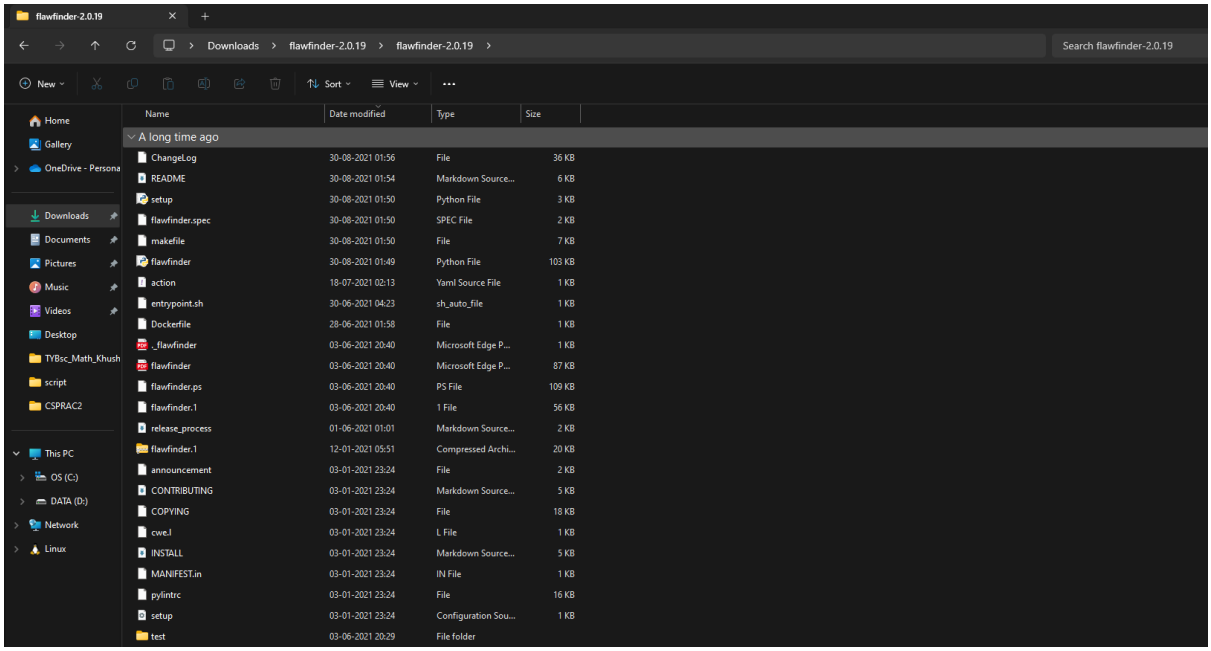
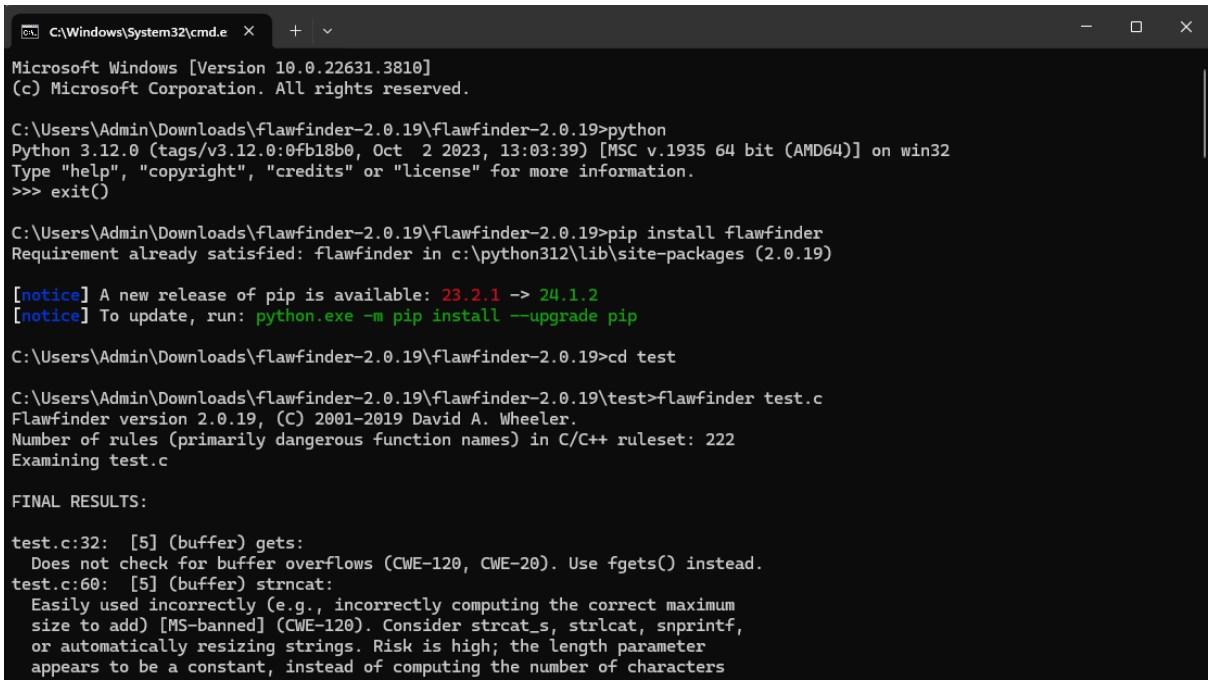
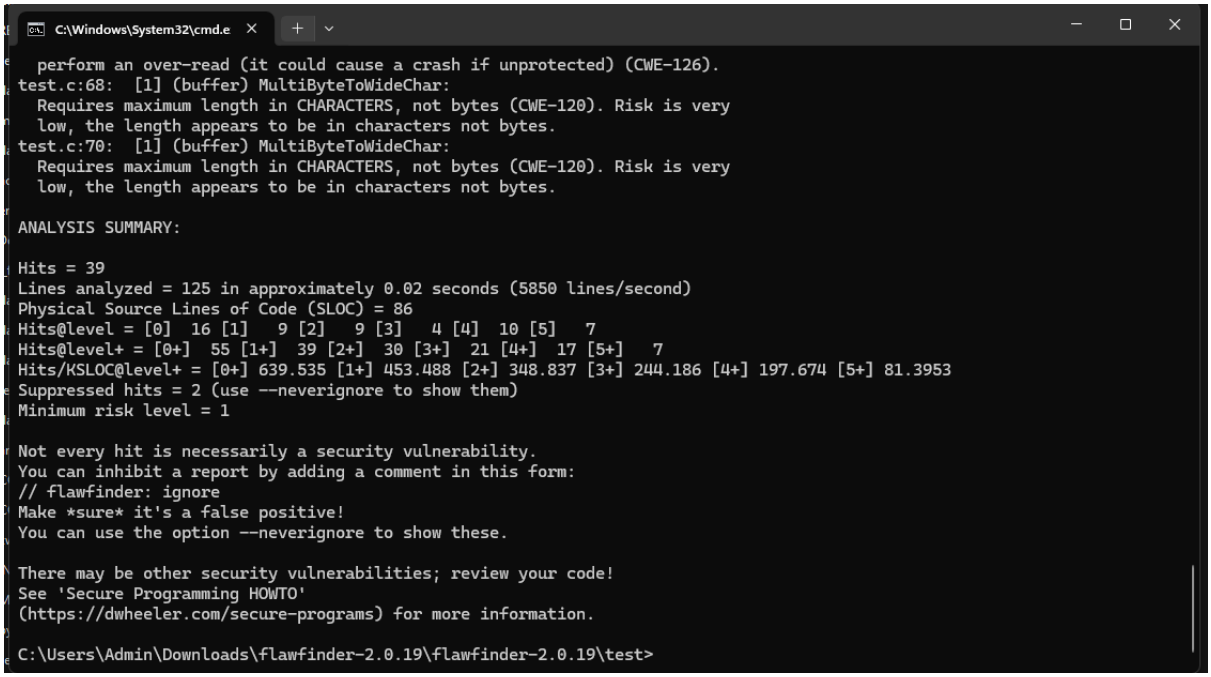
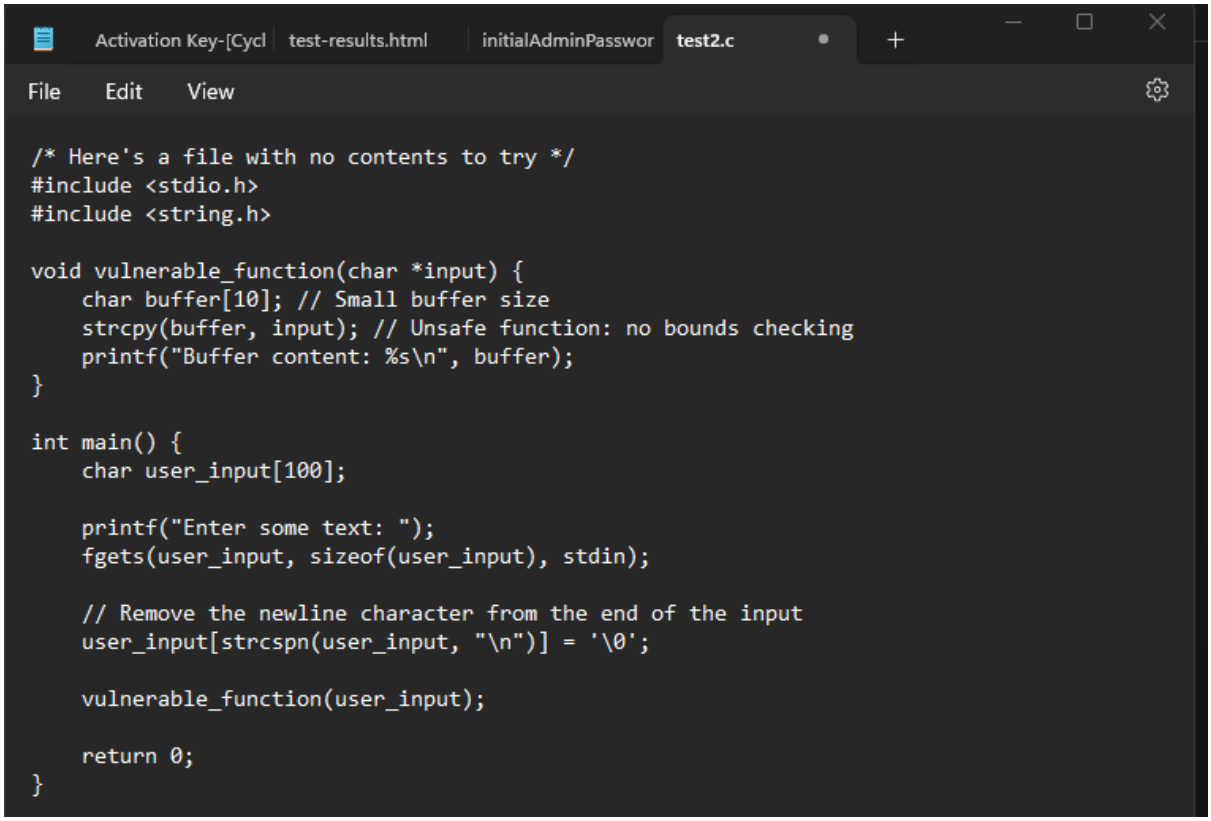

flawfinder-2.0.19
Downloads › flawfinder-2.0.19 › flawfinder-2.0.19 ›
Search flawfinder-2.0.19
New
Sort
View
Home
Gallery
OneDrive - Persona
Downloads
Documents
Pictures
Music
Videos
Desktop
TYBsc_Math_Khush
script
CSPRAC2
This PC
OS (C:)
DATA (D:)
Network
Linux
Name
Date modified
Type
Size
A long time ago
ChangeLog 30-08-2021 01:56 File 36 KB
README 30-08-2021 01:54 Markdown Source... 6 KB
setup 30-08-2021 01:50 Python File 3 KB
flawfinder.spec 30-08-2021 01:50 SPEC File 2 KB
makefile 30-08-2021 01:50 File 7 KB
flawfinder 30-08-2021 01:49 Python File 103 KB
action 18-07-2021 02:13 Yaml Source File 1 KB
entrypoint.sh 30-06-2021 04:23 sh_auto_file 1 KB
Dockerfile 28-06-2021 01:58 File 1 KB
._flawfinder 03-06-2021 20:40 Microsoft Edge P... 1 KB
flawfinder 03-06-2021 20:40 Microsoft Edge P... 87 KB
flawfinder.ps 03-06-2021 20:40 PS File 109 KB
flawfinder.1 03-06-2021 20:40 1 File 56 KB
release_process 01-06-2021 01:01 Markdown Source... 2 KB
flawfinder.1 12-01-2021 05:51 Compressed Archi... 20 KB
announcement 03-01-2021 23:24 File 2 KB
CONTRIBUTING 03-01-2021 23:24 Markdown Source... 5 KB
COPYING 03-01-2021 23:24 File 18 KB
cwe.l 03-01-2021 23:24 L File 1 KB
INSTALL 03-01-2021 23:24 Markdown Source... 5 KB
MANIFEST.in 03-01-2021 23:24 IN File 1 KB
pylintrc 03-01-2021 23:24 File 16 KB
setup 03-01-2021 23:24 Configuration Sou... 1 KB
test 03-06-2021 20:29 File folder


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.3810]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19>python
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct 2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19>pip install flawfinder
Requirement already satisfied: flawfinder in c:\python312\lib\site-packages (2.0.19)
[notice] A new release of pip is available: 23.2.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19>cd test
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>flawfinder test.c
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining test.c
FINAL RESULTS:
test.c:32: [5] (buffer) gets:
Does not check for buffer overflows (CWE-120, CWE-20). Use fgets() instead.
test.c:60: [5] (buffer) strncat:
Easily used incorrectly (e.g., incorrectly computing the correct maximum
size to add) [MS-banned] (CWE-120). Consider strcat_s, strlcat, snprintf,
or automatically resizing strings. Risk is high; the length parameter
appears to be a constant, instead of computing the number of characters


C:\Windows\System32\cmd.e
perform an over-read (it could cause a crash if unprotected) (CWE-126).
test.c:68: [1] (buffer) MultiByteToWideChar:
Requires maximum length in CHARACTERS, not bytes (CWE-120). Risk is very
low, the length appears to be in characters not bytes.
test.c:70: [1] (buffer) MultiByteToWideChar:
Requires maximum length in CHARACTERS, not bytes (CWE-120). Risk is very
low, the length appears to be in characters not bytes.
ANALYSIS SUMMARY:
Hits = 39
Lines analyzed = 125 in approximately 0.02 seconds (5850 lines/second)
Physical Source Lines of Code (SLOC) = 86
Hits@level = [0] 16 [1] 9 [2] 9 [3] 4 [4] 10 [5] 7
Hits@level+ = [0+] 55 [1+] 39 [2+] 30 [3+] 21 [4+] 17 [5+] 7
Hits/KSLOC@level+ = [0+] 639.535 [1+] 453.488 [2+] 348.837 [3+] 244.186 [4+] 197.674 [5+] 81.3953
Suppressed hits = 2 (use --neverignore to show them)
Minimum risk level = 1
Not every hit is necessarily a security vulnerability.
You can inhibit a report by adding a comment in this form:
// flawfinder: ignore
Make *sure* it's a false positive!
You can use the option --neverignore to show these.
There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
(https://dwheeler.com/secure-programs) for more information.
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>


Activation Key-[Cycl
test-results.html
initialAdminPasswor
test2.c
File
Edit
View
/* Here's a file with no contents to try */
#include <stdio.h>
#include <string.h>
void vulnerable_function(char *input) {
char buffer[10]; // Small buffer size
strcpy(buffer, input); // Unsafe function: no bounds checking
printf("Buffer content: %s\n", buffer);
}
int main() {
char user_input[100];
printf("Enter some text: ");
fgets(user_input, sizeof(user_input), stdin);
// Remove the newline character from the end of the input
user_input[strcspn(user_input, "\n")] = '\0';
vulnerable_function(user_input);
return 0;
}

```
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>flawfinder test2.c
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining test2.c

FINAL RESULTS:

test2.c:7:  [4] (buffer) strcpy:
  Does not check for buffer overflows when copying to destination [MS-banned]
  (CWE-120). Consider using snprintf, strcpy_s, or strlcpy (warning: strncpy
  easily misused).
test2.c:6:  [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.
test2.c:12:  [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.
```

```
ANALYSIS SUMMARY:

Hits = 3
Lines analyzed = 23 in approximately 0.01 seconds (2316 lines/second)
Physical Source Lines of Code (SLOC) = 15
Hits@level = [0]   2 [1]   0 [2]   2 [3]   0 [4]   1 [5]   0
Hits@level+ = [0+]   5 [1+]   3 [2+]   3 [3+]   1 [4+]   1 [5+]   0
Hits/KSLOC@level+ = [0+] 333.333 [1+] 200 [2+] 200 [3+] 66.6667 [4+] 66.6667 [5+]   0
Minimum risk level = 1

Not every hit is necessarily a security vulnerability.
You can inhibit a report by adding a comment in this form:
// flawfinder: ignore
Make *sure* it's a false positive!
You can use the option --neverignore to show these.

There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
(https://dwheeler.com/secure-programs) for more information.

C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>
```

Activation Key-[Cycl | test-results.html | initialAdminPasswor | test2.c

File Edit View

```c
#include <stdio.h>
#include <string.h>

void unsafe_string_copy(char *source) {
    char buffer[50]; // Buffer with fixed size
    strcpy(buffer, source); // Unsafe function: no bounds checking
    printf("Copied string: %s\n", buffer);
}

int main() {
    char user_input[100];

    printf("Enter a string: ");
    fgets(user_input, sizeof(user_input), stdin);

    // Remove the newline character if present
    user_input[strcspn(user_input, "\n")] = '\0';

    unsafe_string_copy(user_input);

    return 0;
}
```

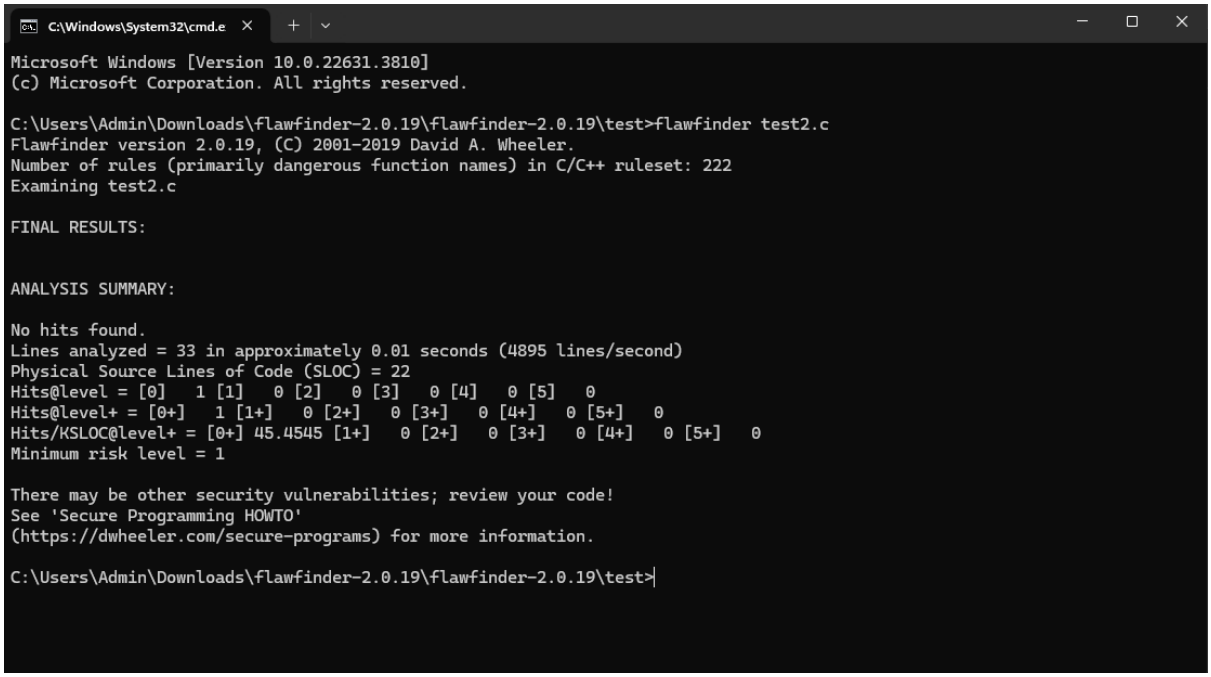
```
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>flawfinder test2.c
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining test2.c

FINAL RESULTS:

test2.c:6:  [4] (buffer) strcpy:
  Does not check for buffer overflows when copying to destination [MS-banned]
  (CWE-120). Consider using snprintf, strcpy_s, or strlcpy (warning: strncpy
  easily misused).
test2.c:5:  [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.
test2.c:11:  [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.
```

```
ANALYSIS SUMMARY:

Hits = 3
Lines analyzed = 22 in approximately 0.01 seconds (3999 lines/second)
Physical Source Lines of Code (SLOC) = 15
Hits@level = [0]   2 [1]   0 [2]   2 [3]   0 [4]   1 [5]   0
Hits@level+ = [0+]   5 [1+]   3 [2+]   3 [3+]   1 [4+]   1 [5+]   0
Hits/KSLOC@level+ = [0+] 333.333 [1+] 200 [2+] 200 [3+] 66.6667 [4+] 66.6667 [5+]   0
Minimum risk level = 1

Not every hit is necessarily a security vulnerability.
You can inhibit a report by adding a comment in this form:
// flawfinder: ignore
Make *sure* it's a false positive!
You can use the option --neverignore to show these.

There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
(https://dwheeler.com/secure-programs) for more information.

C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>
```

Activation Ke | test-results.h | initialAdminF | test2.c | test2.c | test2.

File Edit View

```c
#include <stdio.h>
#include <pthread.h>

#define NUM_THREADS 10
#define NUM_INCREMENTS 1000

// Shared variable
int shared_counter = 0;

void* increment_counter(void* arg) {
    for (int i = 0; i < NUM_INCREMENTS; ++i) {
        shared_counter++; // Unsafe increment operation
    }
    return NULL;
}

int main() {
    pthread_t threads[NUM_THREADS];

    // Create threads
    for (int i = 0; i < NUM_THREADS; ++i) {
        pthread_create(&threads[i], NULL, increment_counter, NULL);
    }

    // Wait for threads to finish
    for (int i = 0; i < NUM_THREADS; ++i) {
        pthread_join(threads[i], NULL);
    }

    printf("Final counter value: %d\n", shared_counter);

    return 0;
}
```


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.3810]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>flawfinder test2.c
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining test2.c
FINAL RESULTS:
ANALYSIS SUMMARY:
No hits found.
Lines analyzed = 33 in approximately 0.01 seconds (4895 lines/second)
Physical Source Lines of Code (SLOC) = 22
Hits@level = [0] 1 [1] 0 [2] 0 [3] 0 [4] 0 [5] 0
Hits@level+ = [0+] 1 [1+] 0 [2+] 0 [3+] 0 [4+] 0 [5+] 0
Hits/KSLOC@level+ = [0+] 45.4545 [1+] 0 [2+] 0 [3+] 0 [4+] 0 [5+] 0
Minimum risk level = 1
There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
(https://dwheeler.com/secure-programs) for more information.
C:\Users\Admin\Downloads\flawfinder-2.0.19\flawfinder-2.0.19\test>