filter 1 - my-projec

my-projec 18

Outstanding reports on a given date...

COMPARE TO

File path 0

Checker name 0

Severity 0

Latest Review Status 2 - Unre...

Unreviewed 18

Confirmed bug 0

Latest Detection Status 3 - N...

New 18

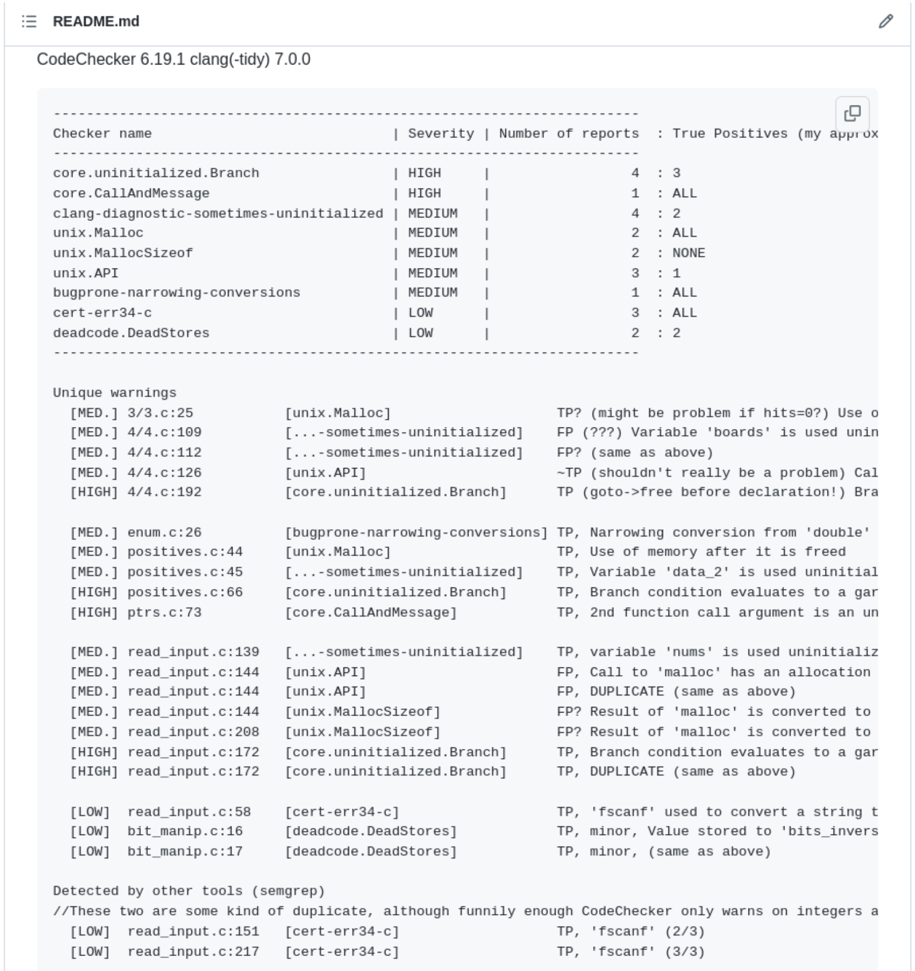
Reopened 0

Unresolved 0

Analyzer name 0

Report hash	File	Message	Checker name	Analyzer	Severity	Bug path length	Latest review status	Latest detection status
db77f58a26...	/home/eroxeli/code_playground/static_analysis_repo/positives.c @ Line 66	Branch condition evaluates to a garbage value	core.uninitialized.Branch	clangsa	H	1	⊘	❗
15f961f1e6...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/read_input.c @ Line 172	Branch condition evaluates to a garbage value	core.uninitialized.Branch	clangsa	H	4	⊘	❗
f9524826c0...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/4/4.c @ Line 192	Branch condition evaluates to a garbage value	core.uninitialized.Branch	clangsa	H	2	⊘	❗
a00ef864b2...	/home/eroxeli/code_playground/static_analysis_repo/ptrs.c @ Line 73	2nd function call argument is an uninitialized value	core.CallAndMessage	clangsa	H	5	⊘	❗
bcef0448bc...	/home/eroxeli/code_playground/static_analysis_repo/enum.c @ Line 26	narrowing conversion from 'double' to 'enum codes'	bugprone-narrowing-conversions	clang-tidy	M	1	⊘	❗
784699414a...	/home/eroxeli/code_playground/static_analysis_repo/positives.c @ Line 44	Use of memory after it is freed	unix.Malloc	clangsa	M	7	⊘	❗
079e54e6ec...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/read_input.c @ Line 139	variable 'nums' is used uninitialized whenever 'if' condition is true	clang-diagnostic-sometimes-uninitialized	clang-tidy	M	4	⊘	❗
404b5fd046...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/4/4.c @ Line 109	variable 'boards' is used uninitialized whenever 'if' condition is true	clang-diagnostic-sometimes-uninitialized	clang-tidy	M	4	⊘	❗
ee29322d13...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/4/4.c @ Line 112	variable 'boards' is used uninitialized whenever 'if' condition is true	clang-diagnostic-sometimes-uninitialized	clang-tidy	M	3	⊘	❗
f7428737af...	/home/eroxeli/code_playground/static_analysis_repo/positives.c @ Line 45	variable 'data_2' is used uninitialized whenever 'if' condition is true	clang-diagnostic-sometimes-uninitialized	clang-tidy	M	4	⊘	❗
4c7cc6b11b...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/read_input.c @ Line 144	Result of 'malloc' is converted to a pointer of type 'int', which is incompatible with sizeof operand type 'unsigned int'	unix.MallocSizeof	clangsa	M	1	⊘	❗
73b35230f4...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/read_input.c @ Line 144	Call to 'malloc' has an allocation size of 0 bytes	unix.API	clangsa	M	3	⊘	❗
35b4a59ebb...	/home/eroxeli/code_playground/static_analysis_repo/advent2021/read_input.c @ Line 208	Result of 'malloc' is converted to a pointer of type 'int', which is incompatible with sizeof operand type 'unsigned int'	unix.MallocSizeof	clangsa	M	1	⊘	❗

Clang-tidy



README.md

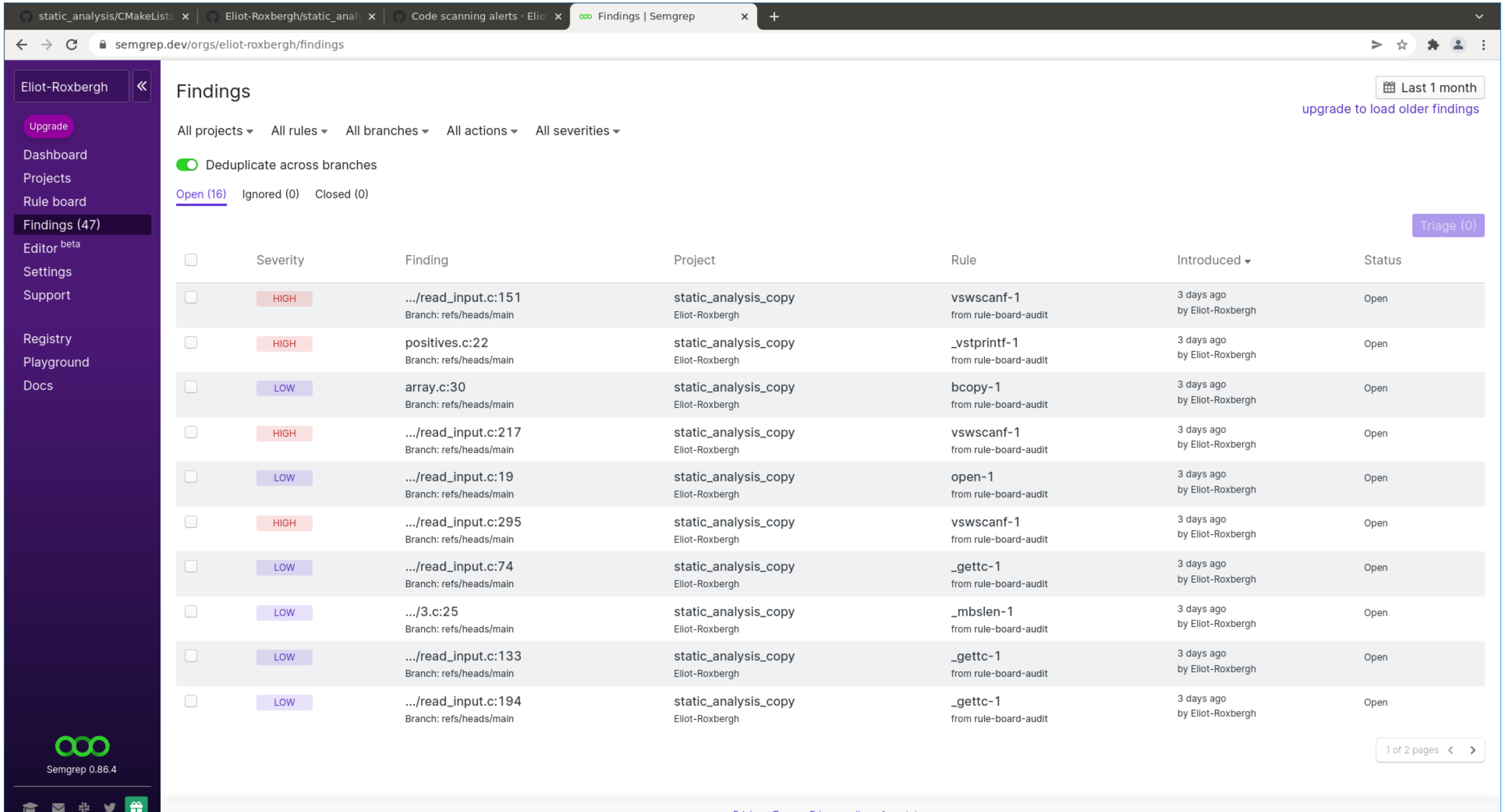
CodeChecker 6.19.1 clang(-tidy) 7.0.0

```
-----------------------------------------------------------------------
Checker name                             | Severity | Number of reports  : True Positives (my approx
-----------------------------------------------------------------------
core.uninitialized.Branch                | HIGH     |                 4  : 3
core.CallAndMessage                      | HIGH     |                 1  : ALL
clang-diagnostic-sometimes-uninitialized | MEDIUM   |                 4  : 2
unix.Malloc                              | MEDIUM   |                 2  : ALL
unix.MallocSizeof                        | MEDIUM   |                 2  : NONE
unix.API                                 | MEDIUM   |                 3  : 1
bugprone-narrowing-conversions           | MEDIUM   |                 1  : ALL
cert-err34-c                             | LOW      |                 3  : ALL
deadcode.DeadStores                      | LOW      |                 2  : 2
-----------------------------------------------------------------------


Unique warnings
  [MED.] 3/3.c:25          [unix.Malloc]                   TP? (might be problem if hits=0?) Use o
  [MED.] 4/4.c:109         [...-sometimes-uninitialized]   FP (???) Variable 'boards' is used unin
  [MED.] 4/4.c:112         [...-sometimes-uninitialized]   FP? (same as above)
  [MED.] 4/4.c:126         [unix.API]                      ~TP (shouldn't really be a problem) Cal
  [HIGH] 4/4.c:192         [core.uninitialized.Branch]     TP (goto->free before declaration!) Bra

  [MED.] enum.c:26         [bugprone-narrowing-conversions] TP, Narrowing conversion from 'double'
  [MED.] positives.c:44    [unix.Malloc]                   TP, Use of memory after it is freed
  [MED.] positives.c:45    [...-sometimes-uninitialized]   TP, Variable 'data_2' is used uninitial
  [HIGH] positives.c:66    [core.uninitialized.Branch]     TP, Branch condition evaluates to a gar
  [HIGH] ptrs.c:73         [core.CallAndMessage]           TP, 2nd function call argument is an un

  [MED.] read_input.c:139  [...-sometimes-uninitialized]   TP, variable 'nums' is used uninitializ
  [MED.] read_input.c:144  [unix.API]                      FP, Call to 'malloc' has an allocation
  [MED.] read_input.c:144  [unix.API]                      FP, DUPLICATE (same as above)
  [MED.] read_input.c:144  [unix.MallocSizeof]             FP? Result of 'malloc' is converted to
  [MED.] read_input.c:208  [unix.MallocSizeof]             FP? Result of 'malloc' is converted to
  [HIGH] read_input.c:172  [core.uninitialized.Branch]     TP, Branch condition evaluates to a gar
  [HIGH] read_input.c:172  [core.uninitialized.Branch]     TP, DUPLICATE (same as above)

  [LOW]  read_input.c:58   [cert-err34-c]                  TP, 'fscanf' used to convert a string t
  [LOW]  bit_manip.c:16    [deadcode.DeadStores]           TP, minor, Value stored to 'bits_invers
  [LOW]  bit_manip.c:17    [deadcode.DeadStores]           TP, minor, (same as above)

Detected by other tools (semgrep)
//These two are some kind of duplicate, although funnily enough CodeChecker only warns on integers a
  [LOW]  read_input.c:151  [cert-err34-c]                  TP, 'fscanf' (2/3)
  [LOW]  read_input.c:217  [cert-err34-c]                  TP, 'fscanf' (3/3)
```

3 free without malloc,
due to [uninitialized.Branch]
1 use after free [unix.malloc]
1 conversion double → enum

Several good suggestions / minor potential bugs

Semgrep



semgrep.dev/orgs/eliot-roxbergh/findings

Eliot-Roxbergh

Upgrade

Dashboard
Projects
Rule board
Findings (47)
Editor beta
Settings
Support

Registry
Playground
Docs

Semgrep 0.86.4

Findings

Last 1 month
upgrade to load older findings

All projects ▾ All rules ▾ All branches ▾ All actions ▾ All severities ▾

Deduplicate across branches

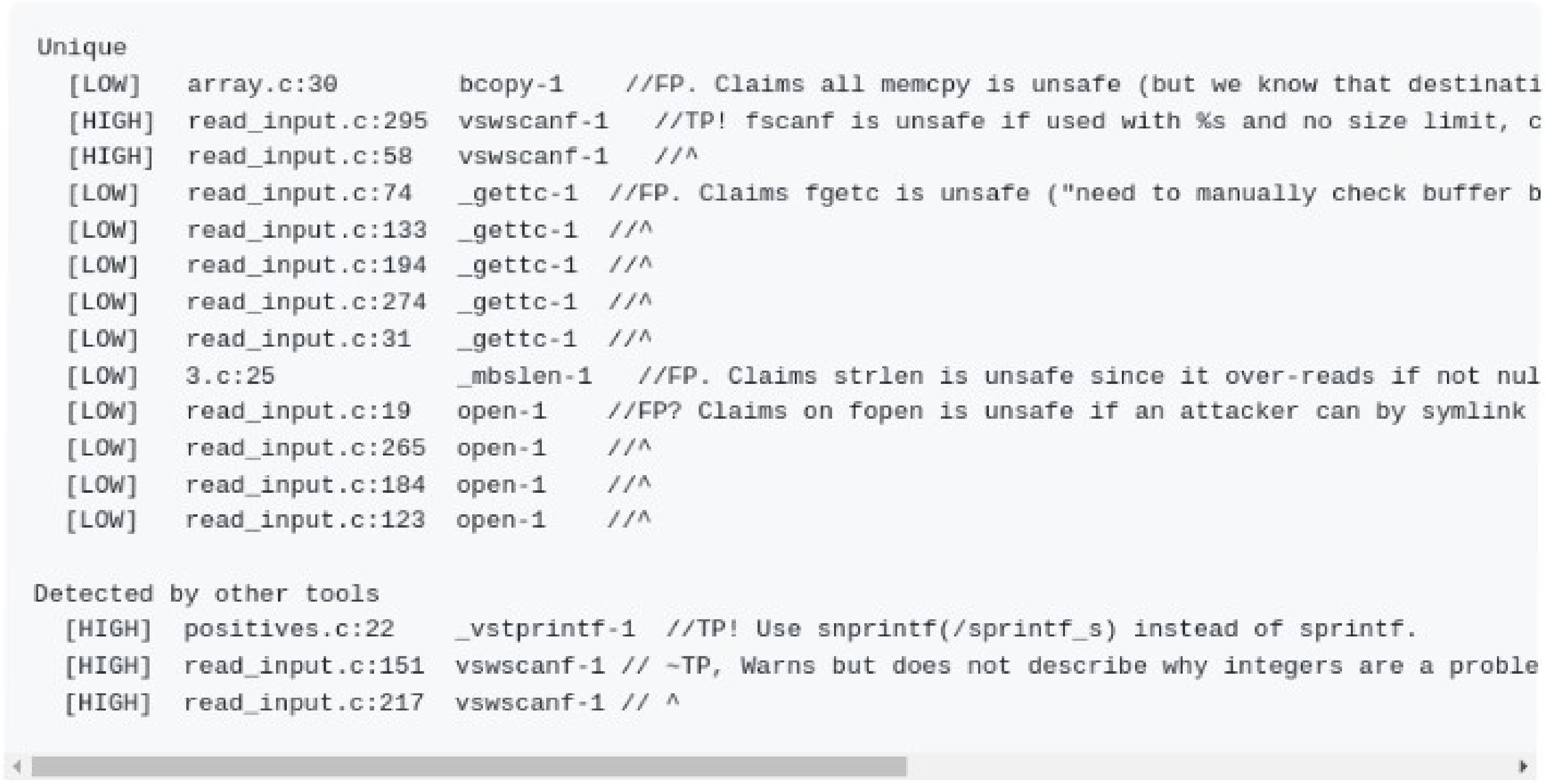
Open (16) Ignored (0) Closed (0)

Triage (0)

Severity	Finding	Project	Rule	Introduced ▾	Status
HIGH	.../read_input.c:151 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	vswscanf-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
HIGH	positives.c:22 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	_vstprintf-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	array.c:30 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	bcopy-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
HIGH	.../read_input.c:217 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	vswscanf-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	.../read_input.c:19 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	open-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
HIGH	.../read_input.c:295 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	vswscanf-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	.../read_input.c:74 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	_gettc-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	.../3.c:25 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	_mbslen-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	.../read_input.c:133 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	_gettc-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open
LOW	.../read_input.c:194 Branch: refs/heads/main	static_analysis_copy Eliot-Roxbergh	_gettc-1 from rule-board-audit	3 days ago by Eliot-Roxbergh	Open

1 of 2 pages

Semgrep

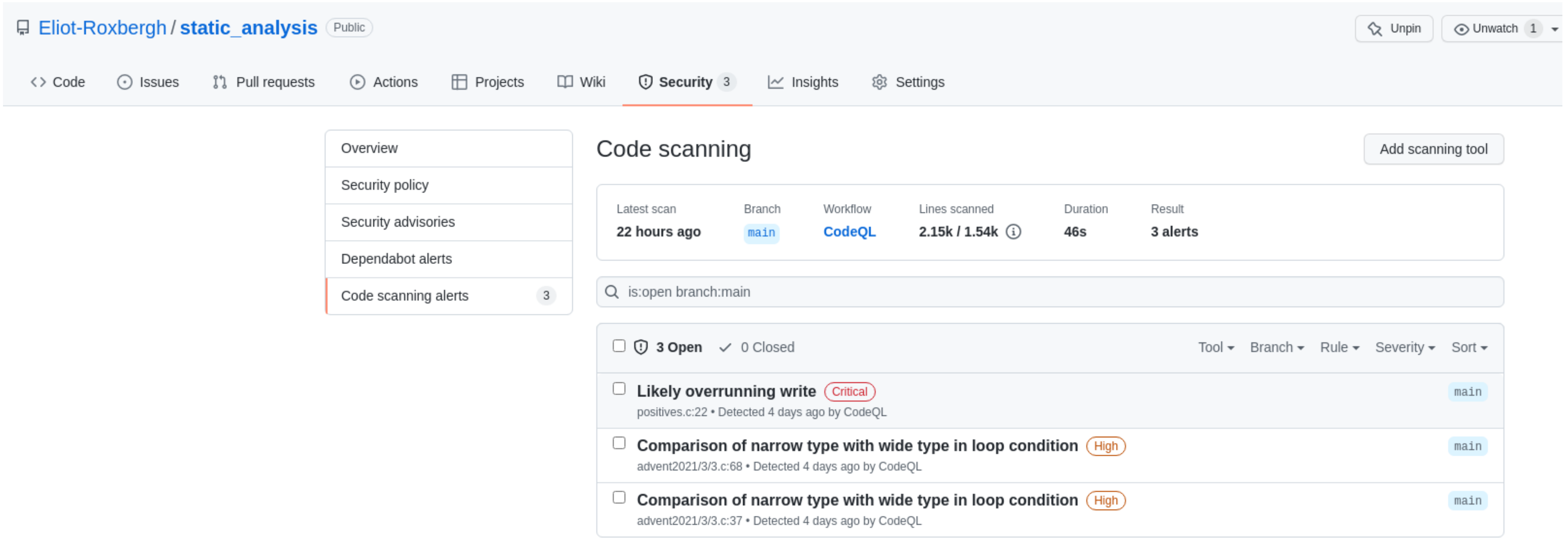


```
Unique
  [LOW]   array.c:30        bcopy-1    //FP. Claims all memcpy is unsafe (but we know that destinati
  [HIGH]  read_input.c:295  vswscanf-1   //TP! fscanf is unsafe if used with %s and no size limit, c
  [HIGH]  read_input.c:58   vswscanf-1   //^
  [LOW]   read_input.c:74   _gettc-1  //FP. Claims fgetc is unsafe ("need to manually check buffer b
  [LOW]   read_input.c:133  _gettc-1  //^
  [LOW]   read_input.c:194  _gettc-1  //^
  [LOW]   read_input.c:274  _gettc-1  //^
  [LOW]   read_input.c:31   _gettc-1  //^
  [LOW]   3.c:25            _mbslen-1   //FP. Claims strlen is unsafe since it over-reads if not nul
  [LOW]   read_input.c:19   open-1    //FP? Claims on fopen is unsafe if an attacker can by symlink
  [LOW]   read_input.c:265  open-1    //^
  [LOW]   read_input.c:184  open-1    //^
  [LOW]   read_input.c:123  open-1    //^

Detected by other tools
  [HIGH]  positives.c:22    _vstprintf-1  //TP! Use snprintf(/sprintf_s) instead of sprintf.
  [HIGH]  read_input.c:151  vswscanf-1 // ~TP, Warns but does not describe why integers are a proble
  [HIGH]  read_input.c:217  vswscanf-1 // ^
```

2 buffer overflow, due to unsafe use of fscanf from user input

Suggests to use safer functions...

CodeQL


Eliot-Roxbergh / static_analysis Public

Unpin | Unwatch 1

Code | Issues | Pull requests | Actions | Projects | Wiki | Security 3 | Insights | Settings

Overview
Security policy
Security advisories
Dependabot alerts
Code scanning alerts 3

Code scanning

Add scanning tool

Latest scan	Branch	Workflow	Lines scanned	Duration	Result
22 hours ago	main	CodeQL	2.15k / 1.54k	46s	3 alerts

is:open branch:main

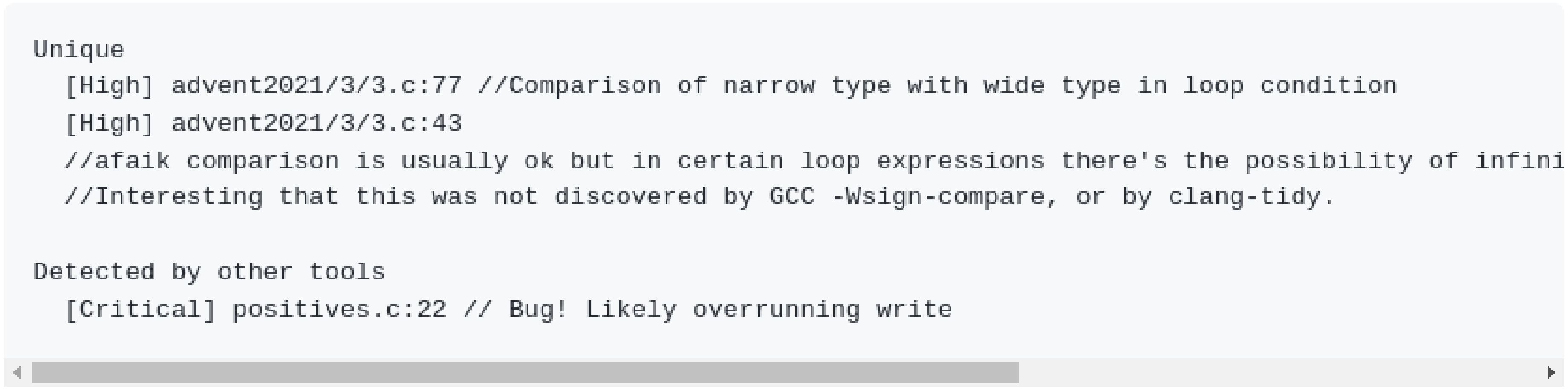
3 Open ✓ 0 Closed — Tool Branch Rule Severity Sort

Likely overrunning write Critical — main
positives.c:22 • Detected 4 days ago by CodeQL

Comparison of narrow type with wide type in loop condition High — main
advent2021/3/3.c:68 • Detected 4 days ago by CodeQL

Comparison of narrow type with wide type in loop condition High — main
advent2021/3/3.c:37 • Detected 4 days ago by CodeQL

CodeQL



```
Unique
  [High] advent2021/3/3.c:77 //Comparison of narrow type with wide type in loop condition
  [High] advent2021/3/3.c:43
  //afaik comparison is usually ok but in certain loop expressions there's the possibility of infini
  //Interesting that this was not discovered by GCC -Wsign-compare, or by clang-tidy.

Detected by other tools
  [Critical] positives.c:22 // Bug! Likely overrunning write
```

2 infinite loop due to comparison

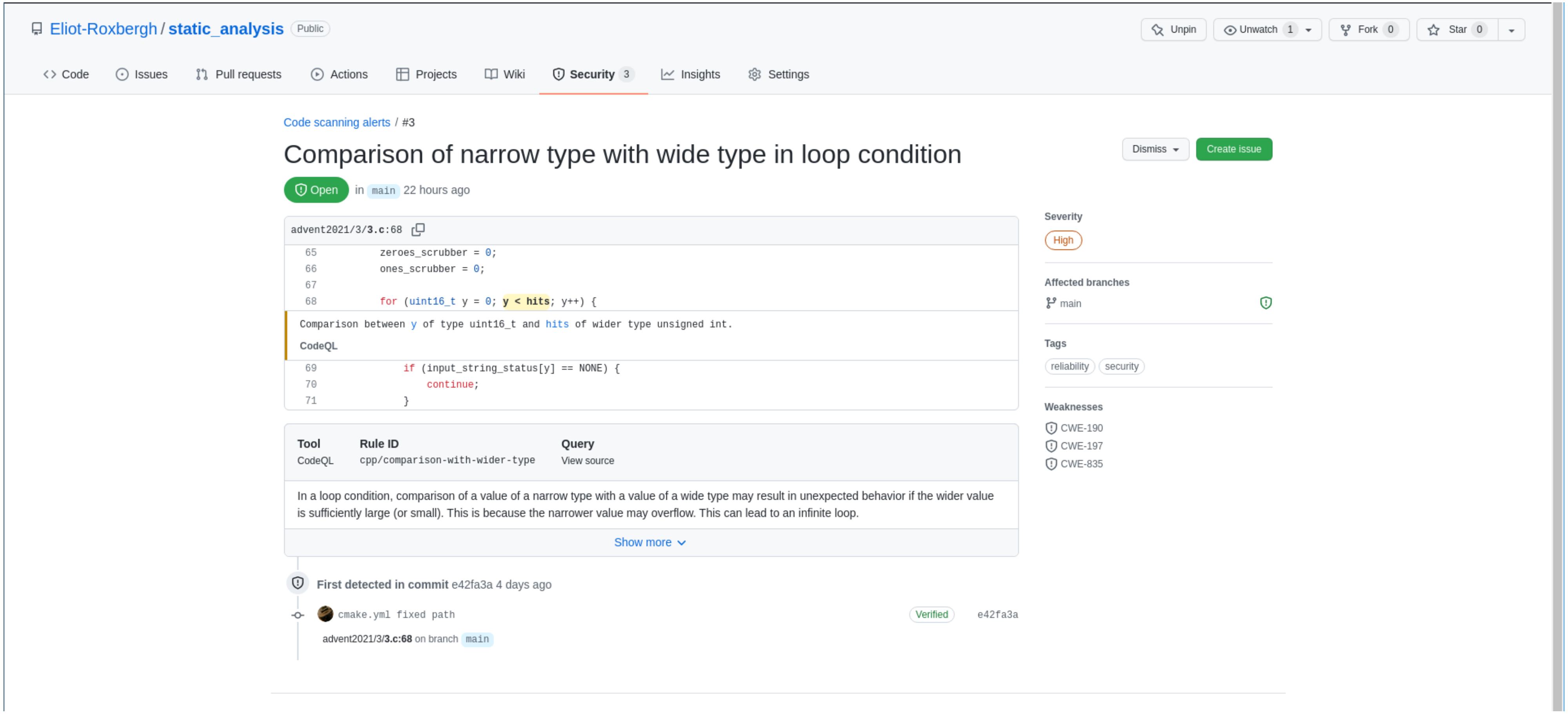
(1 overflow, detected also by semgrep)

Example of Bugs

- DoS
- Double free /
  Free without malloc
- Use after free
- Malloc without free

- Exploitable?

Infinite Loop



Eliot-Roxbergh / static_analysis Public

Code scanning alerts / #3

Comparison of narrow type with wide type in loop condition

Open in `main` 22 hours ago

advent2021/3/**3.c**:68

```
65          zeroes_scrubber = 0;
66          ones_scrubber = 0;
67
68          for (uint16_t y = 0; y < hits; y++) {
```

```
Comparison between y of type uint16_t and hits of wider type unsigned int.
```

CodeQL

```
69              if (input_string_status[y] == NONE) {
70                  continue;
71              }
```

Tool	Rule ID	Query
CodeQL	`cpp/comparison-with-wider-type`	View source

In a loop condition, comparison of a value of a narrow type with a value of a wide type may result in unexpected behavior if the wider value is sufficiently large (or small). This is because the narrower value may overflow. This can lead to an infinite loop.

Show more

First detected in commit e42fa3a 4 days ago

`cmake.yml fixed path` Verified `e42fa3a`

advent2021/3/**3.c:68** on branch `main`

Severity

High

Affected branches

main

Tags

reliability security

Weaknesses

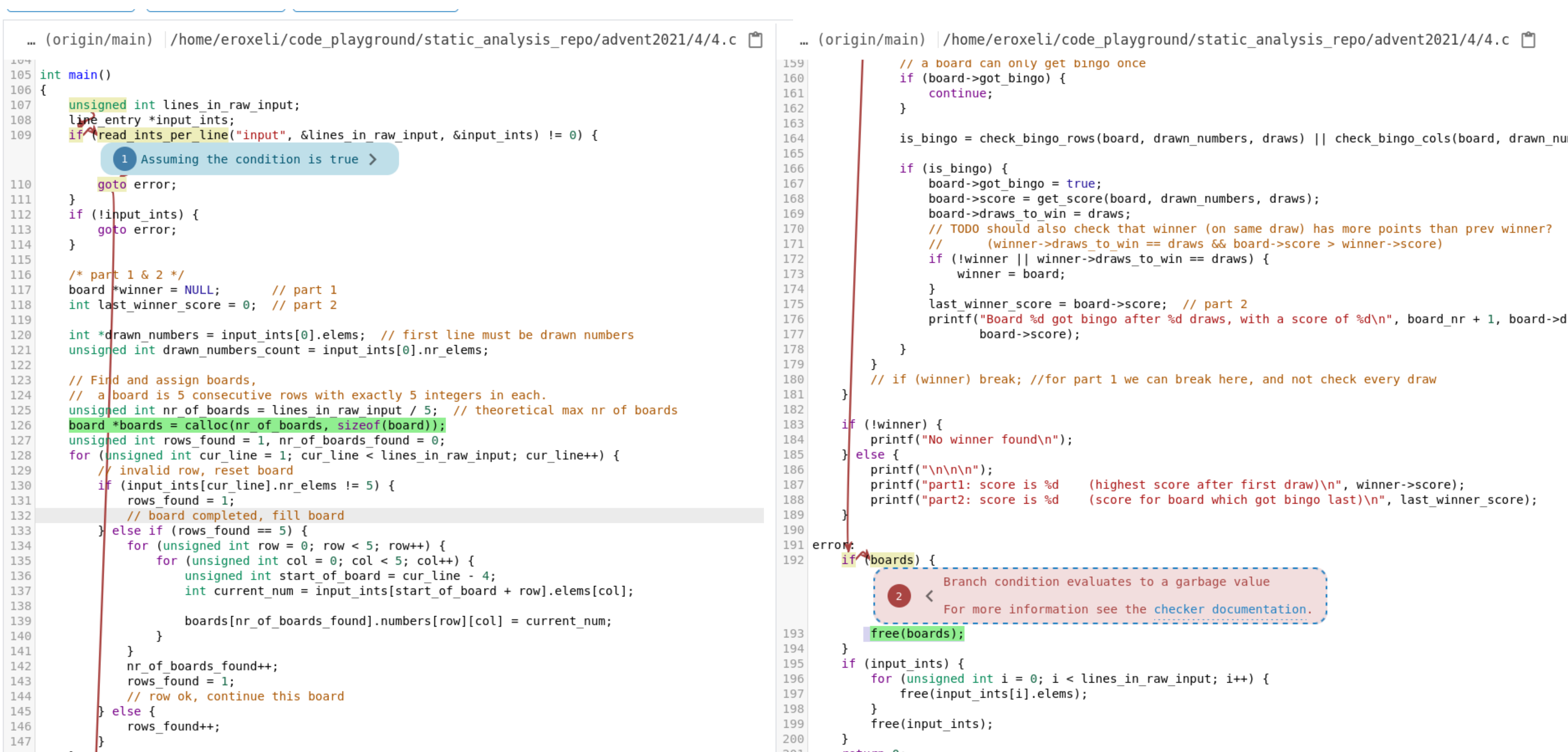
CWE-190
CWE-197
CWE-835

Infinite Loop

- Denial-of-service
- (In this case if the file has more than 2^16 lines)

- Peculiar, not detected by other tools, or gcc -Wconversion, maybe because this is well-defined behavior?

Free without malloc



… (origin/main) /home/eroxeli/code_playground/static_analysis_repo/advent2021/4/4.c

```
int main()
{
    unsigned int lines_in_raw_input;
    line_entry *input_ints;
    if (read_ints_per_line("input", &lines_in_raw_input, &input_ints) != 0) {
        1 Assuming the condition is true >
        goto error;
    }
    if (!input_ints) {
        goto error;
    }

    /* part 1 & 2 */
    board *winner = NULL;       // part 1
    int last_winner_score = 0;  // part 2

    int *drawn_numbers = input_ints[0].elems;  // first line must be drawn numbers
    unsigned int drawn_numbers_count = input_ints[0].nr_elems;

    // Find and assign boards,
    //  a board is 5 consecutive rows with exactly 5 integers in each.
    unsigned int nr_of_boards = lines_in_raw_input / 5;  // theoretical max nr of boards
    board *boards = calloc(nr_of_boards, sizeof(board));
    unsigned int rows_found = 1, nr_of_boards_found = 0;
    for (unsigned int cur_line = 1; cur_line < lines_in_raw_input; cur_line++) {
        // invalid row, reset board
        if (input_ints[cur_line].nr_elems != 5) {
            rows_found = 1;
            // board completed, fill board
        } else if (rows_found == 5) {
            for (unsigned int row = 0; row < 5; row++) {
                for (unsigned int col = 0; col < 5; col++) {
                    unsigned int start_of_board = cur_line - 4;
                    int current_num = input_ints[start_of_board + row].elems[col];

                    boards[nr_of_boards_found].numbers[row][col] = current_num;
                }
            }
            nr_of_boards_found++;
            rows_found = 1;
            // row ok, continue this board
        } else {
            rows_found++;
        }
```

… (origin/main) /home/eroxeli/code_playground/static_analysis_repo/advent2021/4/4.c

```
            // a board can only get bingo once
            if (board->got_bingo) {
                continue;
            }

            is_bingo = check_bingo_rows(board, drawn_numbers, draws) || check_bingo_cols(board, drawn_nu

            if (is_bingo) {
                board->got_bingo = true;
                board->score = get_score(board, drawn_numbers, draws);
                board->draws_to_win = draws;
                // TODO should also check that winner (on same draw) has more points than prev winner?
                //      (winner->draws_to_win == draws && board->score > winner->score)
                if (!winner || winner->draws_to_win == draws) {
                    winner = board;
                }
                last_winner_score = board->score;  // part 2
                printf("Board %d got bingo after %d draws, with a score of %d\n", board_nr + 1, board->d
                       board->score);
            }
        }
        // if (winner) break; //for part 1 we can break here, and not check every draw
    }

    if (!winner) {
        printf("No winner found\n");
    } else {
        printf("\n\n\n");
        printf("part1: score is %d    (highest score after first draw)\n", winner->score);
        printf("part2: score is %d    (score for board which got bingo last)\n", last_winner_score);
    }

error:
    if (boards) {
        2 <  Branch condition evaluates to a garbage value
             For more information see the checker documentation.
        free(boards);
    }
    if (input_ints) {
        for (unsigned int i = 0; i < lines_in_raw_input; i++) {
            free(input_ints[i].elems);
        }
        free(input_ints);
    }
```

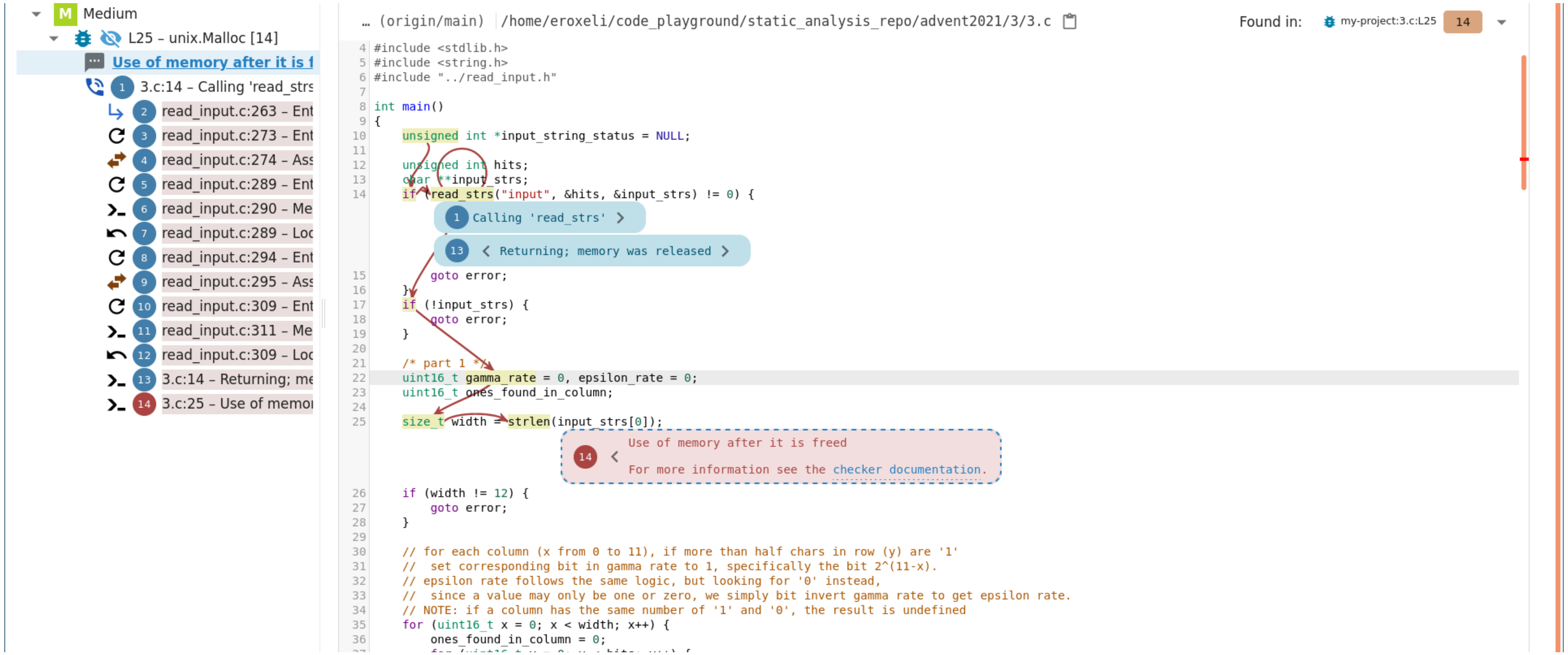
Free without malloc

- Possibly confuse memory manager

  Subsequent malloc return same address

  => In general..
    potentially unauthorized
    read/writes possible

Use after free



Medium
L25 – unix.Malloc [14]
Use of memory after it is f
1 3.c:14 – Calling 'read_strs
2 read_input.c:263 – Ent
3 read_input.c:273 – Ent
4 read_input.c:274 – Ass
5 read_input.c:289 – Ent
6 read_input.c:290 – Me
7 read_input.c:289 – Loo
8 read_input.c:294 – Ent
9 read_input.c:295 – Ass
10 read_input.c:309 – Ent
11 read_input.c:311 – Me
12 read_input.c:309 – Loo
13 3.c:14 – Returning; me
14 3.c:25 – Use of memor

… (origin/main) /home/eroxeli/code_playground/static_analysis_repo/advent2021/3/3.c

Found in: my-project:3.c:L25 14

```
 4 #include <stdlib.h>
 5 #include <string.h>
 6 #include "../read_input.h"
 7
 8 int main()
 9 {
10     unsigned int *input_string_status = NULL;
11
12     unsigned int hits;
13     char **input_strs;
14     if (read_strs("input", &hits, &input_strs) != 0) {
           1 Calling 'read_strs' >
           13 < Returning; memory was released >
15         goto error;
16     }
17     if (!input_strs) {
18         goto error;
19     }
20
21     /* part 1 */
22     uint16_t gamma_rate = 0, epsilon_rate = 0;
23     uint16_t ones_found_in_column;
24
25     size_t width = strlen(input_strs[0]);
           14 < Use of memory after it is freed
                For more information see the checker documentation.
26     if (width != 12) {
27         goto error;
28     }
29
30     // for each column (x from 0 to 11), if more than half chars in row (y) are '1'
31     //  set corresponding bit in gamma rate to 1, specifically the bit 2^(11-x).
32     // epsilon rate follows the same logic, but looking for '0' instead,
33     //  since a value may only be one or zero, we simply bit invert gamma rate to get epsilon rate.
34     // NOTE: if a column has the same number of '1' and '0', the result is undefined
35     for (uint16_t x = 0; x < width; x++) {
36         ones_found_in_column = 0;
```

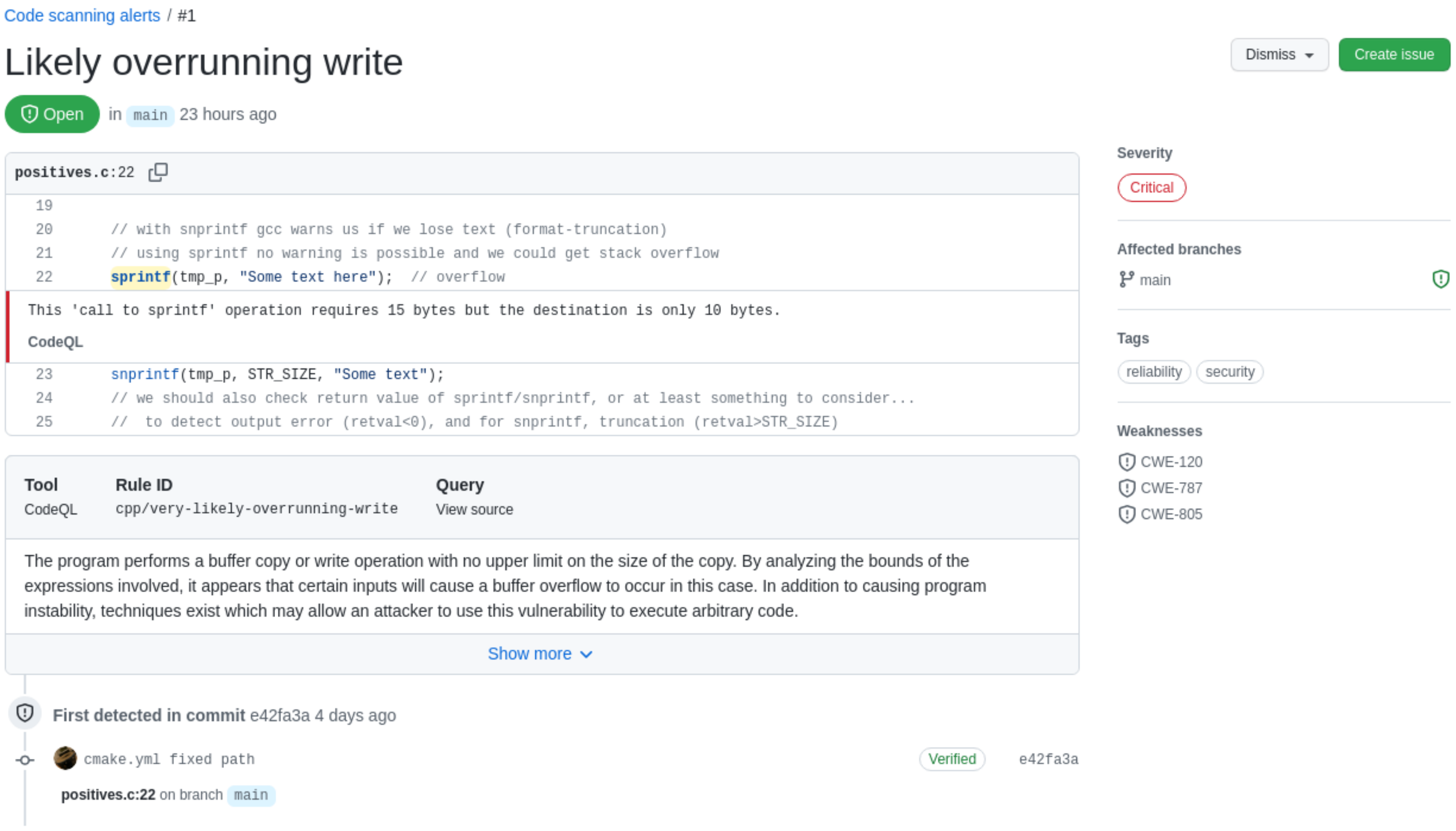
Empty file, doesn't check that: *hits != 0*
=> input_str[0] = /* freed memory */;

Use after free

- This block is free to be reused by another malloc call
  => in general..

  potential unauthorized read/writes possible via the old reference
- Here, it’s only read after free by strlen

Buffer overflow



Code scanning alerts / #1

Likely overrunning write

in main 23 hours ago

positives.c:22

```
19
20      // with snprintf gcc warns us if we lose text (format-truncation)
21      // using sprintf no warning is possible and we could get stack overflow
22      sprintf(tmp_p, "Some text here");  // overflow
```

This 'call to sprintf' operation requires 15 bytes but the destination is only 10 bytes.

CodeQL

```
23      snprintf(tmp_p, STR_SIZE, "Some text");
24      // we should also check return value of sprintf/snprintf, or at least something to consider...
25      //  to detect output error (retval<0), and for snprintf, truncation (retval>STR_SIZE)
```

Tool	Rule ID	Query
CodeQL	cpp/very-likely-overrunning-write	View source

The program performs a buffer copy or write operation with no upper limit on the size of the copy. By analyzing the bounds of the expressions involved, it appears that certain inputs will cause a buffer overflow to occur in this case. In addition to causing program instability, techniques exist which may allow an attacker to use this vulnerability to execute arbitrary code.

First detected in commit e42fa3a 4 days ago

cmake.yml fixed path — Verified — e42fa3a

positives.c:22 on branch main

Severity: Critical

Affected branches: main

Tags: reliability, security

Weaknesses:
- CWE-120
- CWE-787
- CWE-805

Buffer overflow

- Heap overflow
- In general exploitable in a number of ways..
- However here it’s not dependent on user input => low impact

Comments

- Clang-tidy analysis takes by longest to run, tries different code paths
- Semgrep only warned on unsafe functions, very naive, but it’s easy to create custom warnings.
- Clang-tidy is smart when it comes to std libraries, but limited for other projects
  - e.g. it knows printf shouldn’t take a null ptr.
    But for an unknown external function the same is not certain,
    in which case that external code would need to be scanned as well (which in turn gives more false positives ..) to detect the same error.
    Try to use standard functions.