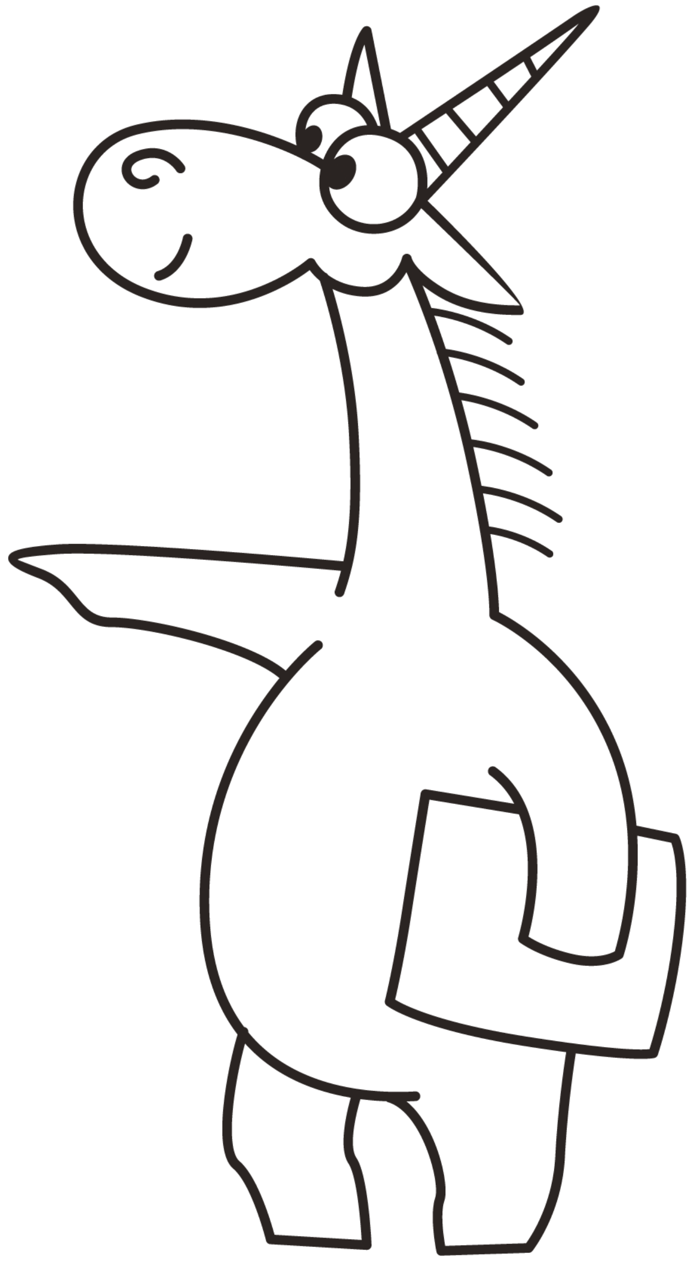
How to write more reliable code?

Tools and best practices for C++

Egor Bredikhin
bredikhin@viva64.com

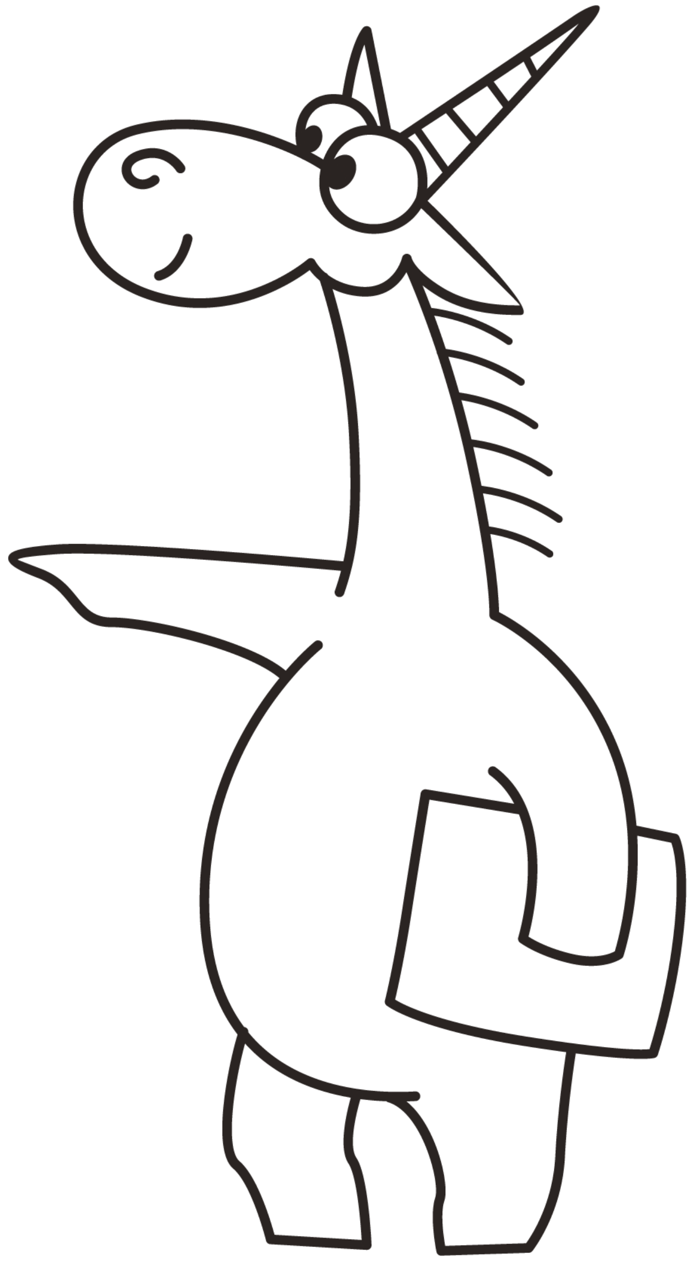
PVS-Studio Team
www.viva64.com

Reliable code?



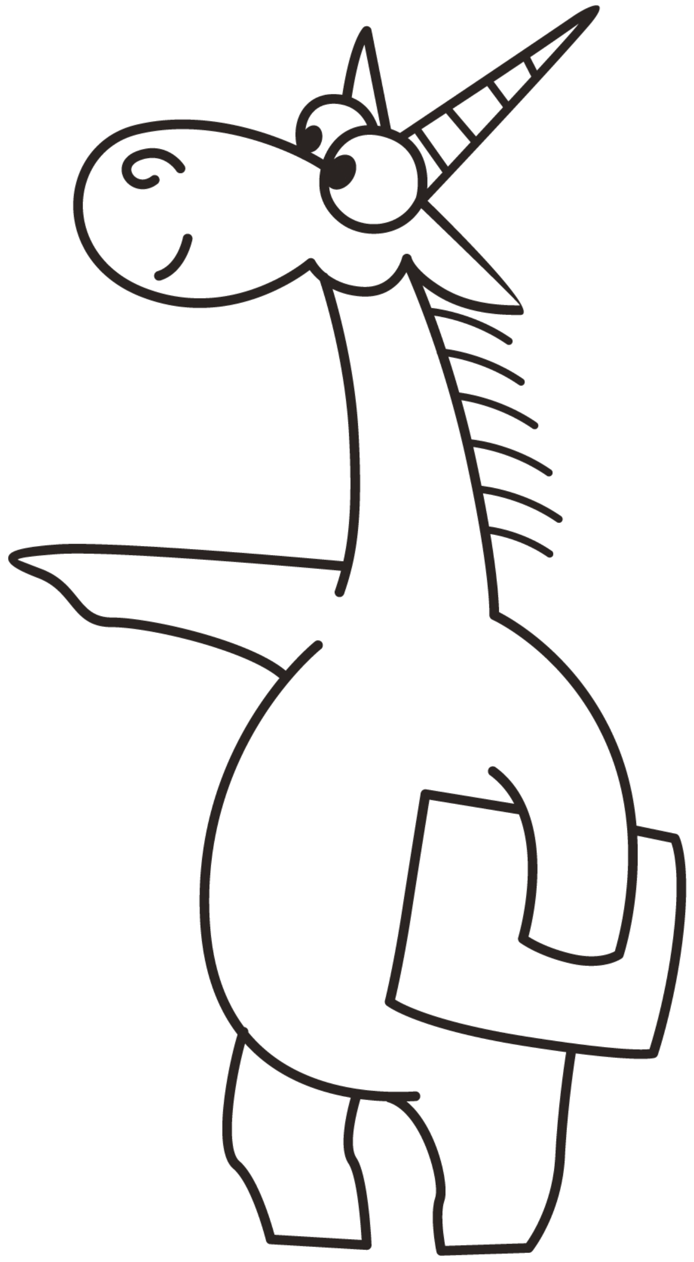
Reliable code?

- Can handle critical situations



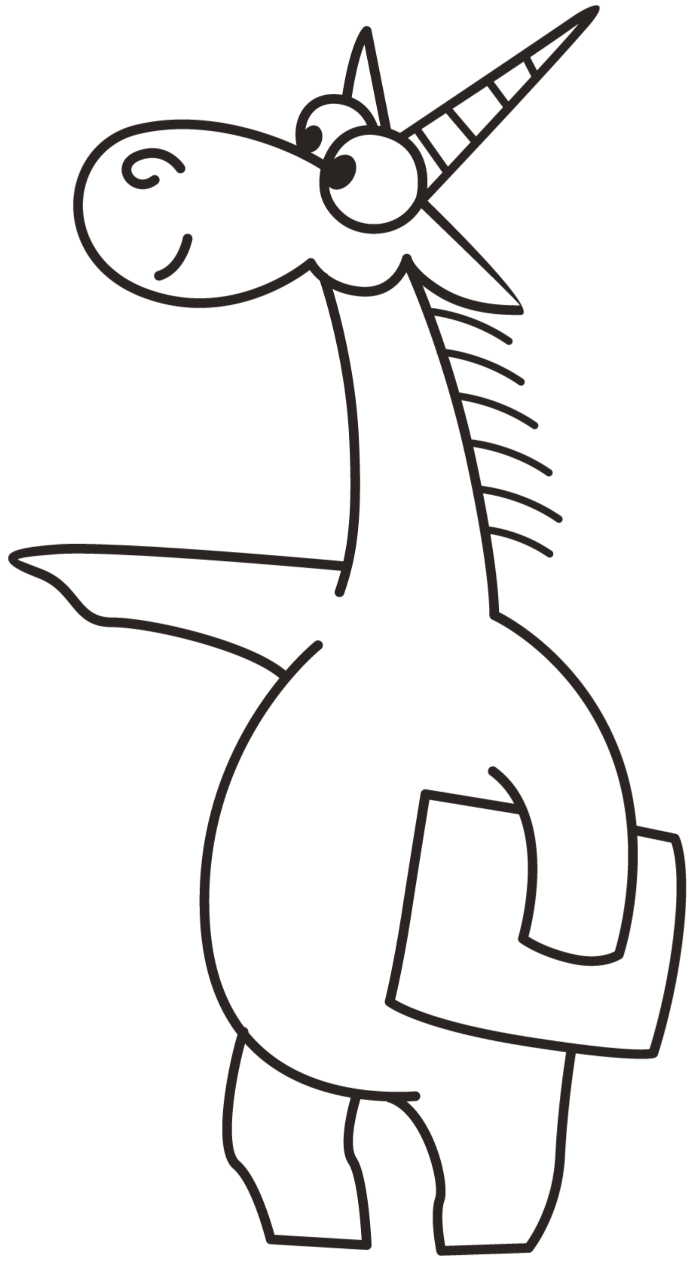
Reliable code?

- Can handle critical situations
- Has no unexpected behaviour



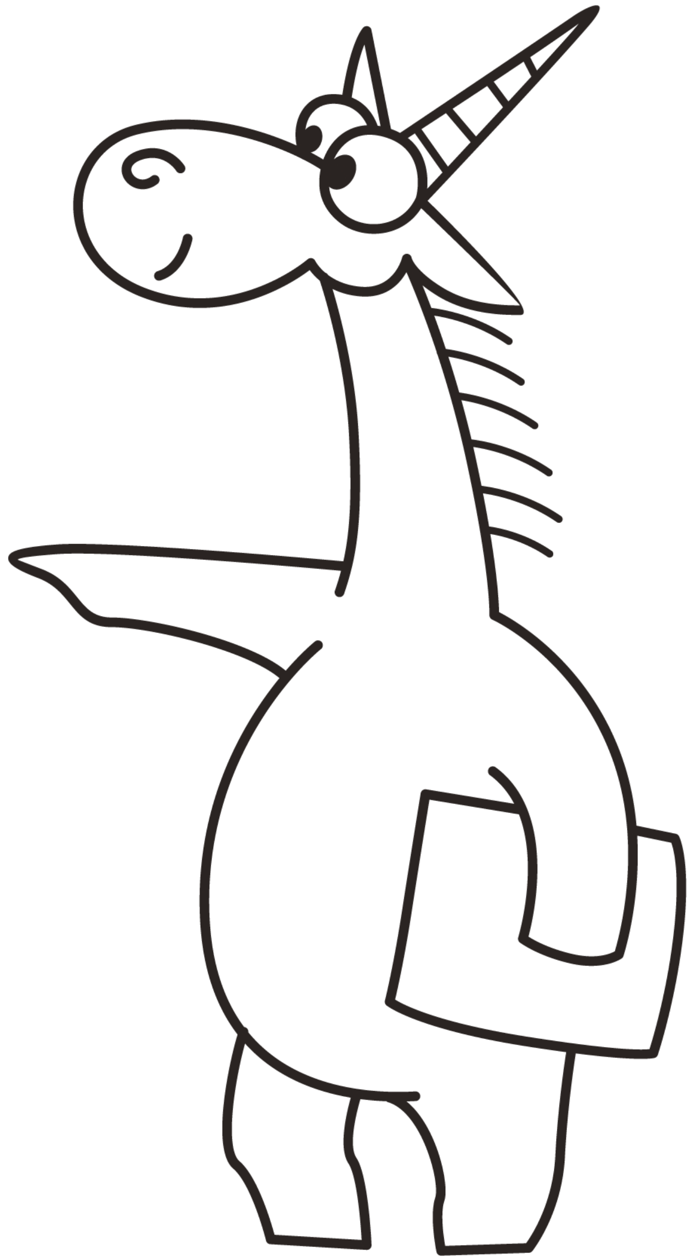
Reliable code?

- Can handle critical situations
- Has no unexpected behaviour
- Secure



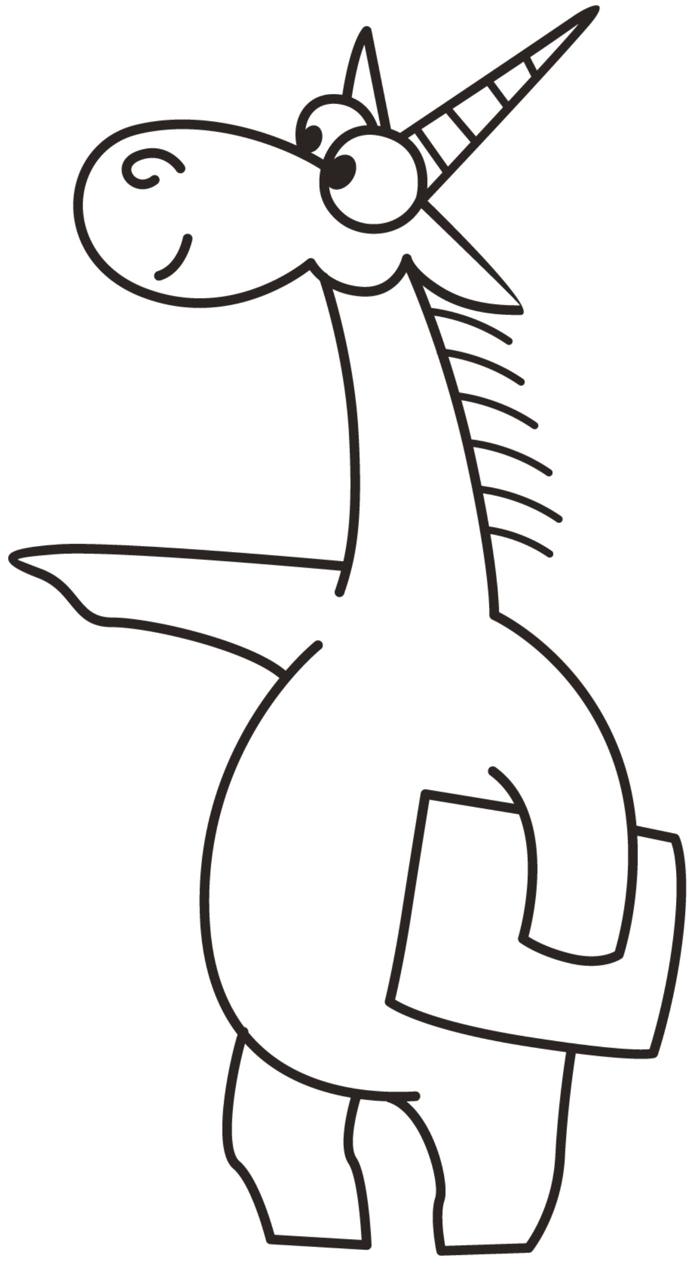
Reliable code?

- Can handle critical situations
- Has no unexpected behaviour
- Secure
- Tested and verified as much as possible



Reliable code?

- Can handle critical situations
- Has no unexpected behaviour
- Secure
- Tested and verified as much as possible
- Easy to maintain



$370 million for an integer overflow

- A data conversion from a **64-bit** floating point number to a **16-bit** signed integer caused a hardware exception.
- The exception halted the inertial reference system, resulting in the destruction of the flight.
- The failure has become known as one of the most infamous and expensive software bugs in history.

https://en.wikipedia.org/wiki/Cluster_(spacecraft)

Microsoft Zune Bug

```
while (days > 365)
{
    if (IsLeapYear(year))
    {
        if (days > 366)
        {
            days -= 366;
            year += 1;
        }
    }
    else
    {
        days -= 365;
        year += 1;
    }
}
```

```
Windows

A fatal exception 0E has occurred at 0157:BFF5DFFF.  The current
application will be terminated.

*  Press any key to terminate the current application.
*  Press CTRL+ALT+DEL again to restart your computer. You will
   lose any unsaved information in all applications.

              Press any key to continue
```

Microsoft Zune Bug

```
while (days > 365) // days == 366 (December 31)
{
    if (IsLeapYear(year)) // year is leap (2008)
    {
        if (days > 366)
        {
            days -= 366;
            year += 1;
        }
        // <= Ooops, we are stuck!
    }
    else
    {
        days -= 365;
        year += 1;
    }
}
```

```
Windows

A fatal exception 0E has occurred at 0167:BFF50FFF.  The current
application will be terminated.

*  Press any key to terminate the current application.
*  Press CTRL+ALT+DEL again to restart your computer. You will
   lose any unsaved information in all applications.

                Press any key to continue
```

MySQL vulnerability

```
typedef char my_bool;

my_bool check_scramble(
  const char *scramble_arg,
  const char *message,
  const uint8 *hash_stage2)
{
  // ...
  return memcmp(hash_stage2, hash_stage2_reassured, SHA1_HASH_SIZE);
}
```

- Because the protocol uses random strings, the probability of hitting this bug is about 1/256.

MySQL vulnerability

```
#!/usr/bin/python
# Title: MySQL Remote Root Authentication Bypass
# Written by : Dave Kennedy(ReL1K)
# http ://www.secmaniac.com

import subprocess

ipaddr = raw_input("Enter the IP address of the mysql server: ")

while 1:
  subprocess.Popen("mysql --host=%s -u root mysql --password=blah" % (ipaddr)).wait()
```

What can I do?

Guidelines

- **C++ Core guidelines (https://github.com/isocpp/CppCoreGuidelines/)**
- Awesome Modern C++ (http://awesomecpp.com/)
- C++ Best Practices (https://github.com/lefticus/cppbestpractices/)
- 42 tips on C++ (https://www.viva64.com/en/b/0391/)
- Google Coding Standard
- MISRA C and MISRA C++
- NASA Software Safety Guidebook
- ….

Manage resources automatically using resource handles and RAII

```cpp
void helper(int i)
{
  if (i < 42)
  {
    throw std::exception("Oops");
  }
  // ...
}

// Bad: possibly leak
void f1(int i)
{
  int* p = new int[12];
  // ...
  helper(i);
  delete[] p;
}
```

```cpp
void helper(int i)
{
  if (i < 42)
  {
    throw std::exception("Oops");
  }
  // ...
}

// OK: resource management done by a handle
void f2(int i)
{
  auto p = std::make_unique<int[]>(12);
  // ...
  helper(i);
}
```

Manage resources automatically using resource handles and RAII

This bug was found in **Far2l**

```
BOOL WINAPI _export SEVENZ_OpenArchive(const char *Name, int *Type)
{
  Traverser *t = new Traverser(Name);
  if (!t->Valid())
  {
    return FALSE;
    delete t;
  }

  delete s_selected_traverser;
  s_selected_traverser = t;
  return TRUE;
}
```

PVS-Studio Warning: V779 Unreachable code detected. It is possible that an error is present. 7z.cpp 203
PVS-Studio Warning: V773 The function was exited without releasing the 't' pointer. A memory leak is possible. 7z.cpp 202

If in doubt about operator precedence, parenthesize

This bug was found in **Linux Kernel**

```
static int nvme_pr_preempt( /* ... */ pr_type type, bool abort)
{
  u32 cdw10 = nvme_pr_type(type) << 8 | abort ? 2 : 1;
  // ...
}
```

PVS-Studio Warning: V502 Perhaps the '?:' operator works in a different way than it was expected. The '?:' operator has a lower priority than the '|' operator. core.c 1046

```
u32 cdw10 = nvme_pr_type(type) << 8 | (abort ? 2 : 1);
```

If in doubt about operator precedence, parenthesize

This bug was found in **Chromium**

```
#define FILE_ATTRIBUTE_DIRECTORY 0x00000010

bool GetPlatformFileInfo(PlatformFile file, PlatformFileInfo* info) {
  info->is_directory = file_info.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY != 0;
  // ...
}
```

PVS-Studio Warning: V564 The '&' operator is applied to bool type value. You've probably forgotten to include parentheses or intended to use the '&&' operator. base platform_file_win.cc 216

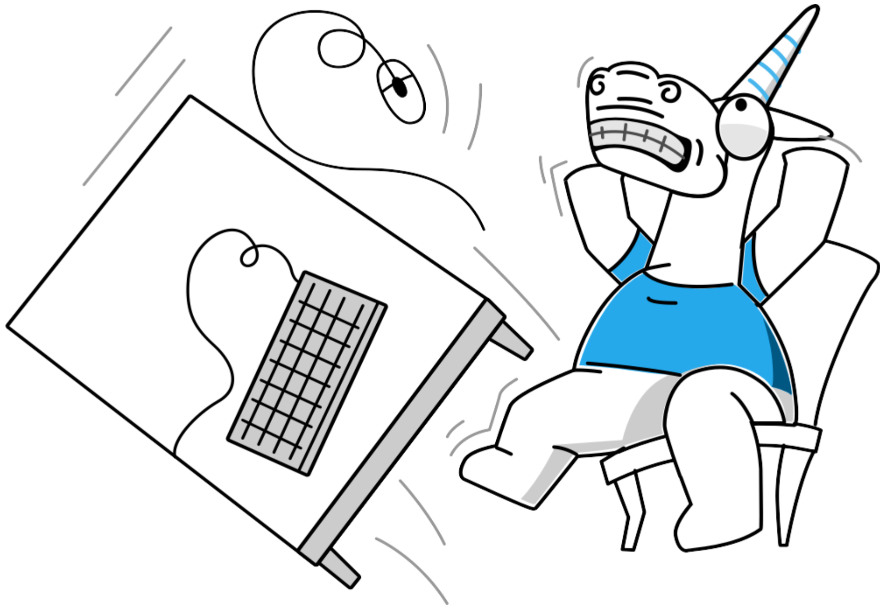
```
info->is_directory = (file_info.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY) != 0;
```

Avoid complicated expressions

This bug was found in **Godot Engine**

```
static real_t out(real_t t, real_t b, real_t c, real_t d)
{
  return c * ((t = t / d - 1) * t * t + 1) + b;
}
```

PVS-Studio Warning: V567 Undefined behavior. The 't' variable is modified while being used twice between sequence points. tween_interpolaters.cpp 265



Don't try to use volatile for synchronization

- volatile does not provide atomicity
- volatile does not synchronize between threads
- volatile does not prevent instruction reordering
- volatile simply has nothing to do with concurrency

Don't try to use volatile for synchronization

```cpp
// Bad
volatile int counter = 42;

void thread_func()
{
  if (counter-- > 0) // data race on counter
  {
    // ...
  }
}
```

```cpp
// OK
std::atomic<int> counter = 42;

void thread_func()
{
  if (counter-- > 0)
  {
    // ...
  }
}
```

Don't try to use volatile for synchronization

```
// OK
// Pointer to a memory-mapped register
unsigned int volatile * const port = (unsigned int *) 0x40000000;

// Write 0x00 three times into a physical port
*port = 0x00;
*port = 0x00;
*port = 0x00;

// Wait until the appropriate bits of a physical port are set
while (!(*port & 0x42)) {};
```

A destructor may not fail

This bug was found in **TortoiseSVN**

```
CCacheFileOutBuffer::~CCacheFileOutBuffer()
{
  // ...
  if (size >= (DWORD)(-1))
    throw CStreamException("stream too large");
}
```

PVS-Studio Warning: V509 The 'throw' operator inside the destructor should be placed within the try..catch block. Raising exception inside the destructor is illegal. cachefileoutbuffer.cpp 52

Don't use macros for program text manipulation
Don't use macros for constants or "functions"

This bug was found in **Linux Kernel**

```
#define CFS_FAIL_TIMEOUT(id, secs) \
  cfs_fail_timeout_set(id, 0, secs * 1000, CFS_FAIL_LOC_NOSET)

int ptl_send_rpc(struct ptlrpc_request *request, int noreply)
{
  CFS_FAIL_TIMEOUT(OBD_FAIL_PTLRPC_DELAY_SEND, request->rq_timeout + 5);
  // ...
}
```

Don't use macros for program text manipulation
Don't use macros for constants or "functions"

This bug was found in **Linux Kernel**

```
#define CFS_FAIL_TIMEOUT(id, secs) \
  cfs_fail_timeout_set(id, 0, secs * 1000, CFS_FAIL_LOC_NOSET)

int ptl_send_rpc(struct ptlrpc_request *request, int noreply)
{
  CFS_FAIL_TIMEOUT(OBD_FAIL_PTLRPC_DELAY_SEND, request->rq_timeout + 5);
  // cfs_fail_timeout_set(0x506, 0, request->rq_timeout + 5 * 1000, 0);
  // ...
}
```

PVS-Studio Warning: V733 It is possible that macro expansion resulted in incorrect evaluation order. Check expression: request->rq_timeout + 5 * 1000. niobuf.c 637

Unit testing and TDD

Unit testing and TDD

+ TDD allows to find problems early

Unit testing and TDD

+ TDD allows to find problems early

+ Facilitates changes (regression testing)

Unit testing and TDD

+ TDD allows to find problems early

+ Facilitates changes (regression testing)

+ Encourages the creation of independent components

Unit testing and TDD

+ TDD allows to find problems early

+ Facilitates changes (regression testing)

+ Encourages the creation of independent components

+ Tests act like a documentation

Unit testing and TDD

+ TDD allows to find problems early
+ Facilitates changes (regression testing)
+ Encourages the creation of independent components
+ Tests act like a documentation

- You cannot test every execution path

Unit testing and TDD

+ TDD allows to find problems early
+ Facilitates changes (regression testing)
+ Encourages the creation of independent components
+ Tests act like a documentation

- You cannot test every execution path
- Writing good realistic tests is time and effort consuming

Unit testing and TDD

+ TDD allows to find problems early
+ Facilitates changes (regression testing)
+ Encourages the creation of independent components
+ Tests act like a documentation

- You cannot test every execution path
- Writing good realistic tests is time and effort consuming
- Some problems cannot be easily tested

Unit testing and TDD

+ TDD allows to find problems early
+ Facilitates changes (regression testing)
+ Encourages the creation of independent components
+ Tests act like a documentation

- You cannot test every execution path
- Writing good realistic tests is time and effort consuming
- Some problems cannot be easily tested
- Platform differences

Unit testing and TDD

+ TDD allows to find problems early

+ Facilitates changes (regression testing)

+ Encourages the creation of independent components

+ Tests act like a documentation

- You cannot test every execution path

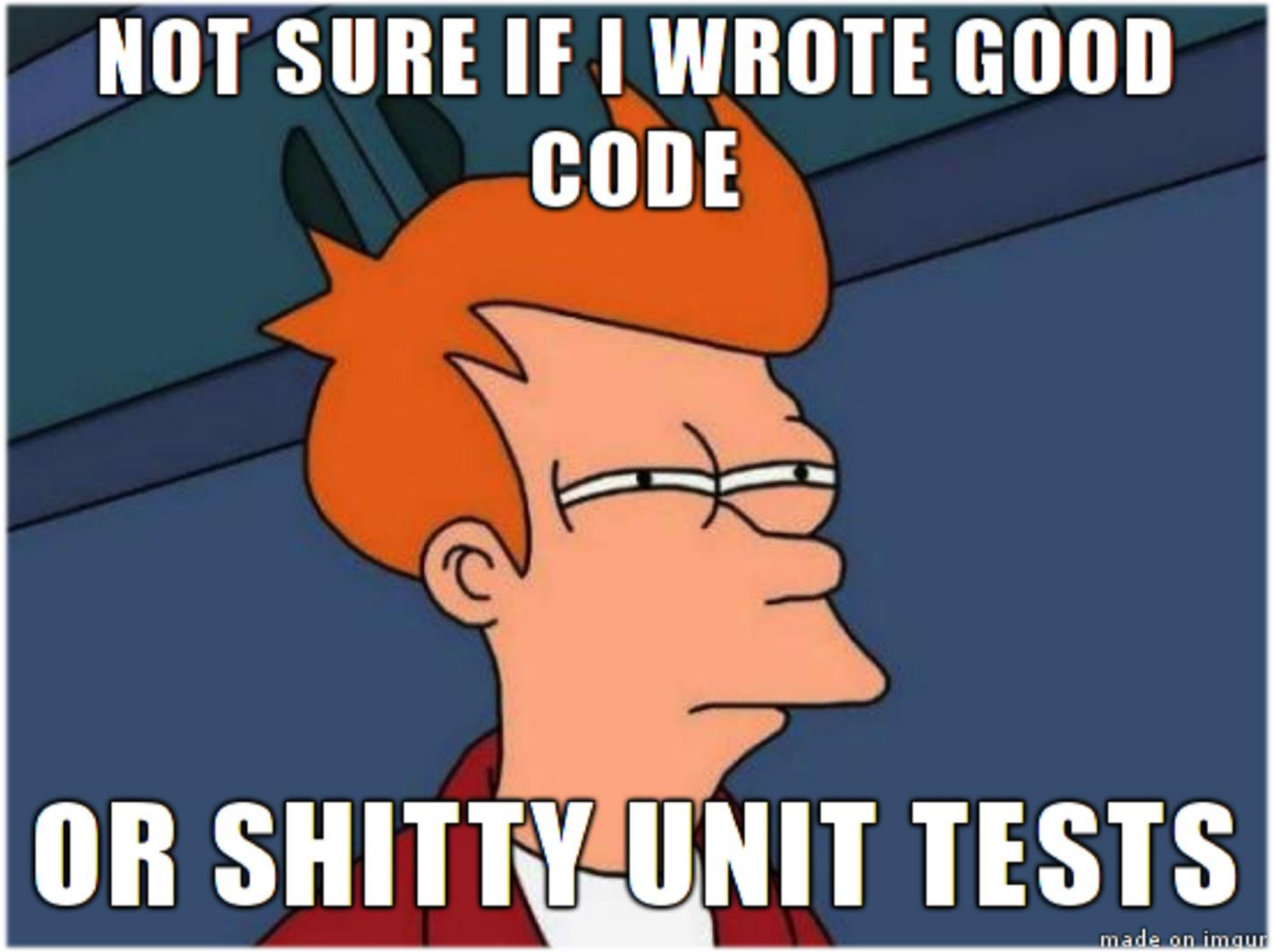
- Writing good realistic tests is time and effort consuming

- Some problems cannot be easily tested

- Platform differences

- Code for unit tests is at least as buggy as the code it is testing

Errors In Unit Tests



Errors In Unit Tests

This bug was found in **Chromium**

```
TEST(SharedMemoryTest, MultipleThreads) {
    // ...
    int threadcounts[] = { 1, kNumThreads };
    for (size_t i = 0; i < sizeof(threadcounts) / sizeof(threadcounts); i++) {
      // ...
  }
}
```

PVS-Studio Warning: V501 There are identical sub-expressions to the left and to the right of the '/' operator: sizeof (threadcounts) / sizeof (threadcounts)

Errors In Unit Tests

This bug was found in **OpenDDS**

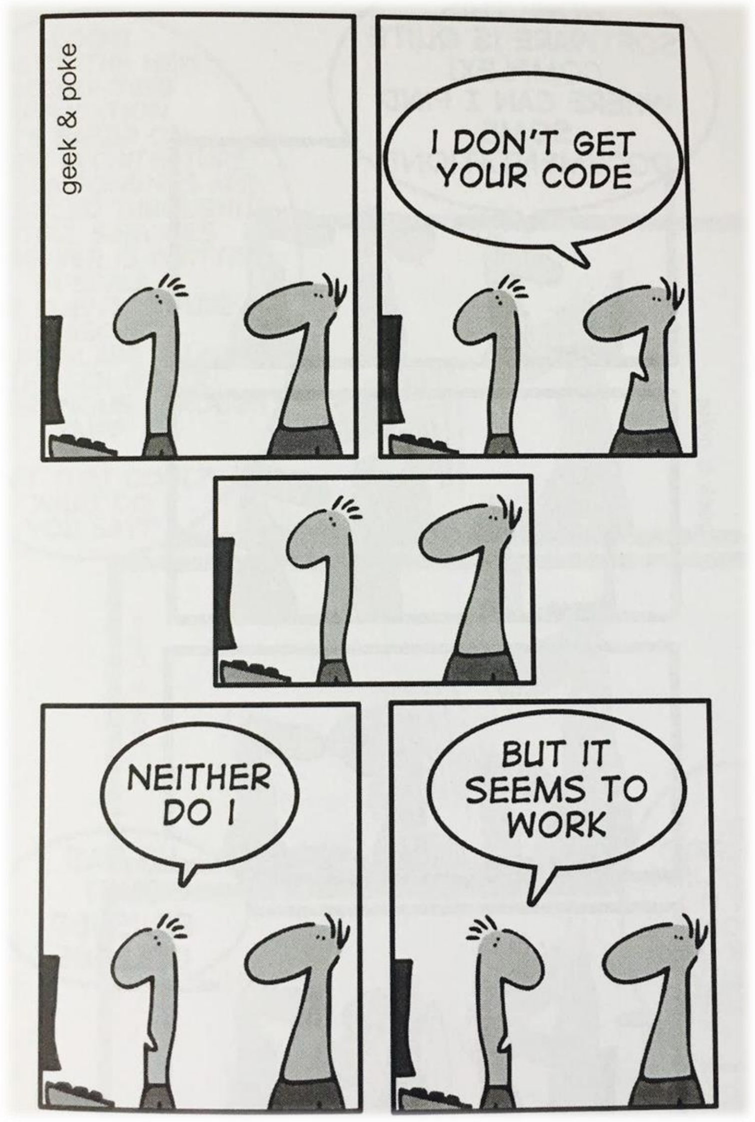
```
TEST_ASSERT(dsqp.service_cleanup_delay.sec = 4);
TEST_ASSERT(dsqp.service_cleanup_delay.nanosec = 2000);
TEST_ASSERT(dsqp.history_kind == KEEP_LAST_HISTORY_QOS);
TEST_ASSERT(dsqp.history_depth == 172);
TEST_ASSERT(dsqp.max_samples == 389);
TEST_ASSERT(dsqp.max_instances == 102);
TEST_ASSERT(dsqp.max_samples_per_instance == 20);
```

PVS-Studio Warning: V559 Suspicious assignment inside the condition expression of 'if' operator: dsqp.service_cleanup_delay.sec = 4. ut_parameterlistconverter.cpp 1295

Tools

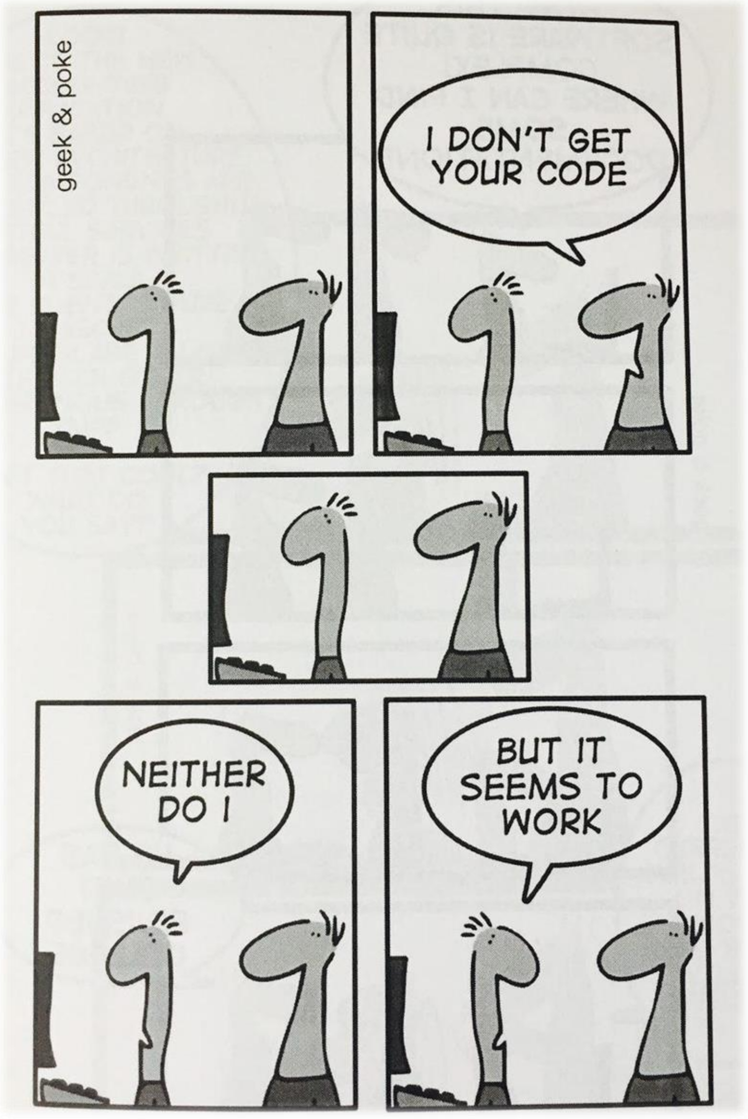
- Google Test
- Boost.Test
- CppUnit
- Catch
- QTest
- …

Code review


geek & poke
I DON'T GET YOUR CODE
NEITHER DO I
BUT IT SEEMS TO WORK

Code review

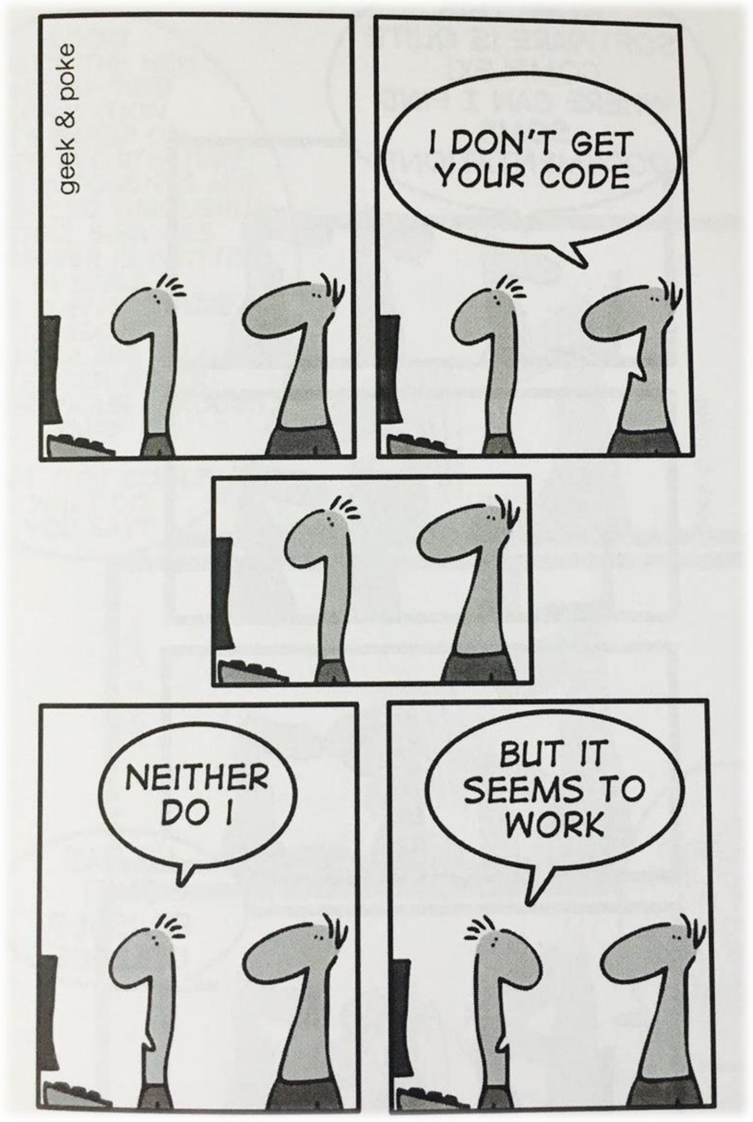
+ Complicated errors could be detected



Code review

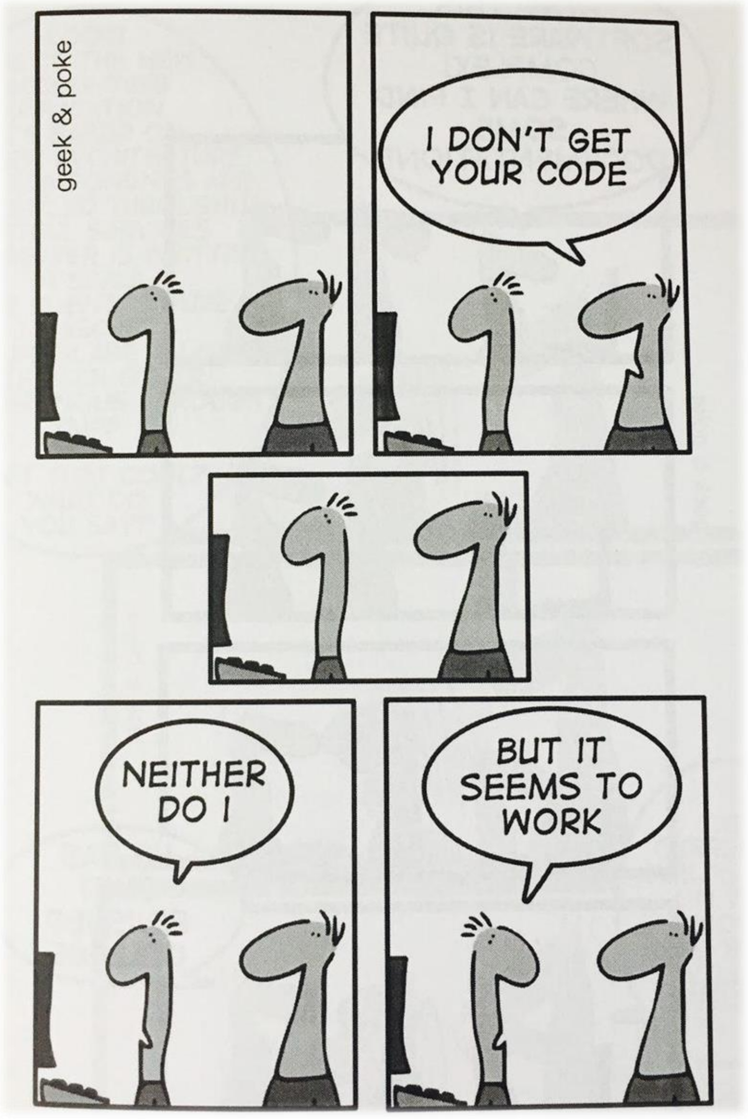
+ Complicated errors could be detected

+ A better solution could be proposed



Code review

+ Complicated errors could be detected
+ A better solution could be proposed
+ Educates team members



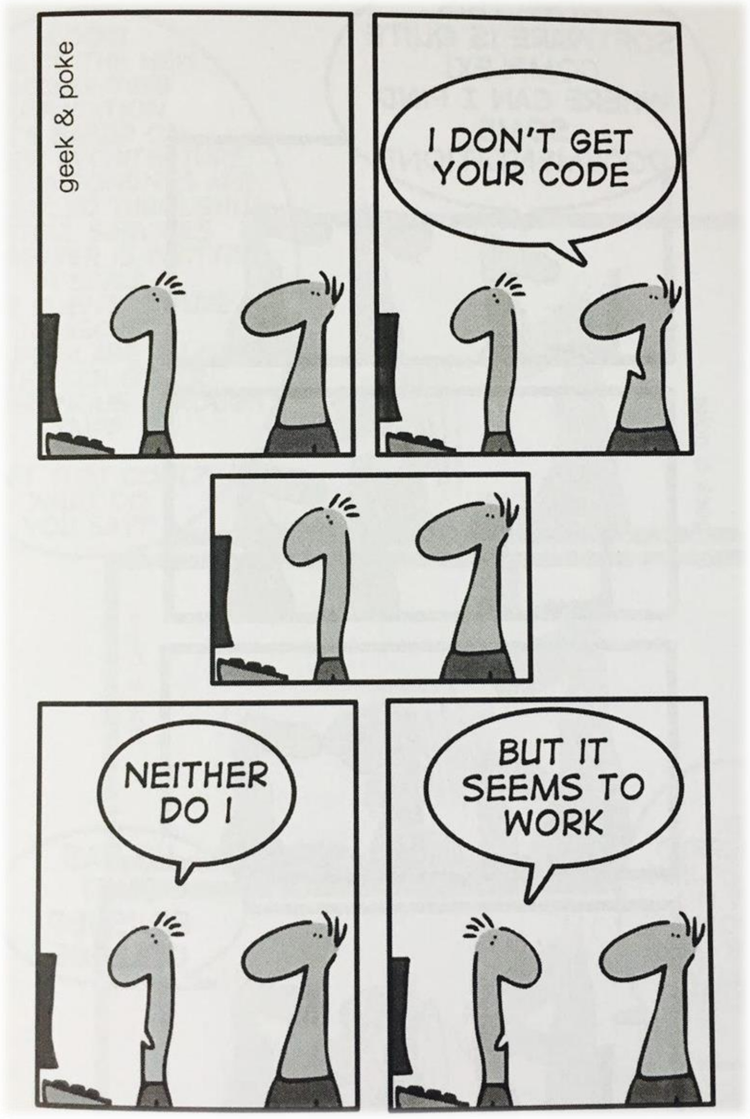
Code review

+ Complicated errors could be detected

+ A better solution could be proposed

+ Educates team members

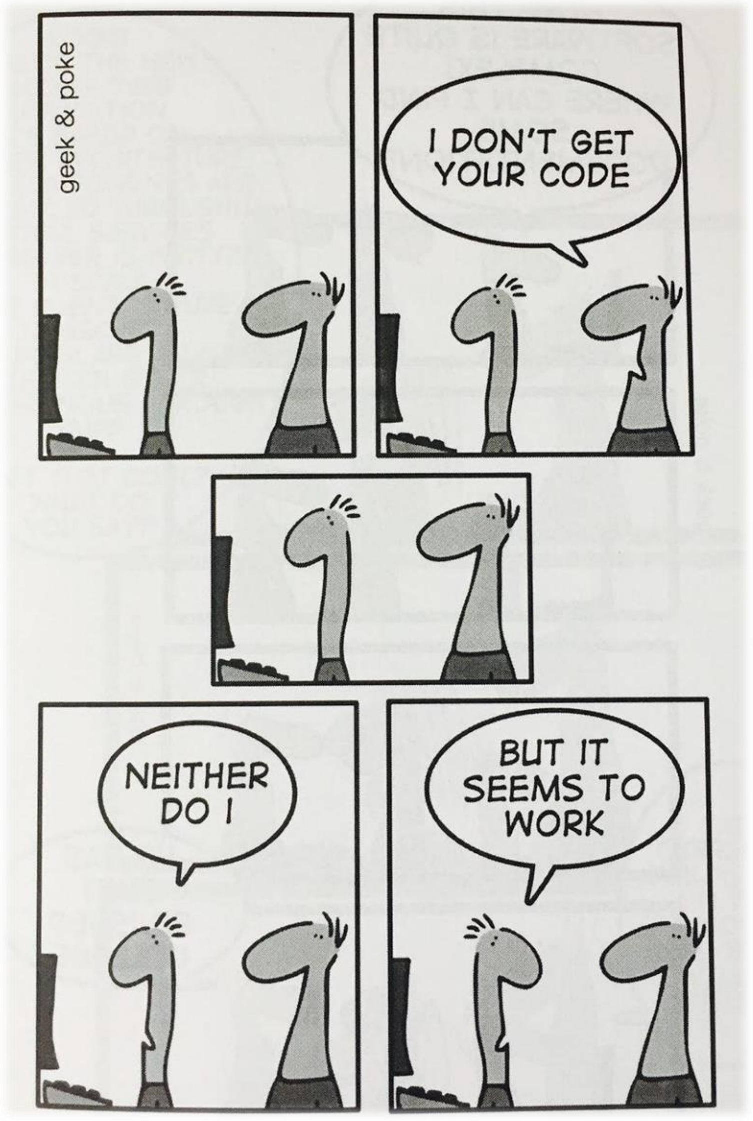
- High cost



Code review

+ Complicated errors could be detected
+ A better solution could be proposed
+ Educates team members

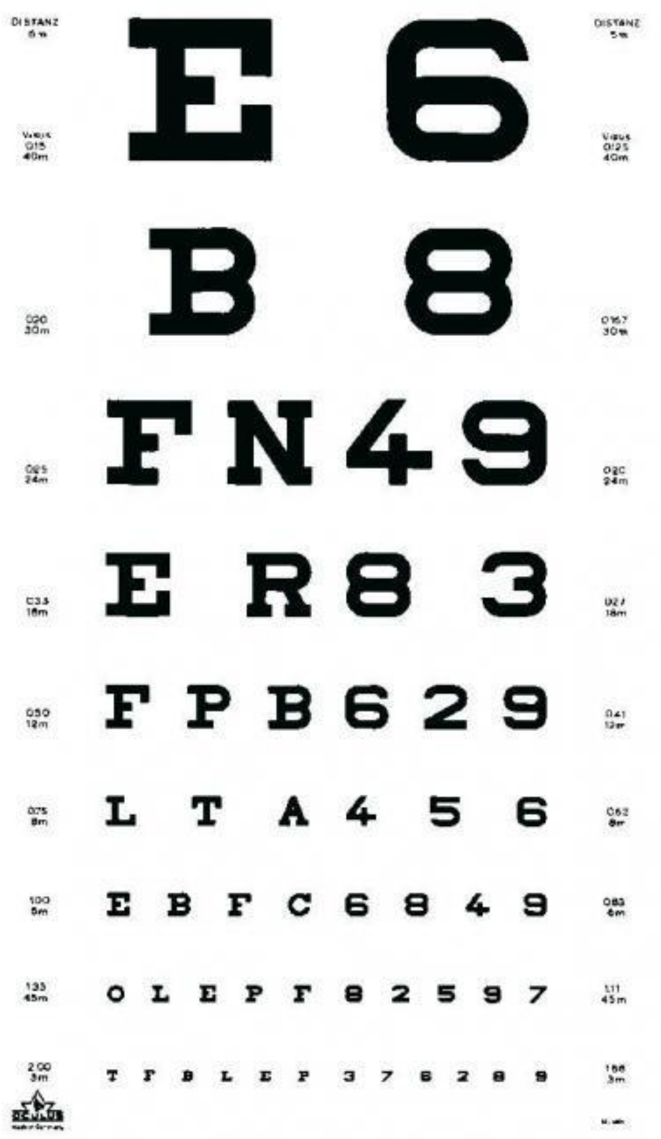
- High cost
- Human reliability



Human reliability

This bug was found in **OpenSSL**

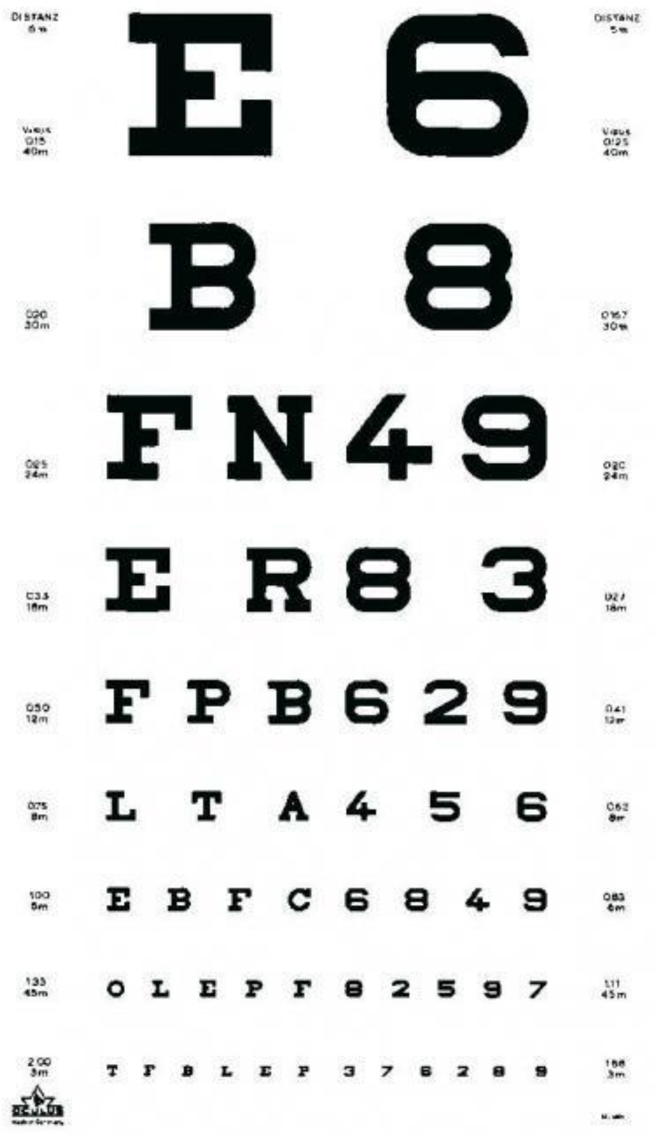
```
if (!strncmp(vstart,"ASCII",5))
  arg->format = ASN1_GEN_FORMAT_ASCII;
else if (!strncmp(vstart,"UTF8",4))
  arg->format = ASN1_GEN_FORMAT_UTF8;
else if (!strncmp(vstart,"HEX",3))
  arg->format = ASN1_GEN_FORMAT_HEX;
else if (!strncmp(vstart,"BITLIST",3))
  arg->format = ASN1_GEN_FORMAT_BITLIST;
else
  ...
```



Human reliability

This bug was found in **OpenSSL**

```
if (!strncmp(vstart,"ASCII",5))
  arg->format = ASN1_GEN_FORMAT_ASCII;
else if (!strncmp(vstart,"UTF8",4))
  arg->format = ASN1_GEN_FORMAT_UTF8;
else if (!strncmp(vstart,"HEX",3))
  arg->format = ASN1_GEN_FORMAT_HEX;
else if (!strncmp(vstart,"BITLIST",3))
  arg->format = ASN1_GEN_FORMAT_BITLIST;
else
  ...
```



Human reliability

This bug was found in **MySQL**

```
static int rr_cmp(uchar *a, uchar *b)
{
  if (a[0] != b[0])
    return (int)a[0] - (int)b[0];
  if (a[1] != b[1])
    return (int)a[1] - (int)b[1];
  if (a[2] != b[2])
    return (int)a[2] - (int)b[2];
  if (a[3] != b[3])
    return (int)a[3] - (int)b[3];
  if (a[4] != b[4])
    return (int)a[4] - (int)b[4];
  if (a[5] != b[5])
    return (int)a[1] - (int)b[5];
  if (a[6] != b[6])
    return (int)a[6] - (int)b[6];
  return (int)a[7] - (int)b[7];
}
```

Static Analysis

Static Analysis

+ Covers all program branches

Static Analysis

+ Covers all program branches

+ Detects misprints, copy-paste errors, and stuff

Static Analysis

+ Covers all program branches

+ Detects misprints, copy-paste errors, and stuff

+ Performs control-flow and data-flow analysis

Static Analysis

+ Covers all program branches
+ Detects misprints, copy-paste errors, and stuff
+ Performs control-flow and data-flow analysis
+ Platform independent

Static Analysis

+ Covers all program branches

+ Detects misprints, copy-paste errors, and stuff

+ Performs control-flow and data-flow analysis

+ Platform independent

+ Some analyzers have a knowledge base

Static Analysis

+ Covers all program branches
+ Detects misprints, copy-paste errors, and stuff
+ Performs control-flow and data-flow analysis
+ Platform independent
+ Some analyzers have a knowledge base

- False positives

Static Analysis

+ Covers all program branches

+ Detects misprints, copy-paste errors, and stuff

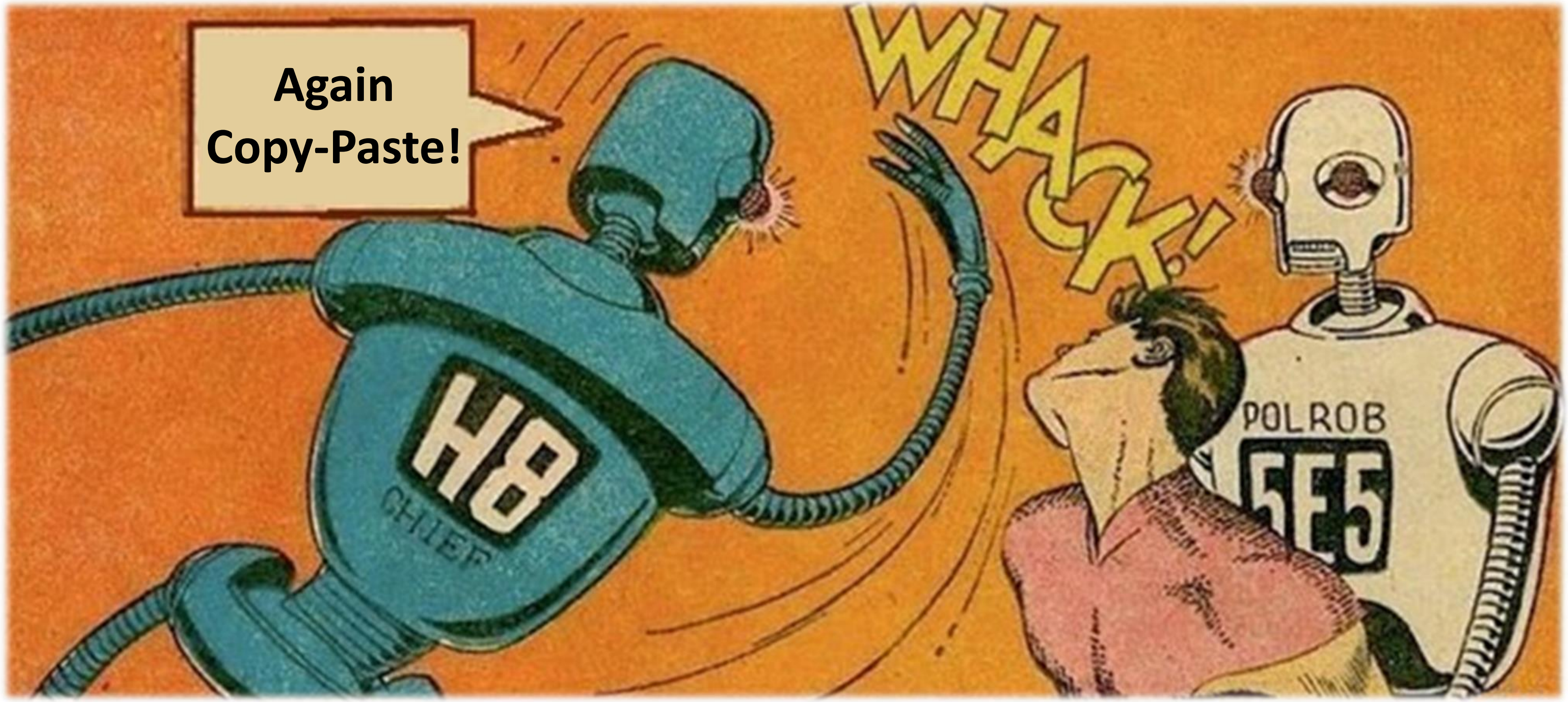
+ Performs control-flow and data-flow analysis

+ Platform independent

+ Some analyzers have a knowledge base

- False positives
- Not so good in detecting memory leaks, parallel errors

Static Analysis in a nutshell



Misprints

This bug was found in **Inkscape**

```
void GlyphComboBox::update(SPFont* spfont)
{
  if (!spfont) return
  //TODO: figure out why do we need to append("")
  //before clearing items properly...

  //Gtk is refusing to clear the combobox
  //when I comment out this line
  this->append("");
  this->remove_all();
}
```

PVS-Studio Warning: V504 It is highly probable that the semicolon ';' is missing after 'return' keyword. svg-fonts-dialog.cpp 167

Misprints

This bug was found in **Clang**

```
Sema::DeduceTemplateArguments(/* ... */)
{
    if ((P->isPointerType() && A->isPointerType()) ||
      (P->isMemberPointerType() && P->isMemberPointerType()))
    // ...
}
```

PVS-Studio Warning: V501 There are identical sub-expressions 'P->isMemberPointerType ()' to the left and to the right of the '&&' operator. clangSema sematemplatededuction.cpp 3240

Misprints

This bug was found in **MySQL**

```
mysqlx::XProtocol* active()
{
  if (!active_connection)
    std::runtime_error("no active session");
  return active_connection.get();
}
```

PVS-Studio Warning: V596 The object was created but it is not being used. The 'throw' keyword could be missing: throw runtime_error(…); mysqlxtest.cc 509

Control-flow analysis

This bug was found in **Amazon Lumberyard**

```
Events::ProcessingResult FbxSkinImporter::ImportSkin(/* ... */)
{
  // ...
  if (BuildSceneMeshFromFbxMesh(/* ... */) {
    context.m_createdData.push_back(std::move(createdData));
    return Events::ProcessingResult::Success;
  } else {
    return Events::ProcessingResult::Failure;
  }
  context.m_createdData.push_back();
  return Events::ProcessingResult::Success;
}
```

PVS-Studio Warning: V779 Unreachable code detected. It is possible that an error is present. fbxskinimporter.cpp 67

Data-flow analysis

This bug was found in **Qt**

```
QV4::ReturnedValue QQuickJSContext2DPixelData::getIndexed(/* ... */)
{
  // ...
  if (!m)
    return m->engine()->currentContext()->throwTypeError();
}
```

PVS-Studio Warning: V522 Dereferencing of the null pointer 'm' might take place. qquickcontext2d.cpp 3169

Data-flow analysis

This bug was found in **ClickHouse**

```
for (size_t offset = 8; offset <= 24; offset += 8)
{
    if (offset > 0)
        *(out++) = '.';
     // ...
}
```

PVS-Studio Warning: V547 Expression 'offset > 0' is always true. FunctionsCoding.h 649

Data-flow analysis

This bug was found in **Notepad++**

```
int encodings[] = { 1250, 1251, 1252, /* ... */ };

for (int i = 0; i <= sizeof(encodings) / sizeof(int); i++)
{
  int cmdID = em->getIndexFromEncoding(encodings[i]);
  // ...
}
```

PVS-Studio Warning: V557 Array overrun is possible. The value of 'i' index could reach 46. Notepad++ preferencedlg.cpp 984

Knowledge base

This bug was found in **ReactOS**

```
static void _Stl_loc_combine_names(_Locale_impl* L,
  const char* name1, const char* name2, locale::category c)
{
  if ((c & locale::all) == 0 || strcmp(name1, name1) == 0)
    // ...
}
```

PVS-Studio Warning: V549 The first argument of 'strcmp' function matches it's the second argument. stlport locale.cpp 211

Tools

- Clang Static Analyzer & Clang Tidy
- CppCheck
- Coverity
- Klockwork
- PVS-Studio
- …

Dynamic Analysis

Dynamic Analysis

+ Detects leaks and memory corruption

Dynamic Analysis

+ Detects leaks and memory corruption

+ Detects parallel errors (race conditions, deadlocks and stuff)

Dynamic Analysis

+ Detects leaks and memory corruption

+ Detects parallel errors (race conditions, deadlocks and stuff)

+ Detects uninitialized variables

Dynamic Analysis

+ Detects leaks and memory corruption

+ Detects parallel errors (race conditions, deadlocks and stuff)

+ Detects uninitialized variables

+ Almost no false positives

Dynamic Analysis

+ Detects leaks and memory corruption
+ Detects parallel errors (race conditions, deadlocks and stuff)
+ Detects uninitialized variables
+ Almost no false positives

- Still needs a set of tests to run on

Dynamic Analysis

+ Detects leaks and memory corruption
+ Detects parallel errors (race conditions, deadlocks and stuff)
+ Detects uninitialized variables
+ Almost no false positives

- Still needs a set of tests to run on
- Checks only executed paths

Dynamic Analysis

+ Detects leaks and memory corruption
+ Detects parallel errors (race conditions, deadlocks and stuff)
+ Detects uninitialized variables
+ Almost no false positives

- Still needs a set of tests to run on
- Checks only executed paths
- Sometimes it’s difficult to trace the exact place in code with an error

Dynamic Analysis

+ Detects leaks and memory corruption
+ Detects parallel errors (race conditions, deadlocks and stuff)
+ Detects uninitialized variables
+ Almost no false positives

- Still needs a set of tests to run on
- Checks only executed paths
- Sometimes it's difficult to trace the exact place in code with an error
- Slow and demanding

Tools

- **Dynamic binary instrumentation**

  Valgrind (Memcheck), Hellgrind
  20x slowdown, Linux/macOS/Solaris/Android

  Dr.Memory
  10x slowdown, Linux/Windows x86 only

  Intel Inspector
  10x-160x slowdown, Linux/Windows/macOS

Tools

- **Compile-time instrumentation**

  Address sanitizer (use after free, use after return, oob, leaks)
  2x slowdown, clang > 3.1, gcc > 4.8, Linux/macOS/FreeBSD/Android/IOS

  Memory sanitizer (uninitialized memory reads)
  1.5x-2.5x slowdown, clang > 3.3, Linux x86_64 only

  Thread sanitizer (data race detector)
  2x-20x slowdown, clang > 3.2, gcc > 4.8, Linux x86_64 only

Tools

- clang++ main.cpp -o main -fsanitize=address
- clang++ main.cpp -o main -fsanitize=memory
- clang++ main.cpp -o main -fsanitize=thread

Dynamic Analysis: Example 1

This bug was found in **Chromium**

```
// WebEmbeddedWorkerStartData.h:
struct WebEmbeddedWorkerStartData {
  WebURL scriptURL;
  WebString userAgent;
  WebEmbeddedWorkerStartMode startMode;
};
```

```
enum WebEmbeddedWorkerStartMode {
  WebEmbeddedWorkerStartModeDontPauseOnStart,
  WebEmbeddedWorkerStartModePauseOnStart
};
```

```
// embedded_worker_dispatcher.cc:
void EmbeddedWorkerDispatcher::OnStartWorker(....) {
  ....
  blink::WebEmbeddedWorkerStartData start_data;
  start_data.scriptURL = script_url;
  start_data.userAgent = base::UTF8ToUTF16(webkit_glue::GetUserAgent(script_url));
  wrapper->worker()->startWorkerContext(start_data);
}
```

Dynamic Analysis: Example 1

This bug was found in **Chromium**

```
// WebEmbeddedWorkerStartData.h:
struct WebEmbeddedWorkerStartData {
  WebURL scriptURL;
  WebString userAgent;
  WebEmbeddedWorkerStartMode startMode;
};
```

```
enum WebEmbeddedWorkerStartMode {
  WebEmbeddedWorkerStartModeDontPauseOnStart,
  WebEmbeddedWorkerStartModePauseOnStart
};
```

```
// embedded_worker_dispatcher.cc:
void EmbeddedWorkerDispatcher::OnStartWorker(....) {
  ....
  blink::WebEmbeddedWorkerStartData start_data;
  start_data.scriptURL = script_url;
  start_data.userAgent = base::UTF8ToUTF16(webkit_glue::GetUserAgent(script_url));
  // start_data.startMode was not initialized!
  wrapper->worker()->startWorkerContext(start_data);
}
```

```
// WebEmbeddedWorkerImpl.cpp:
void WebEmbeddedWorkerImpl::startWorkerContext(const WebEmbeddedWorkerStartData& data) {
  m_workerStartData = data;
  m_mainScriptLoader = Loader::create(
    m_loadingContext.get(),
    data.scriptURL,
    bind(&WebEmbeddedWorkerImpl::onScriptLoaderFinished, this));
}

void WebEmbeddedWorkerImpl::onScriptLoaderFinished() {
  if (m_mainScriptLoader->failed() || m_askedToTerminate) {
    m_workerContextClient->workerContextFailedToStart();
    m_mainScriptLoader.clear();
    return;
  }

  WorkerThreadStartMode startMode =
    (m_workerStartData.startMode == WebEmbeddedWorkerStartModePauseOnStart)
    ? PauseWorkerGlobalScopeOnStart : DontPauseWorkerGlobalScopeOnStart;

  ....
}
```

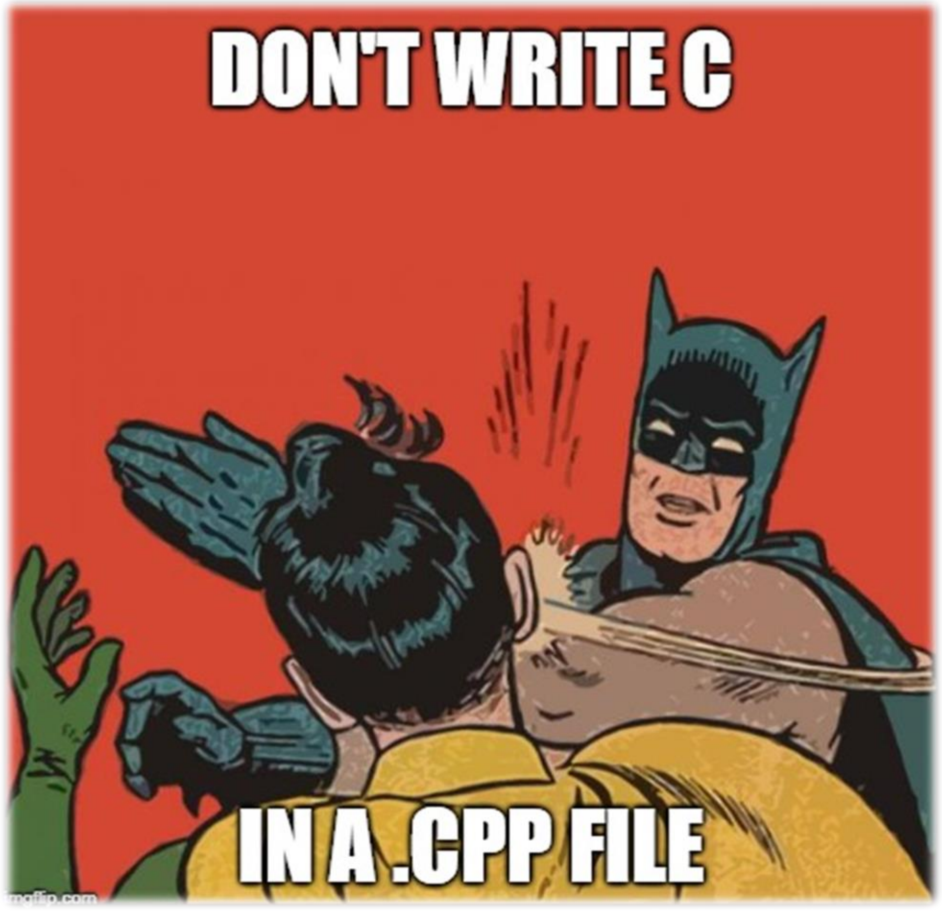
Example of an error that is «invisible» for dynamic analysis

This bug was found in **Valgrind**

```
if (guard->tag == Iex_Const
     && guard->Iex.Const.con->tag == Ico_U1
     && guard->Iex.Const.con->Ico.U1 == True) {
  /* unconditional -- do nothing */
} else {
  goto no_match; //ATC
  cc = iselCondCode( env, guard );
}
```

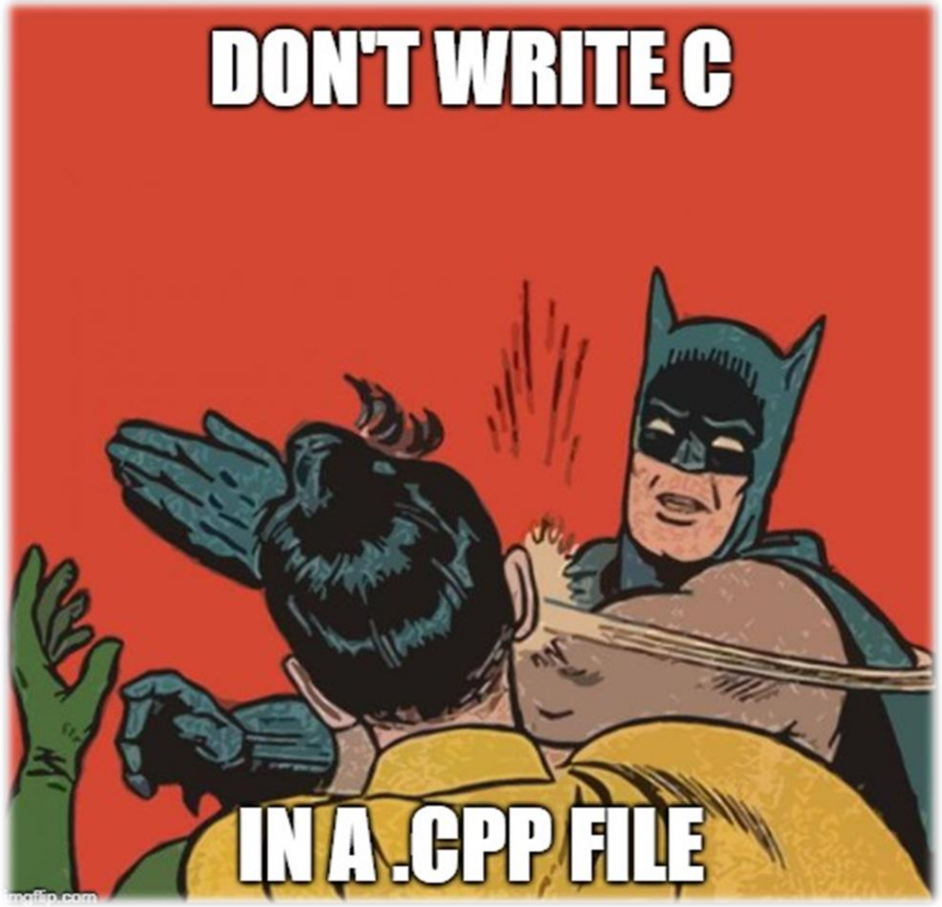
PVS-Studio Warning: V779 Unreachable code detected. It is possible that an error is present. host_arm_isel.c 461

Modern C++ helps to avoid bugs



Modern C++ helps to avoid bugs

- enum class



This bug was found in **Linux Kernel**

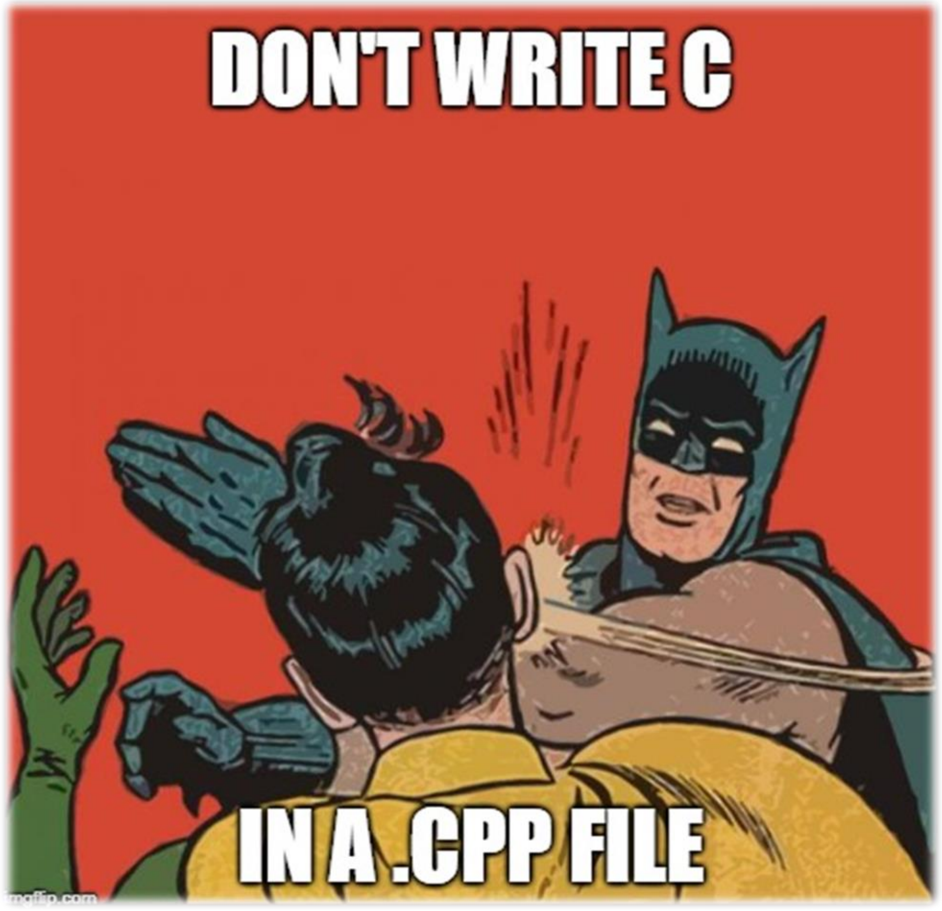
```
enum iscsi_param {
  ISCSI_PARAM_CONN_PORT,
  ISCSI_PARAM_CONN_ADDRESS,
  ISCSI_HOST_PARAM_PORT_STATE,
  // ...
};

enum iscsi_host_param {
  ISCSI_HOST_PARAM_IPADDRESS,
  // ...
};

int iscsi_conn_get_addr_param(iscsi_param param, /* ... */)
{
  // ...
  switch (param) {
  case ISCSI_PARAM_CONN_ADDRESS:
  case ISCSI_HOST_PARAM_IPADDRESS:
  case ISCSI_HOST_PARAM_PORT_STATE:
    // ...
  }
  return len;
}
```

This bug was found in **Linux Kernel**

```
enum iscsi_param {
  ISCSI_PARAM_CONN_PORT,
  ISCSI_PARAM_CONN_ADDRESS,
  ISCSI_HOST_PARAM_PORT_STATE,
  // ...
};

enum iscsi_host_param {
  ISCSI_HOST_PARAM_IPADDRESS,
  // ...
};

int iscsi_conn_get_addr_param(iscsi_param param, /* ... */)
{
  // ...
  switch (param) {
  case ISCSI_PARAM_CONN_ADDRESS:
  case ISCSI_HOST_PARAM_ADDRESS:
  case ISCSI_HOST_PARAM_PORT_STATE:
    // ...
  }
  return len;
}
```

Modern C++ helps to avoid bugs

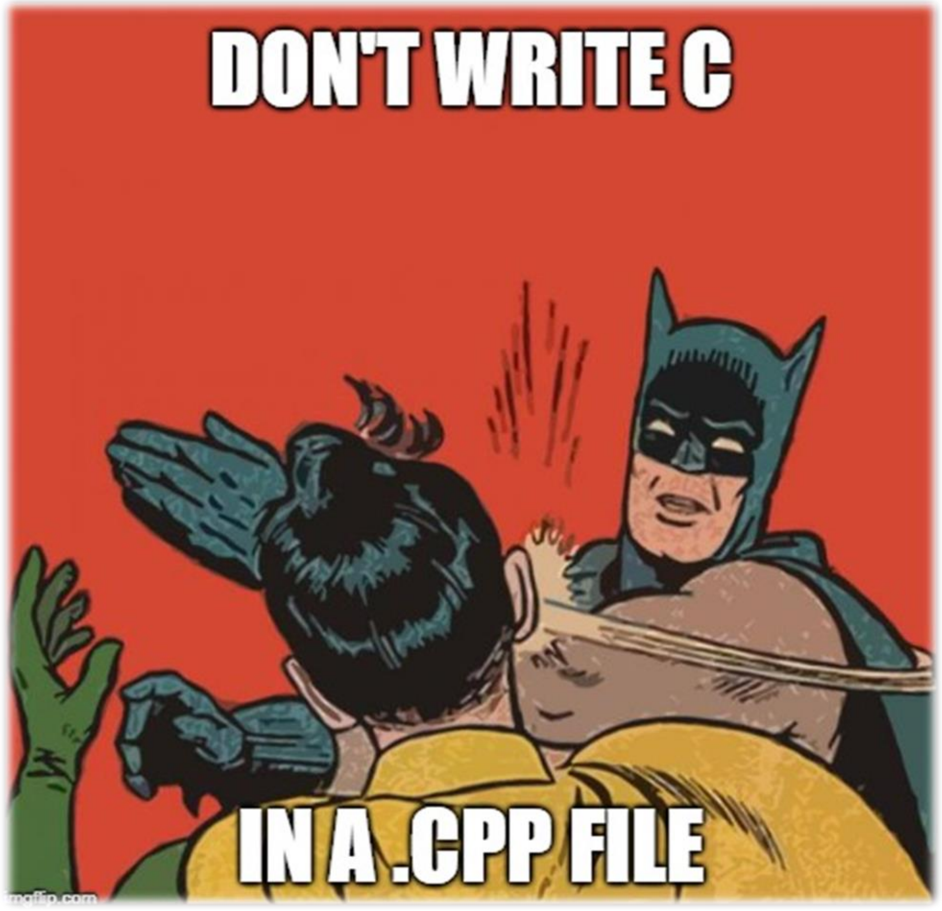
- enum class
- nullptr



```
void Foo(int x, int y, const char *name);

void Foo(int x, int y, int ResourceID);

Foo(1, 2, NULL); // Foo(int, int, int) is called!



HRESULT WinApiFoo(int a, int b);

if (WinApiFoo(a, b) != NULL)    // Bad

if (WinApiFoo(a, b) != nullptr) // Good, compilation error
```

Modern C++ helps to avoid bugs

- enum class
- nullptr
- override, final



This bug was found in **MongoDB**

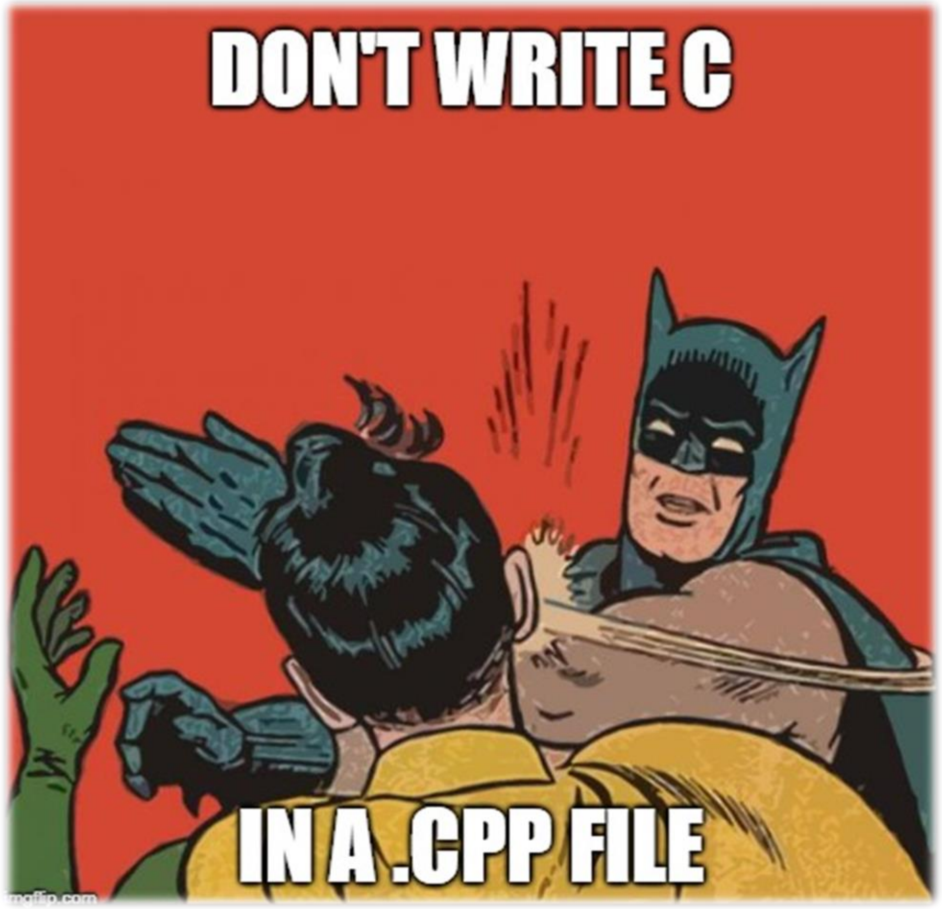
```
class DBClientBase : ....
{
public:
  virtual auto_ptr<DBClientCursor> query(
    const string &ns, Query query, int nToReturn = 0,
    int nToSkip = 0, const BSONObj *fieldsToReturn = 0,
    int queryOptions = 0, int batchSize = 0);
};

class DBDirectClient : public DBClientBase
{
public:
  virtual auto_ptr<DBClientCursor> query(
    const string &ns, Query query, int nToReturn = 0,
    int nToSkip = 0, const BSONObj *fieldsToReturn = 0,
    int queryOptions = 0);
};
```

This bug was found in **MongoDB**

```
class DBClientBase : ....
{
public:
  virtual auto_ptr<DBClientCursor> query(
    const string &ns, Query query, int nToReturn = 0,
    int nToSkip = 0, const BSONObj *fieldsToReturn = 0,
    int queryOptions = 0, int batchSize = 0);
};

class DBDirectClient : public DBClientBase
{
public:
  virtual auto_ptr<DBClientCursor> query(
    const string &ns, Query query, int nToReturn = 0,
    int nToSkip = 0, const BSONObj *fieldsToReturn = 0,
    int queryOptions = 0) override; // Good, compilation error
};
```

Modern C++ helps to avoid bugs

- enum class
- nullptr
- override, final
- constexpr, static_assert

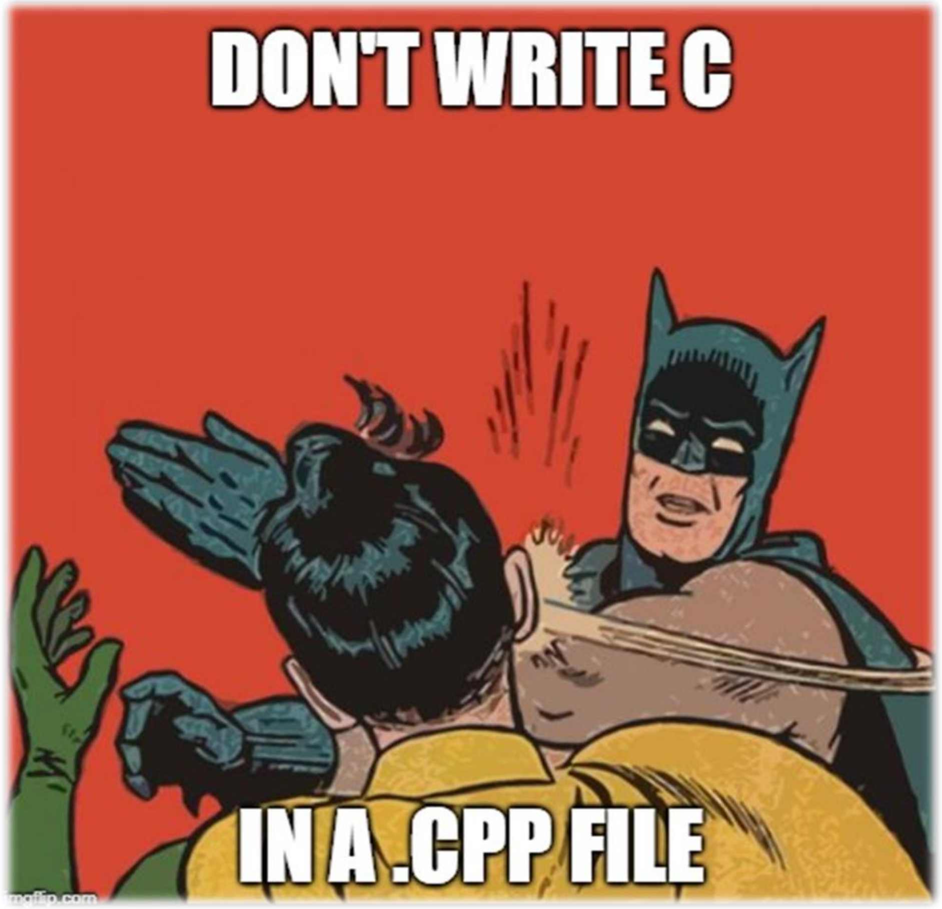


```cpp
// Prehistoric C++
template<int i> struct Factorial {
  static const int result = i * Factorial<i - 1>::result;
};

template<> struct Factorial<1> {
  static const int result = 1;
};

// C++11 (constexpr functions use recursion rather than iteration)
constexpr int factorial(int n) {
  return n ? (n * factorial(n - 1)) : 1;
}

// C++14 (constexpr functions may use local variables and loops)
constexpr int factorial_modern(int n) {
  int res = n;
  while (--n > 0) res *= n;
  return res;
}
```

```cpp
void test() {
  static_assert(Factorial<10>::result == 3628800);
  static_assert(factorial(10) == 3628800);
  static_assert(factorial_modern(10) == 3628800);
}
```

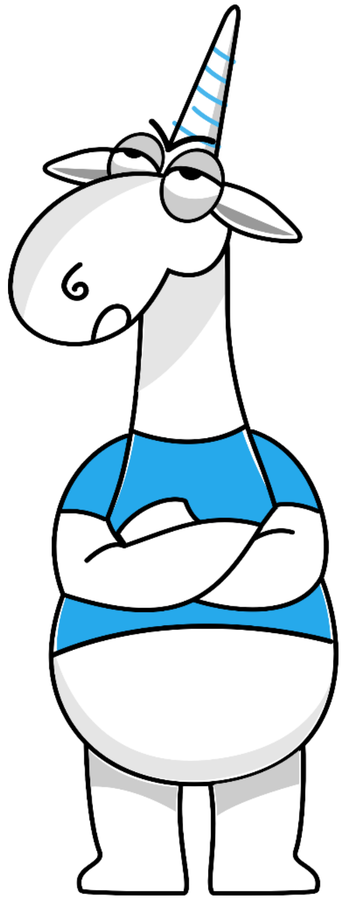
Modern C++ helps to avoid bugs

- enum class
- nullptr
- override, final
- constexpr, static_assert
- [[no_discard]], [[fallthrough]]



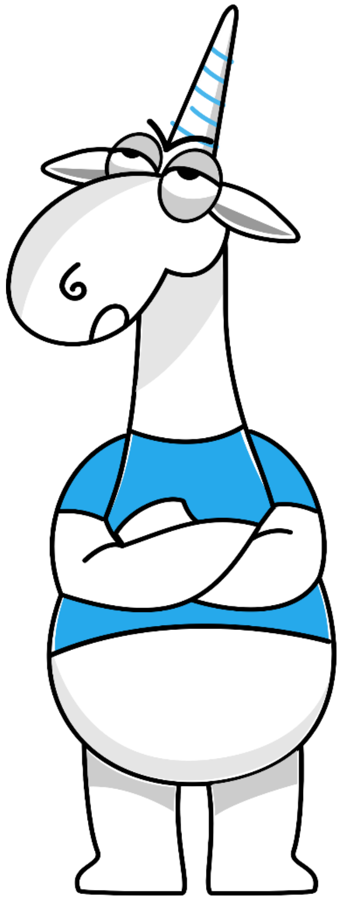
This bug was found in **PVS-Studio :-)**

```
IntegerInterval Clamp(const IntegerInterval &) const noexcept;

ListVirtualValue& ListVirtualValue::RemoveIfPresent(const IntegerInterval &sizeInterval)
{
  if (m_length)
  {
    m_length->min -= 1;
    m_length->Clamp(sizeInterval);
  }
  return *this;
}
```



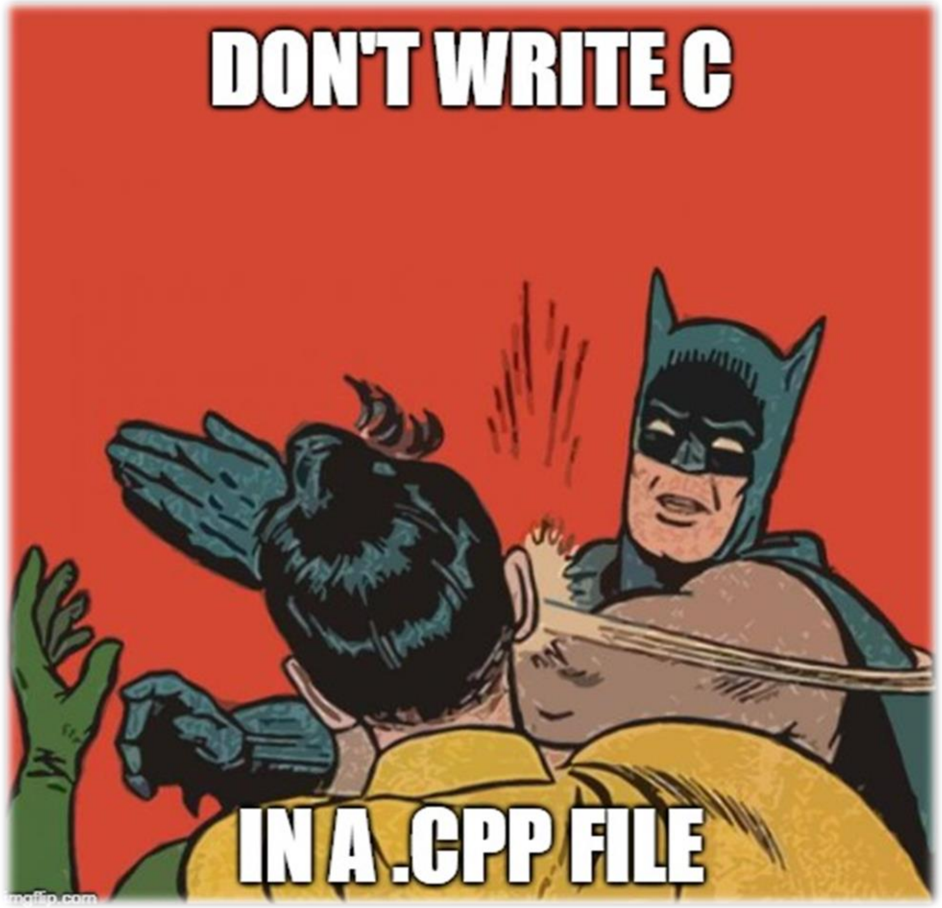
This bug was found in **PVS-Studio :-)**

```
[[nodiscard]] IntegerInterval Clamp(const IntegerInterval &) const noexcept;

ListVirtualValue& ListVirtualValue::RemoveIfPresent(const IntegerInterval &sizeInterval)
{
  if (m_length)
  {
    m_length->min -= 1;
    m_length = m_length->Clamp(sizeInterval);
  }
  return *this;
}
```



Modern C++ helps to avoid bugs

- enum class
- nullptr
- override, final
- constexpr, static_assert
- [[no_discard]], [[fallthrough]]
- STL containers and algorithms



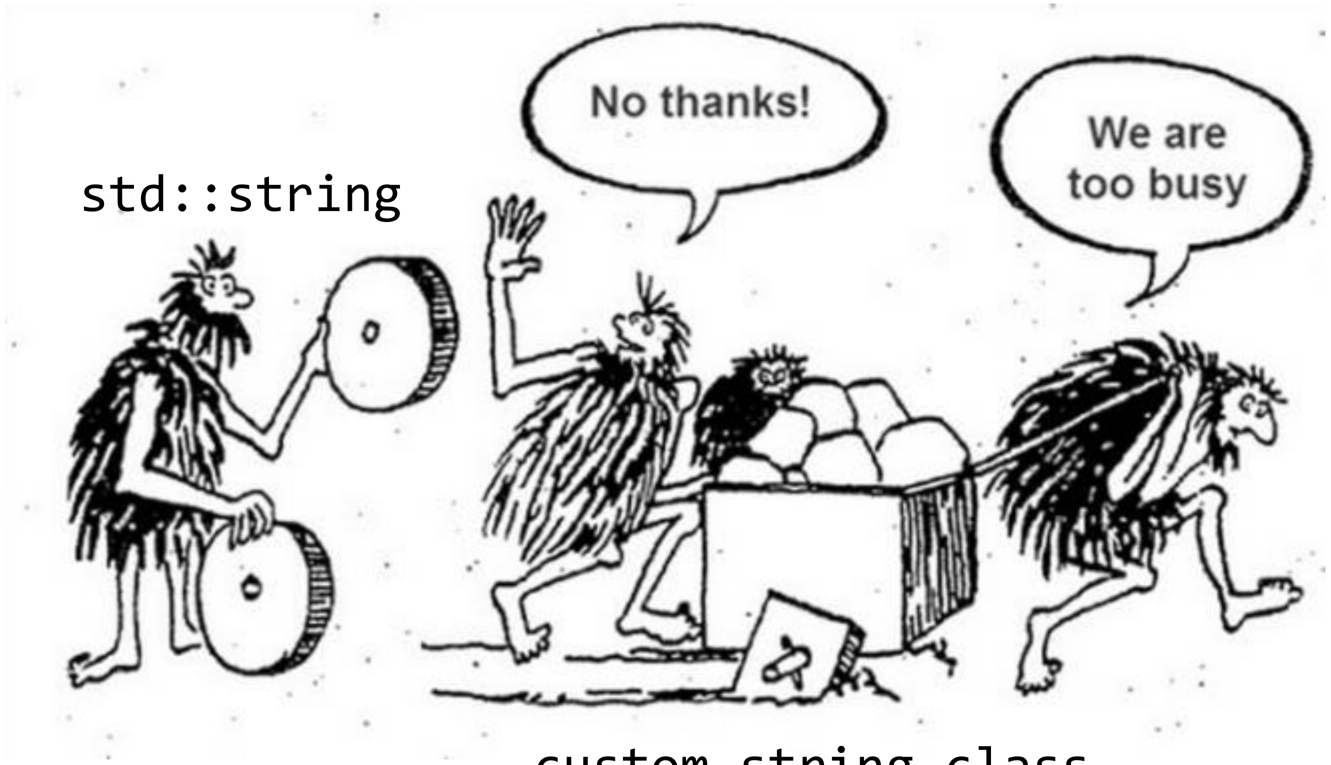
Do not reinvent the wheel

```
std::sort(begin(v), end(v), std::greater<>());
```

“Unless you are an expert in sorting algorithms and have plenty of time, this is more likely to be correct and to run faster than anything you write for a specific application. You need a reason not to use the standard library rather than a reason to use it.”

C++ Core Guidelines ©

Do not reinvent the wheel



Conclusions

- Guidelines
- Unit tests
- Code review
- Static analysis
- Dynamic analysis
- Modern C++

Questions?

Egor Bredikhin: bredikhin@viva64.com

PVS-Studio site: https://www.viva64.com

Twitter: @Code_Analysis