PVS-Studio in 2019

Windows, Linux, macOS

C, C++, C#, Java

OOO «Program Verification Systems»

Website: www.viva64.com

Contacts: support@viva64.com

PVS-Studio static code analyzer

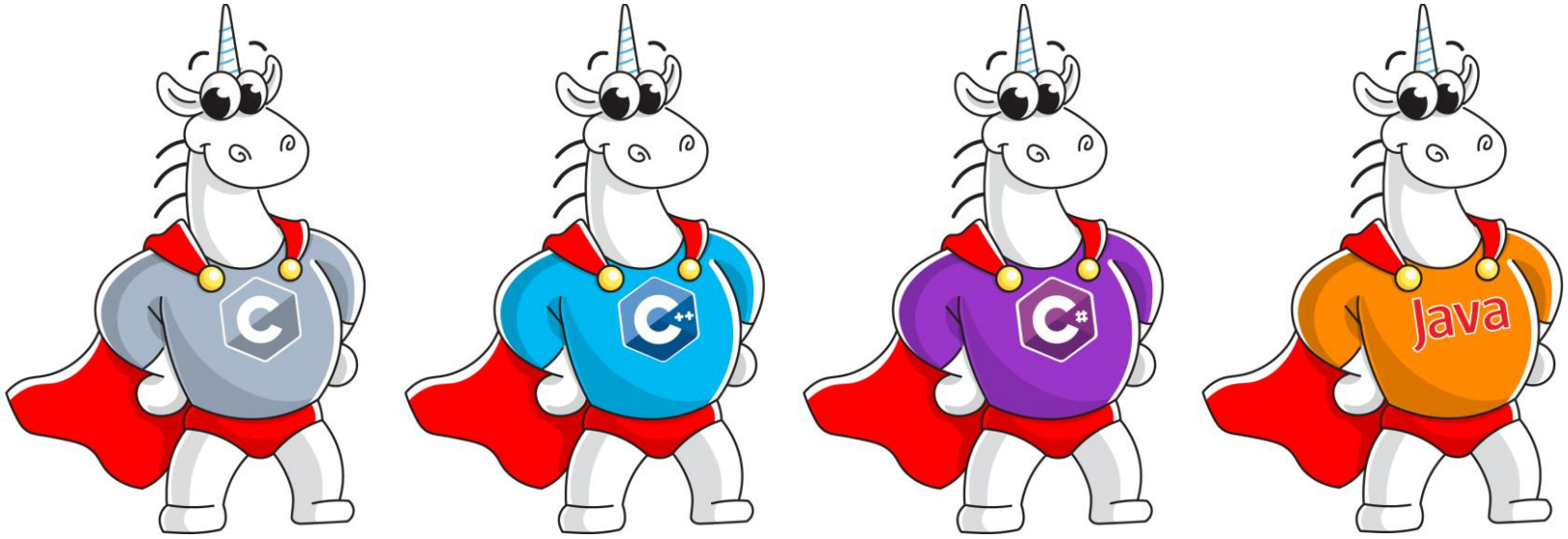
- Windows. Visual Studio 2010-2017. **C, C++, C++/CLI, C++/CX, C#**
- Windows/Linux. Keil µVision, DS-MDK. ARM Compiler 5/6: **C, C++**
- Windows. IAR Embedded Workbench. C/C++ Compiler for ARM: **C, C++**
- Windows/Linux. Texas Instruments Code Composer Studio, ARM Code Generation Tools - Compiler: **C, C++**
- Windows/Linux/macOS. Clang. **C, C++, Java**
- Linux/macOS. GCC. **C, C++**
- Windows. MinGW. **C, C++**

PVS-Studio static code analyzer

- Plugin for **Visual Studio 2010-2019**
- Integration with SonarQube, QtCreator, CLion, Eclipse CDT, Anjuta DevStudio and so on
- Cloud CI platforms integration (for example, Travis CI)
- C and C++ Compiler Monitoring UI utility for IDE-independent analysis and working with analysis reports
- HTML report with built-in analyzed sources

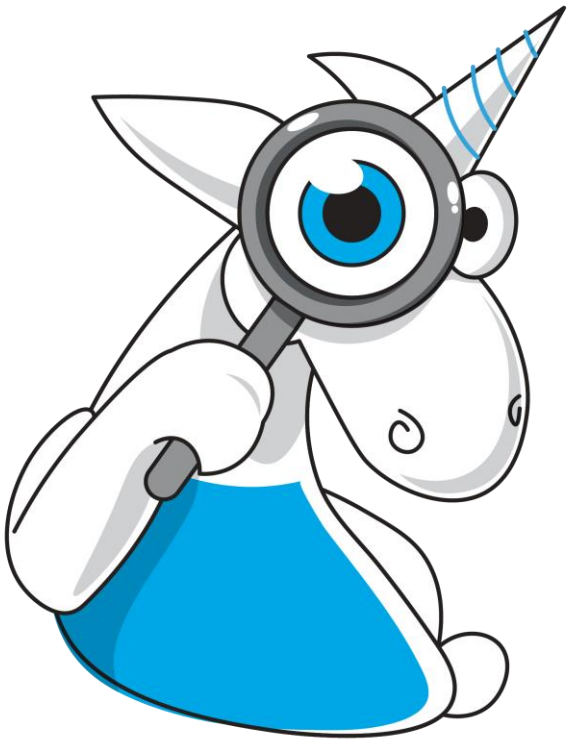
By August 2019 we've implemented in PVS-Studio:

- **C, C++** diagnostics: **465**
- **C#** diagnostics : **141**
- **Java** diagnostics : **69**



Great attention is paid to analyzer warnings:

- Warnings classification is supported according to:
  - Common Weakness Enumeration (CWE)
  - SEI CERT C Coding Standard
  - SEI CERT C++ Coding Standard
  - MISRA C, MISRA C++
- Detailed documentation in Russian and English:
  - Online
  - PDF



Main features

- **Quick start** (compiler monitoring)
  - Windows utility: CLMonitoring
  - Linux/macOS utility : pvs-studio-analyzer
- Ability to view warnings only on newly written code (suppress files)
- Direct integration of the analyzer into build automation systems and the BlameNotifier utility (distribution of warnings by mail)
- Automatic analysis of changed files
- Perfect scalability
- Dealing with false positives

Cloud CI-systems integration

- Integration into third-party cloud platforms (i.e. CircleCI, Travis CI, GitLab) is available.
- You can read more about this in our article by the link:

«PVS-Studio in the Clouds - Running the Analysis on Travis CI»

Тема **PVS-Studio gcc report, commit:560eed87f5c55b1169acc5b85b09feb528f8d8e2** 19.06.2019, 16:40

Кому Мне

PVS-Studio-gcc.html

MESSAGES

Location	Code	Message
		Fails/Info
(1)	Renew	Your license will expire in 7 days. Click 'Renew' to learn more or contact us at support@viva64.com.
		General Analysis (GA)
af_unix.c (68)	V590	Consider inspecting this expression. The expression is excessive or contains a misprint.
af_unix.c (183)	V519	The 'msg.msg_controllen' variable is assigned values twice successively. Perhaps this is a mistake. Check lines: 176, 183.
af_unix.c (236)	V560	A part of conditional expression is always true: cmsg.
attach.c (107)	V590	Consider inspecting the 'ret != (- 1) && ret == 1' expression. The expression is excessive or contains a misprint.
attach.c (579)	V590	Consider inspecting the 'ret != (- 1) && ret == 1' expression. The expression is excessive or contains a misprint.
attach.c (583)	V590	Consider inspecting the 'ret != (- 1) && ret == 1' expression. The expression is excessive or contains a misprint.
caps.c (250)	V560	A part of conditional expression is always true: uid

2 вложения 82,9 КБ Сохранить все

PVS-Studio-gcc.log 57,8 КБ PVS-Studio-gcc.html 25,0 КБ

Diagnostic capabilities of PVS-Studio

Condition is always true

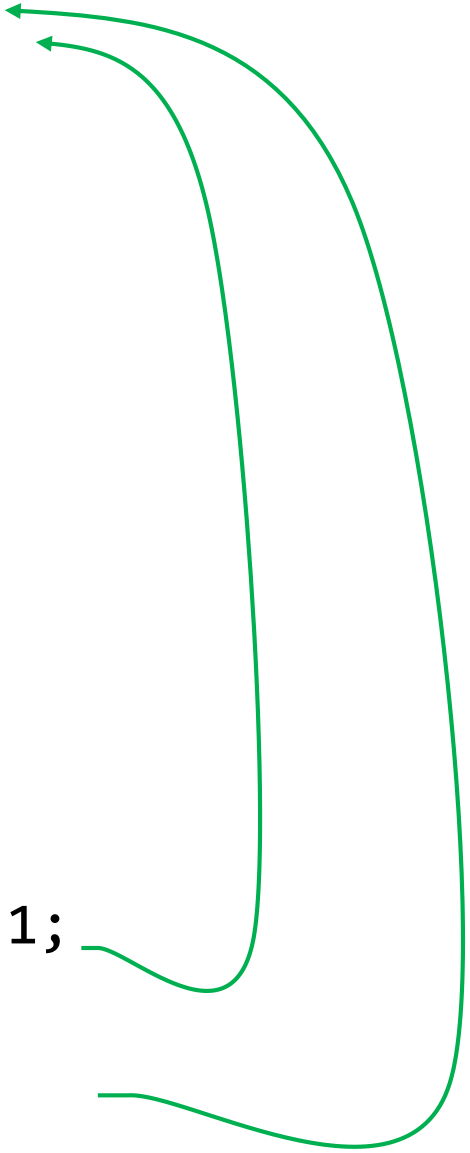
- This error demonstrates greatly how **Data Flow analysis** works in PVS-Studio
- This error was found using PVS-Studio in Chromium project (Protocol Buffers)

- The analyzer issues two warnings:
  - V547 Expression 'time.month <= kDaysInMonth[time.month] + 1' is always true. time.cc 83
  - V547 Expression 'time.month <= kDaysInMonth[time.month]' is always true. time.cc 85

```
static const int kDaysInMonth[13] = {
  0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
};

bool ValidateDateTime(const DateTime& time) {
  if (time.year < 1   || time.year > 9999 ||
      time.month < 1  || time.month > 12  ||
      time.day < 1    || time.day > 31    ||
      time.hour < 0   || time.hour > 23   ||
      time.minute < 0 || time.minute > 59 ||
      time.second < 0 || time.second > 59) {
    return false;
  }
  if (time.month == 2 && IsLeapYear(time.year)) {
    return time.month <= kDaysInMonth[time.month] + 1;
  } else {
    return time.month <= kDaysInMonth[time.month];
  }
}
```

```
static const int kDaysInMonth[13] = {
  0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
};

bool ValidateDateTime(const DateTime& time) {
  if (time.year < 1   || time.year > 9999 ||
      time.month < 1  || time.month > 12  ||
      time.day < 1    || time.day > 31    ||
      time.hour < 0   || time.hour > 23   ||
      time.minute < 0 || time.minute > 59 ||
      time.second < 0 || time.second > 59) {
    return false;
  }
  if (time.month == 2 && IsLeapYear(time.year)) {
    return time.month <= kDaysInMonth[time.month] + 1;
  } else {
    return time.month <= kDaysInMonth[time.month];
  }
}
```

```
static const int kDaysInMonth[13] = {
  0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
};

bool ValidateDateTime(const DateTime& time) {
  if (time.year < 1   || time.year > 9999 ||
      time.month < 1  || time.month > 12  ||
      time.day < 1    || time.day > 31    ||
      time.hour < 0   || time.hour > 23   ||
      time.minute < 0 || time.minute > 59 ||
      time.second < 0 || time.second > 59) {
    return false;
  }
  if (time.month == 2 && IsLeapYear(time.year)) {
    return time.month <= kDaysInMonth[time.month] + 1;
  } else {
    return time.month <= kDaysInMonth[time.month];
  }
}
```



```
static const int kDaysInMonth[13] = {
  0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
};

bool ValidateDateTime(const DateTime& time) {
  if (time.year < 1   || time.year > 9999 ||
      time.month < 1  || time.month > 12  ||
      time.day < 1    || time.day > 31    ||
      time.hour < 0   || time.hour > 23   ||
      time.minute < 0 || time.minute > 59 ||
      time.second < 0 || time.second > 59) {
    return false;
  }
  if (time.month == 2 && IsLeapYear(time.year)) {
    return time.month <= kDaysInMonth[time.month] + 1;
  } else {
    return time.month <= kDaysInMonth[time.month];
  }
}
```

time.day

Address of a local variable is returned from a function by reference

This error was found using PVS-Studio in **LLVM** project

```
SingleLinkedListIterator<T> &operator++(int) {
  SingleLinkedListIterator res = *this;
  ++*this;
  return res;
}
```

V558 Function returns the reference to temporary local object: res. LiveInterval.h 679

Arithmetic overflow and underflow

This error was found using PVS-Studio in **OpenXRay** project

```
float CRenderTarget::im_noise_time;
....
param_noise_fps       = 25.f;
param_noise_scale     = 1.f;
im_noise_time         = 1/100;
....
```

V636 The '1 / 100' expression was implicitly cast from 'int' type to 'float' type. Consider utilizing an explicit type cast to avoid the loss of a fractional part. An example: double A = (double)(X) / Y;. gl_rendertarget.cpp 245

Array overrun

This error was found using PVS-Studio in **Notepad++** project

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

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

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

Dead code

This error was found using PVS-Studio in **Unreal Engine 4** project

```
int32 NumByteProperties = 0;
....
if (bIsByteProperty)
{
  NumByteProperties;
}
```

V607 Ownerless expression 'NumByteProperties'. codegenerator.cpp 633

Unreachable code

This error was found using PVS-Studio in **Linux Kernel** project

```
if (val > 511)
  val = (val >> 1) | (1 << 9);
else if (val > 1022)
  val = (val >> 2) | (3 << 9);
```

V695 Range intersections are possible within conditional expressions.
Example: if (A < 5) { ... } else if (A < 2) { ... }. Check lines: 439, 441. ad5933.c 441

Uninitialized variables

This error was found using PVS-Studio in **Mono** project

```
class ResXResourceWriter : IResourceWriter, IDisposable
{
  public static readonly string ResourceSchema = schema;
  ....
  static string schema = ....;
}
```

V3070 Uninitialized variable 'schema' is used when initializing the 'ResourceSchema' variable. ResXResourceWriter.cs 59

By the time of *ResourceSchema* initialization, the *schema* field will be initialized by default value (*null* in this case).

Unused variables and arguments

This error was found using PVS-Studio in **Xenko** project

```
public static Image New3D(int width, int height, int depth, ....)
{
    return new Image(
                CreateDescription(
                    TextureDimension.Texture3D,
                    width, width, depth,
                    mipMapCount, format, 1),
                dataPointer, 0, null, false);
}
```

V3065 Parameter 'height' is not utilized inside method's body. SiliconStudio.Xenko Image.cs 473

Incorrect shift operations

This error was found using PVS-Studio in **Bitcoin** project

```
static int64_t set_vch(....) {
  int64_t result = 0;
  ....
  return -(result & ~(0x80 << (8 * (vch.size() - 1))));
```

V629 Consider inspecting the '0x80 << (8 * (vch.size() - 1))' expression. Bit shifting of the 32-bit value with a subsequent expansion to the 64-bit type. script.h 169

Overflow occurs when shifting the 32-bit value of 0x80.
The correct code looks as follows:

```
return -((int64_t)(result & ~(0x80ULL << (8 * (vch.size() - 1)))));
```

Incorrect handling of types

This error was found using PVS-Studio in **VirtualBox** project

```
HRESULT EventClassID(BSTR bstrEventClassID);


static HRESULT VBoxCredentialProviderRegisterSENS(void)
{
  hr = pIEventSubscription->put_EventClassID(
          L"{d5978630-5b9f-11d1-8dd2-00aa004abd5e}");
```

V745 A 'wchar_t *' type string is incorrectly converted to 'BSTR' type string. Consider using 'SysAllocString' function. vboxcredentialprovider.cpp 231

Incorrect understanding about how a function/class operates

This error was found using PVS-Studio in **Unity3D** project

```
private static readonly Regex UnsafeCharsWindows =
  new Regex("[^A-Za-z0-9\\_\\-\\.\\:\\,\\/\\@\\\\]");
```

V3057 Invalid regular expression patern in constructor. Inspect the first argument. AssetBundleDemo ExecuteInternalMono.cs 48

When attempting to create an instance of *Regex* class with this pattern, we'll get the **System.ArgumentException** exception with the message:

parsing \"[^A-Za-z0-9_\\-\\.\\:\\,\\/\\@\\]\" -
Unrecognized escape sequence '_'.

Absence of virtual destructor

All examples are long, so it’s difficult to insert them in a presentation. The main point is that the analyzer detects such problems.

Code formatting doesn't correspond with its operational logic

This error was found using PVS-Studio in **Sony ATF** project

```
public static QuatF Slerp(QuatF q1, QuatF q2, float t)
{
  double dot = q2.X * q1.X + q2.Y * q1.Y +
               q2.Z * q1.Z + q2.W * q1.W;
  if (dot < 0)
    q1.X = -q1.X; q1.Y = -q1.Y; q1.Z = -q1.Z; q1.W = -q1.W;
```

V3043 The code's operational logic does not correspond with its formatting. The statement is indented to the right, but it is always executed. It is possible that curly brackets are missing. Atf.Core.vs2010 QuatF.cs 282

Error when working with exceptions

This error was found using PVS-Studio in **OpenMW** project

```
if (t1==t2)
  mOperands.push_back (t1);
else if (t1=='f' || t2=='f')
  mOperands.push_back ('f');
else
  std::logic_error ("failed to ....... ");
```

~~throw~~

V596 The object was created but it is not being used. The 'throw' keyword could be missing: throw logic_error(FOO); components exprparser.cpp 101

Buffer overflow

This error was found using PVS-Studio in **FreeBSD** project

```
#define   SID_VENDOR_SIZE    8
char    vendor[SID_VENDOR_SIZE];
....
strcpy(p->vendor,"Adaptec ");
```

V512 A call of the 'strcpy' function will lead to overflow of the buffer 'p->vendor'. aacraid_cam.c 571

A string contains 8 characters. However, it has to be taken into account that the function *strcpy* will add a terminal null to the string. It will be written out of bounds of the buffer.

Security issues

This error was found using PVS-Studio in **PostgreSQL** project

```
char *px_crypt_md5(....) {
  unsigned char final[MD5_SIZE];
  ....
  /* Don't leave anything around in vm they could use. */
  memset(final, 0, sizeof final);
}
```

Compiler deletes a call of *memset* function: http://www.viva64.com/en/w/V597/

V597 The compiler could delete the 'memset' function call, which is used to flush 'final' buffer. The RtlSecureZeroMemory() function should be used to erase the private data. pgcrypto crypt-md5.c 157

Confusion with priority of operations

This error was found using PVS-Studio in **Linux Kernel** project

```
static int nvme_pr_preempt(struct block_device *bdev,
  u64 old, u64 new, pr_type type, bool abort)
{
  u32 cdw10 = nvme_pr_type(type) << 8 | abort ? 2 : 1;
```

1

2

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

Dereference of a null pointer / null reference

This error was found using PVS-Studio in **LibreOffice** project

```
MenuBar *pMBar = pSysWin->GetMenuBar();


if ( pSysWin && pMBar )
{
  AddMenuBarIcon( pSysWin, true );
}
```

V595 The 'pSysWin' pointer was utilized before it was verified against nullptr. Check lines: 738, 739. updatecheckui.cxx 738

Synchronization errors

This error was found using PVS-Studio in **Unity3D** project

```
internal void OnUnload()
{
  m_AssetBundle.Unload(false);
  if (unload != null)
    unload();
}
```

V3083 Unsafe invocation of event 'unload', NullReferenceException is possible. Consider assigning event to a local variable before invoking it. AssetBundleDemo AssetBundleManager.cs 47

Integer division by 0

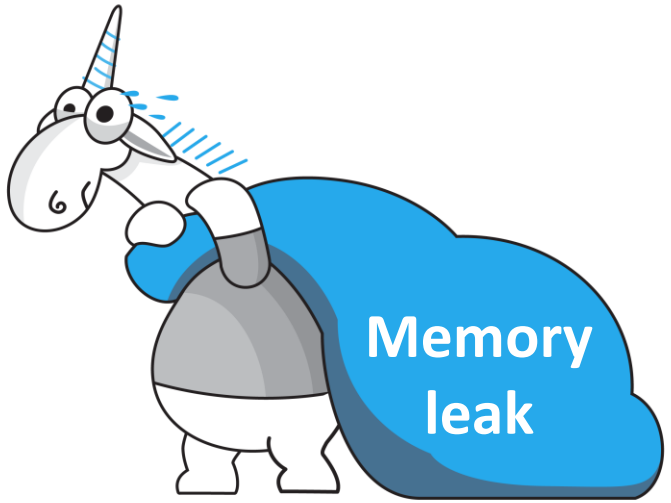
This error was found using PVS-Studio in **Inkscape** project

```
} else if (type >= 3000 && type < 4000) {
    unsigned int chamferSubs = type-3000;
    double chamfer_stepsTime = 1.0/chamferSubs;
```

V609 Divide by zero. Denominator range [0..999]. lpe-fillet-chamfer.cpp 607

Memory leaks

- Frequent question: does the PVS-Studio analyzer detect memory leaks?
- Short answer: yes
- A fuller answer: yes, but, as any other static analyzer, PVS-Studio can do it with less precision than a dynamic code analyzer
- More details about searching for memory leaks using PVS-Studio: https://www.viva64.com/en/b/0543/
- Let's see the example



Memory leaks

This error was found using PVS-Studio in **PDFium** project

```
std::unique_ptr<CCodec_JpegModule::Context>
CCodec_JpegModule::Start()
{
  auto* pContext = new CJpegContext();
  if (setjmp(pContext->m_JumpMark) == -1)
    return nullptr;
  ....
}
```

V773 The function was exited without releasing the 'pContext' pointer. A memory leak is possible. fx_codec_jpeg.cpp 421

Errors, occurring when porting code to 64-bit platforms

This error was found using PVS-Studio in **TortoiseSVN** project

```
DialogBoxParam(g_hmodThisDll,
               MAKEINTRESOURCE(IDD_LOGIN),
               g_hwndMain,
               (DLGPROC)(LoginDialogProc),
               (long)this);
```

V220 Suspicious sequence of types castings: memsize -> 32-bit integer -> memsize. The value being casted: 'this'. logindialog.cpp 105

Type ***long*** in Win64 is still a 32-bit one. In a 64-bit program, an object can be created outside the bounds of first 4 Gigabytes of memory addresses. In this case, the value of the pointer will be corrupted. Unpleasant error, which can reveal itself very seldom after a program works fine for a long time.

Correct: (LPARAM)(this).

PVS-Studio and Java-projects

Integer division

This error was found using PVS-Studio in **IntelliJ IDEA** project

```
private static boolean checkSentenceCapitalization(@NotNull String value)
{
  List<String> words = StringUtil.split(value, " ");
  ....
  int capitalized = 1;
  ....
  return capitalized/words.size() < 0.2; // allow reasonable amount of
                                         // capitalized words
}
```

V6011 [CWE-682] The '0.2' literal of the 'double' type is compared to a value of the 'int' type. TitleCapitalizationInspection.java 169

NullPointerException

This error was found using PVS-Studio in **Elasticsearch** project

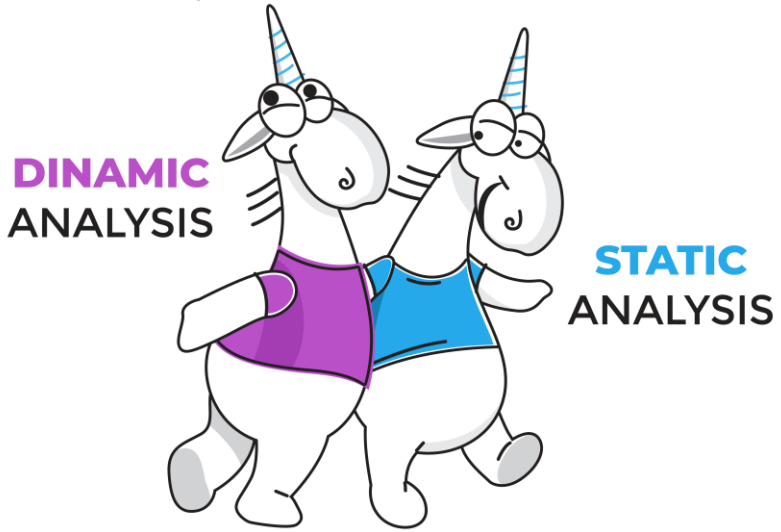
```
private static PathTrie<RequestHandler> defaultHandlers(....) {
  ....
  handlers.insert("POST /batch/storage/v1", (request) -> {
    ....
    line = reader.readLine();
    byte[] batchedBody = new byte[0];
    if ((line != null) ||
        (line.startsWith("--" + boundary) == false))
    {
      ....
    }
    ....
  });
  ....
}
```

V6008 Null dereference of 'line'. GoogleCloudStorageFixture.java(451)

Is dynamic analysis better than static?

- No
- Each type of analysis has its strengths and weaknesses
- These approaches don’t compete with each other
- Static and dynamic analyses supplement each other

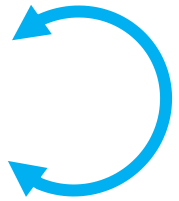
- What's the Use of Dynamic Analysis When You Have Static Analysis?
  https://www.viva64.com/en/b/0643/



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

This error was found using PVS-Studio in **Valgrind** project

```
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 );
}
```



V779 Unreachable code detected. It is possible that an error is present. host_arm_isel.c 461

Typos and Copy-Paste

- PVS-Studio analyzer effectively detects typos and consequences of erroneous Copy-Paste
- In the analyzer, there are many diagnostics for detecting errors of such a type
- Let’s consider them in more detail and see several examples of errors of this type
- We also recommend these exciting articles for reading:
  - The Last Line Effect- http://www.viva64.com/en/b/0260/
  - The Evil within the Comparison Functions- https://www.viva64.com/en/b/0509/

Typos and Copy-Paste (example N1)

This error was found using PVS-Studio in **Clang** project

```
if ((OpcodeLHS == BO_EQ ||
     OpcodeLHS == BO_LE ||
     OpcodeLHS == BO_LE)
    &&
    (OpcodeRHS == BO_EQ ||
     OpcodeRHS == BO_GT ||
     OpcodeRHS == BO_GE))
```

V501 There are identical sub-expressions 'OpcodeLHS == BO_LE' to the left and to the right of the '||' operator. RedundantExpressionCheck.cpp 174

Typos and Copy-Paste (example N2)

This error was found using PVS-Studio in **GCC** project

```
return (!strcmp (a->v.val_vms_delta.lbl1,
                 b->v.val_vms_delta.lbl1)
        && !strcmp (a->v.val_vms_delta.lbl1,
                    b->v.val_vms_delta.lbl1));
```

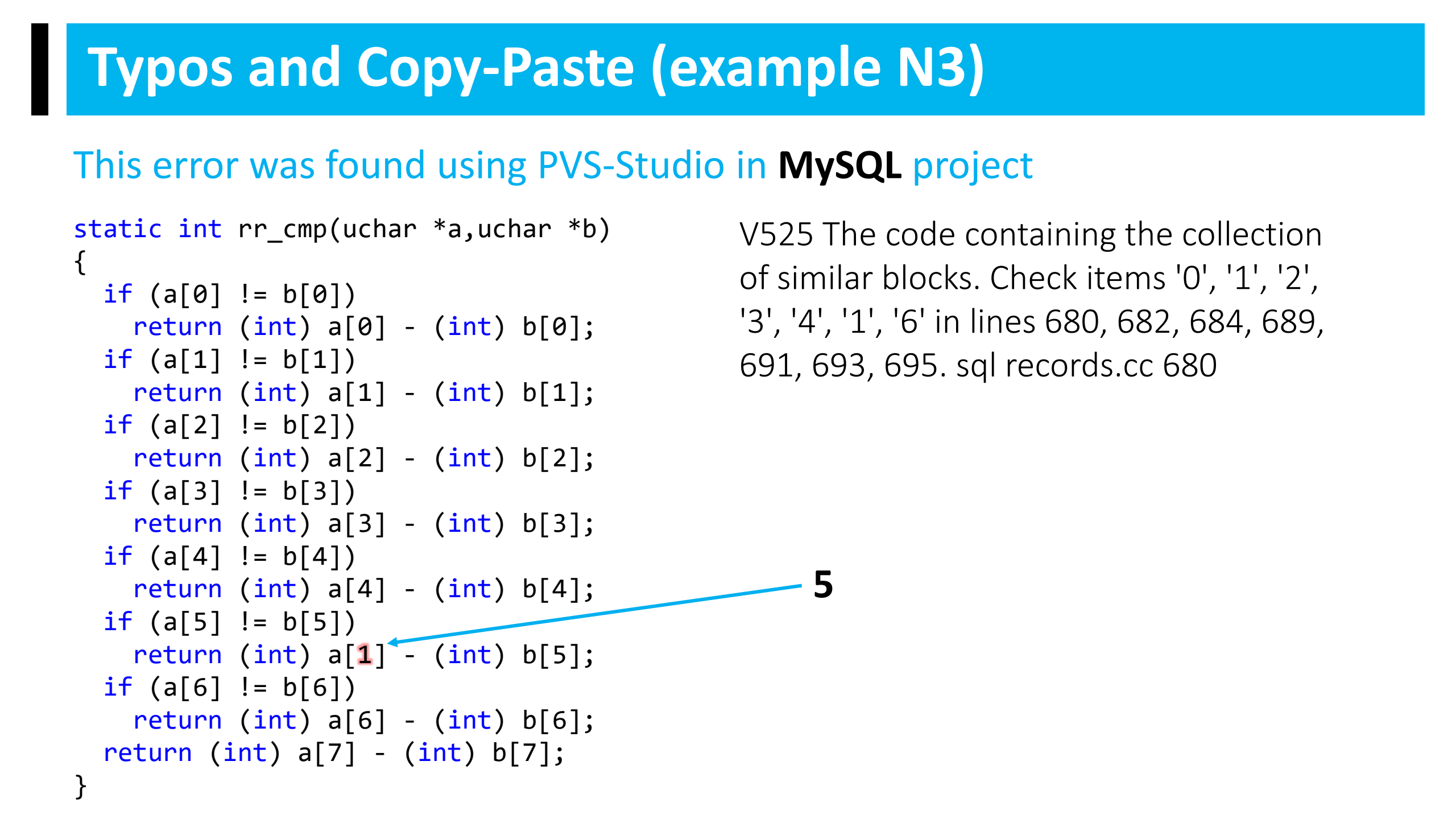
V501 There are identical sub-expressions '!strcmp(a->v.val_vms_delta.lbl1, b->v.val_vms_delta.lbl1)' to the left and to the right of the '&&' operator. dwarf2out.c 1428

Typos and Copy-Paste (example N3)

This error was found using PVS-Studio in **MySQL** project

```
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];
}
```

V525 The code containing the collection of similar blocks. Check items '0', '1', '2', '3', '4', '1', '6' in lines 680, 682, 684, 689, 691, 693, 695. sql records.cc 680



Typos and Copy-Paste (example N4)

This error was found using PVS-Studio in **PowerShell** project

```
internal Version BaseMinimumVersion { get; set; }
internal Version BaseMaximumVersion { get; set; }

protected override void ProcessRecord()
{
  if (BaseMaximumVersion != null &&
      BaseMaximumVersion != null &&
      BaseMaximumVersion < BaseMinimumVersion)
```

V3001 There are identical sub-expressions 'BaseMaximumVersion != null' to the left and to the right of the '&&' operator. System.Management.Automation ImportModuleCommand.cs 1663

We've demonstrated only a small part of what PVS-Studio analyzer can find

- Detailed table of diagnostic capabilities: http://www.viva64.com/en/w/
- You will also find a detailed description of all diagnostics

Main PVS-Studio diagnostic abilities	C, C++ diagnostics	C# diagnostics
64-bit issues	V101-V128, V201-V207, V220, V221, V301-V303	-
Check that addresses to stack memory does not leave the function	V506, V507, V558, V758	-
Arithmetic over/underflow	V636, V658	V3040, V3041
Array index out of bounds	V557, V582, V643	V3106
Check for double-free	V586, V749	-
Dead code	V606, V607	-
Microoptimization	V801-V815	-
Unreachable code	V551, V695, V734	-
Uninitialized variables	V573, V614, V679, V730, V737	V3070
Unused variables	V603, V751, V763	V3061, V3065, V3077
Illegal bitwise/shift operations	V610, V629, V673, V684	-
Undefined/unspecified behavior	V567, V610, V611, V681, V704, V708, V726, V736	-
Incorrect handling of the types (HRESULT, BSTR, BOOL, VARIANT_BOOL)	V543, V544, V545, V716, V721, V724, V745, V750, V676, V767	-
Improper understanding of function/class operation logic	V518, V530, V540, V541, V554, V575, V597, V598, V618, V630, V632, V663, V668, V698, V701, V702, V717, V718, V720, V723, V725, V727, V738, V742, V743, V748, V762, V764	V3010, V3057, V3068, V3072, V3073, V3074, V3082, V3084, V3094, V3096, V3097, V3102, V3103, V3104, V3108
Misprints	V501, V503, V504, V508, V511, V516, V519, V520, V521, V525, V527, V528, V529, V532, V533, V534, V535, V536, V537, V539, V546, V549, V552, V556, V559, V560, V561, V564, V568, V570, V571, V575, V577, V578, V584, V587, V588, V589, V590, V592, V600, V602, V604, V606, V607, V616, V617, V620, V621, V622, V625, V626, V627, V633, V637, V638, V639, V644, V646, V650, V651, V653, V654, V655, V660, V661, V662, V666, V669, V671, V672, V678, V682, V683, V693, V715, V722, V735, V747, V754, V756, V765, V767	V3001, V3003, V3005, V3007, V3008, V3009, V3011, V3012, V3014, V3015, V3016, V3020, V3028, V3029, V3034, V3035, V3036, V3037, V3038, V3050, V3055, V3056, V3057, V3062, V3063, V3066, V3081, V3086, V3091, V3092, V3107, V3109
Missing Virtual destructor	V599, V689	-
Coding style not matching the operation logic of the source code	V563, V612, V628, V640, V646, V705	V3018, V3033, V3043, V3067, V3069
Copy-Paste	V501, V517, V519, V523, V524, V571, V581, V649, V656, V691, V760, V766	V3001, V3003, V3004, V3008, V3012, V3013, V3021, V3030, V3058
Incorrect usage of exceptions	V509, V565, V596, V667, V740, V741, V746, V759	V3006, V3052, V3100
Buffer overrun	V512, V514, V594, V635, V641, V645, V752, V755	-
Security issues	V505, V510, V511, V512, V518, V531, V541, V547, V559, V560, V569, V570, V575, V576, V579, V583, V597, V598, V618, V623, V642, V645, V675, V676, V724, V727, V729, V733, V743, V745, V750	V3022, V3023, V3025, V3027, V3053, V3063
Operation priority	V502, V562, V593, V634, V648	-
Null pointer pointer/null reference dereference	V522, V595, V664, V757	V3019, V3042, V3080, V3095, V3105
Unchecked parameter dereference	V595, V664	V3095
Synchronization errors	V712	V3032, V3054, V3079, V3083, V3089, V3090
WPF usage errors	-	V3044 - V3049
Check for integer division by zero	V609	V3064
Customized user rules	V2001-V2013	-

Demonstration of PVS-Studio capabilities

- To demonstrate the capabilities of the analyzer we check open source projects.
- Indirect result: in these projects our team found **13 124** errors
- By saying **13 124** errors, we're not saying about the number of warnings, issued by the analyzer, but about the number of actual errors

Demonstration of PVS-Studio capabilities

- Thanks to our team and PVS-Studio analyzer, **10 000** errors have been fixed in open source projects
- You can see all these errors by this link: http://www.viva64.com/en/examples/
- Error base is constantly updated, and it can be used when writing articles about code quality and forming coding standards

Correct scenario of using the analyzer

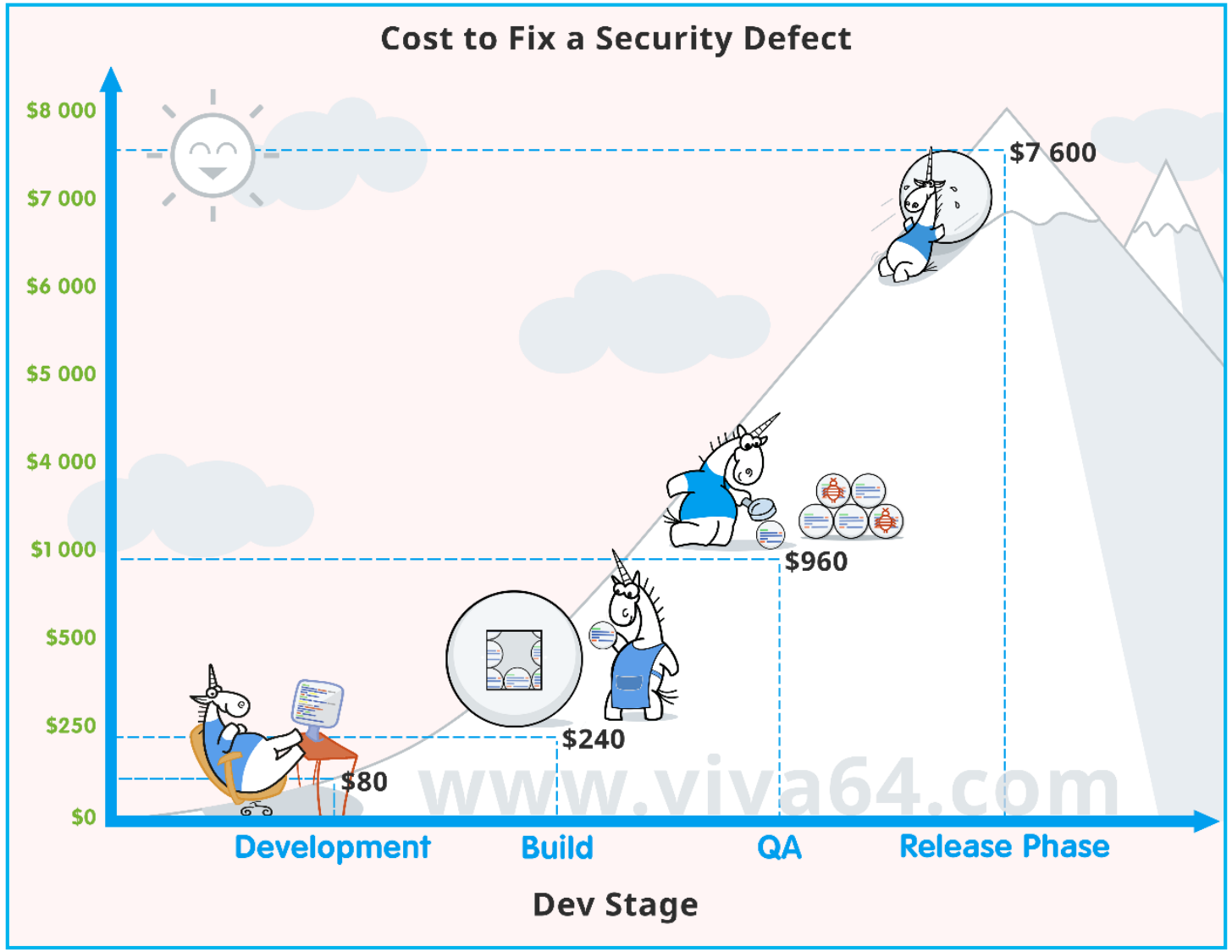
- Sure, it's exciting and useful to run the PVS-Studio analyzer and find a bug that was unsuccessfully searched for 50 hours beforehand http://www.viva64.com/en/b/0221/
- It's good to check projects and describe detected errors, as we usually do for promotion purposes http://www.viva64.com/en/inspections/
- **But one should remember that single check is not the right way of using static code analyzers!**

Correct scenario of using the analyzer

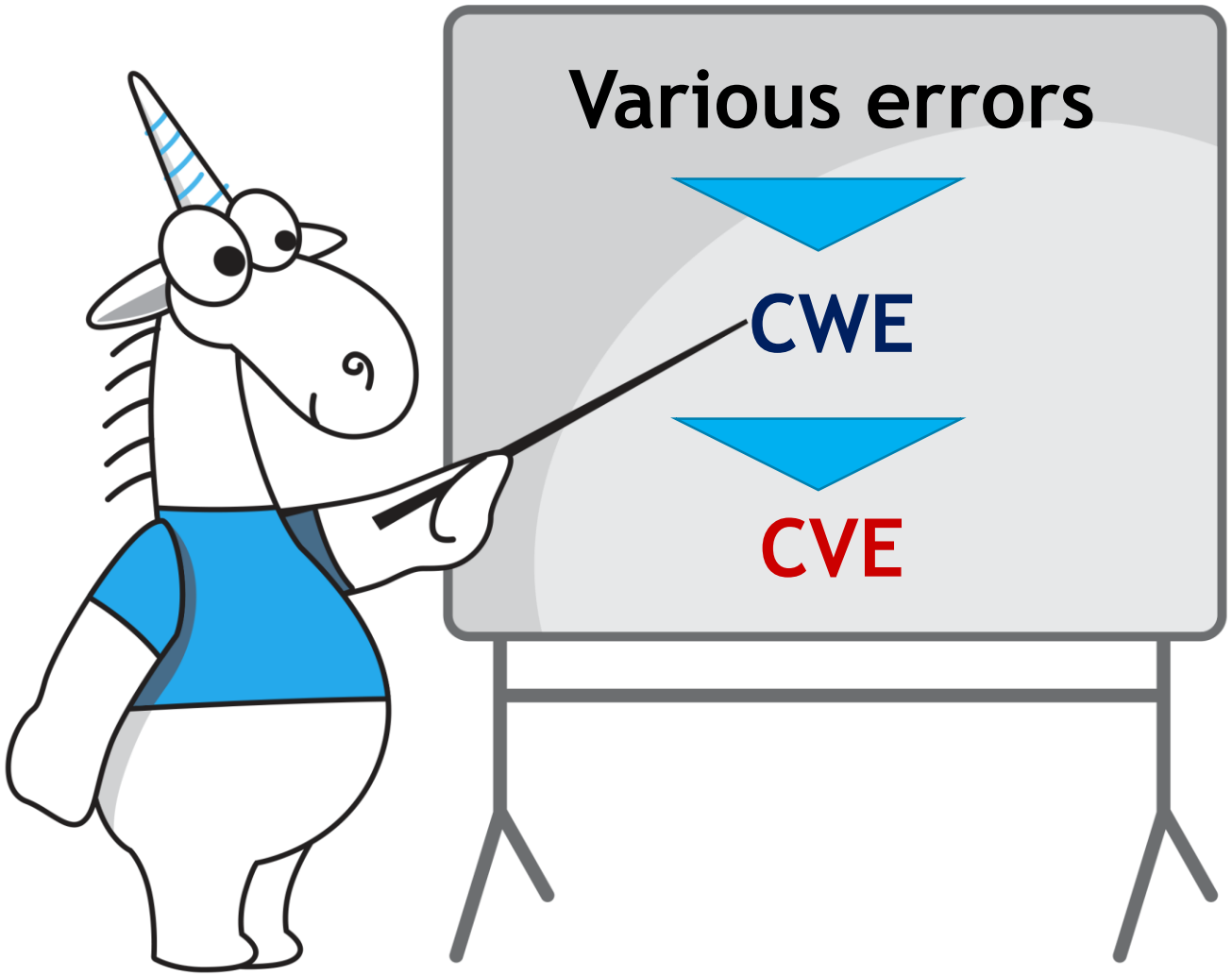
- Static analyzer provides benefit when it is used regularly
- Two ways:
  - Automatic analysis of changed code
  - Nightly checks
- These modes are described in more detail in the documentation https://www.viva64.com/en/m/

PVS-Studio and search for vulnerabilities

Fixing vulnerabilities at late stages is very expensive



PVS-Studio will help to detect many vulnerabilities



For example, this vulnerability could have been found using PVS-Studio

```
static OSStatus
SSLVerifySignedServerKeyExchange(.....)
{
  OSStatus err;
  ....

  if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
  if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;
  if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;
  ....

fail:
  SSLFreeBuffer(&signedHashes);
  SSLFreeBuffer(&hashCtx);
  return err;
}
```

CVE-2014-1266

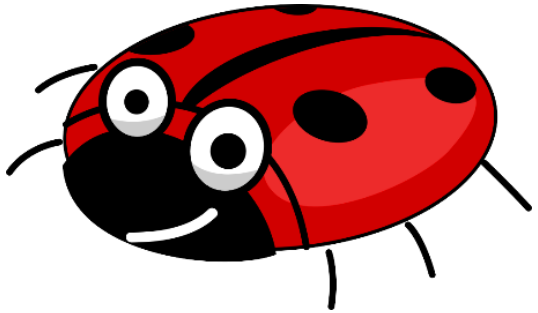
For example, this vulnerability could have been found using PVS-Studio

- PVS-Studio reports about two anomalies:
  - V640 / **CWE-483** The code's operational logic does not correspond with its formatting. The statement is indented to the right, but it is always executed. It is possible that curly brackets are missing.
  - V779 / **CWE-561** Unreachable code detected. It is possible that an error is present.

- Details in the article "**How Can PVS-Studio Help in the Detection of Vulnerabilities?**"
  https://www.viva64.com/en/b/0514/

V1010

- In 2018 we introduced a specialized diagnostic V1010, which detects the usage of unreliable tainted data (data that comes from an external source) without its preliminary check
- A diagnostic will help to detect potential vulnerabilities which can be classified as CWE-20: Improper Input Validation

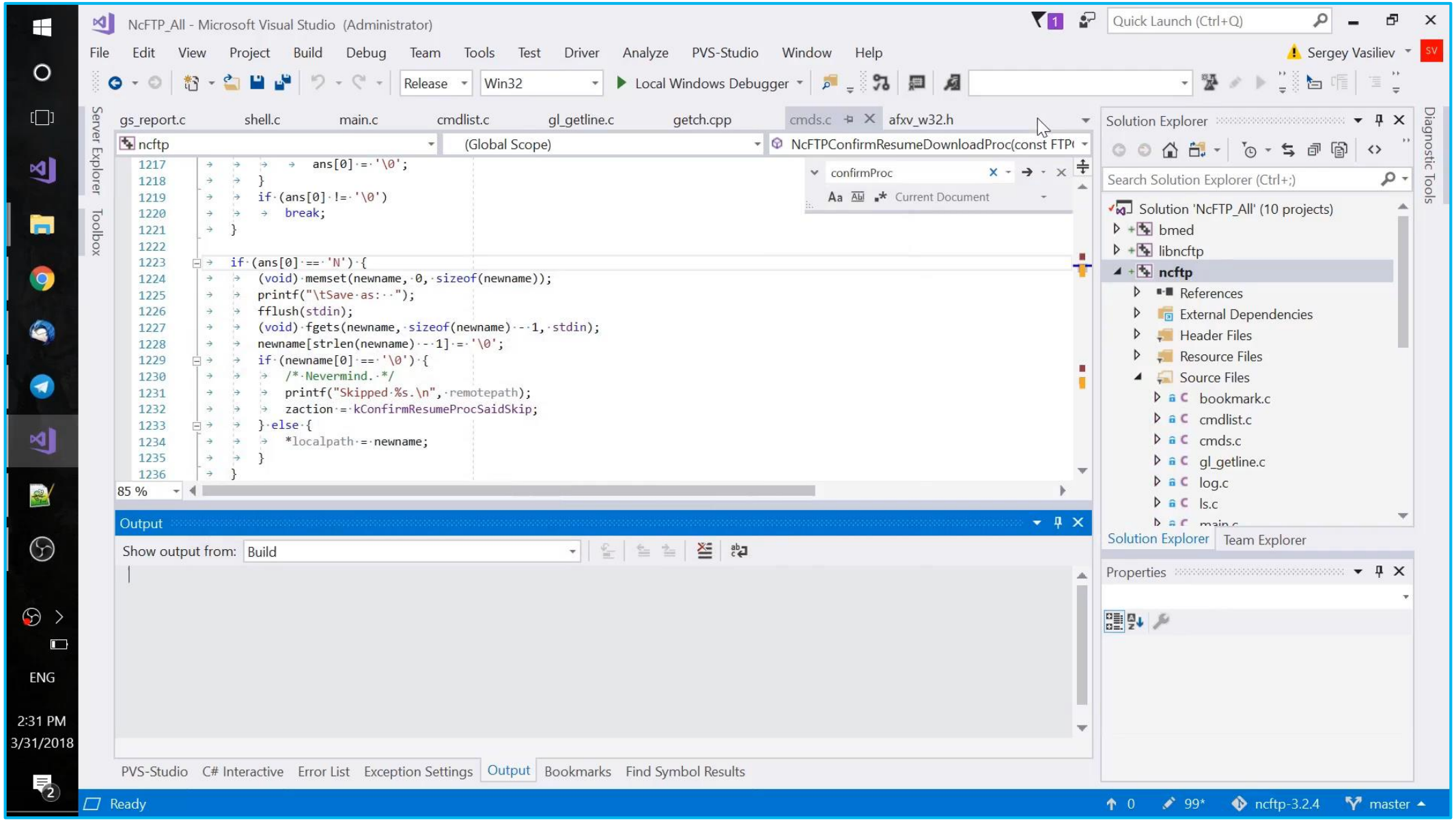


Example of V1010 in NcFTP project

```
static int NcFTPConfirmResumeDownloadProc(....)
{
  ....
  (void) fgets(newname, sizeof(newname) - 1, stdin);
  newname[strlen(newname) - 1] = '\0';
  if (newname[0] == '\0') {
    ....
  } else {
    ....
  }
  ....
}
```

V1010 Unchecked tainted data is used in index: 'strlen(newname)'. ncftp cmds.c 1228

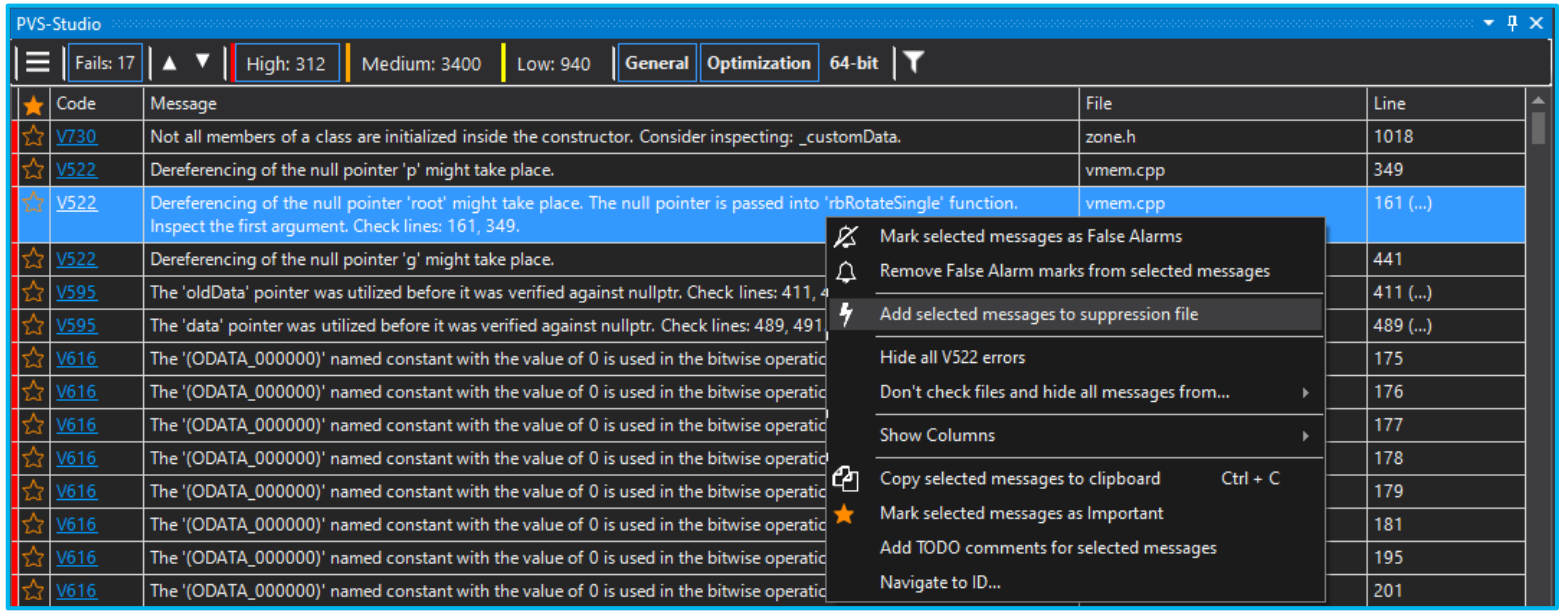
Example of V1010 in NcFTP project



Using PVS-Studio

Using PVS-Studio: introduction

- It can be difficult to start using static analysis in a large project
- It’s not clear what to do with warnings in old code
- We suggest a decision: hiding messages using suppress files
- Read more: http://www.viva64.com/en/b/0364/



Using PVS-Studio: suppressing of false positives

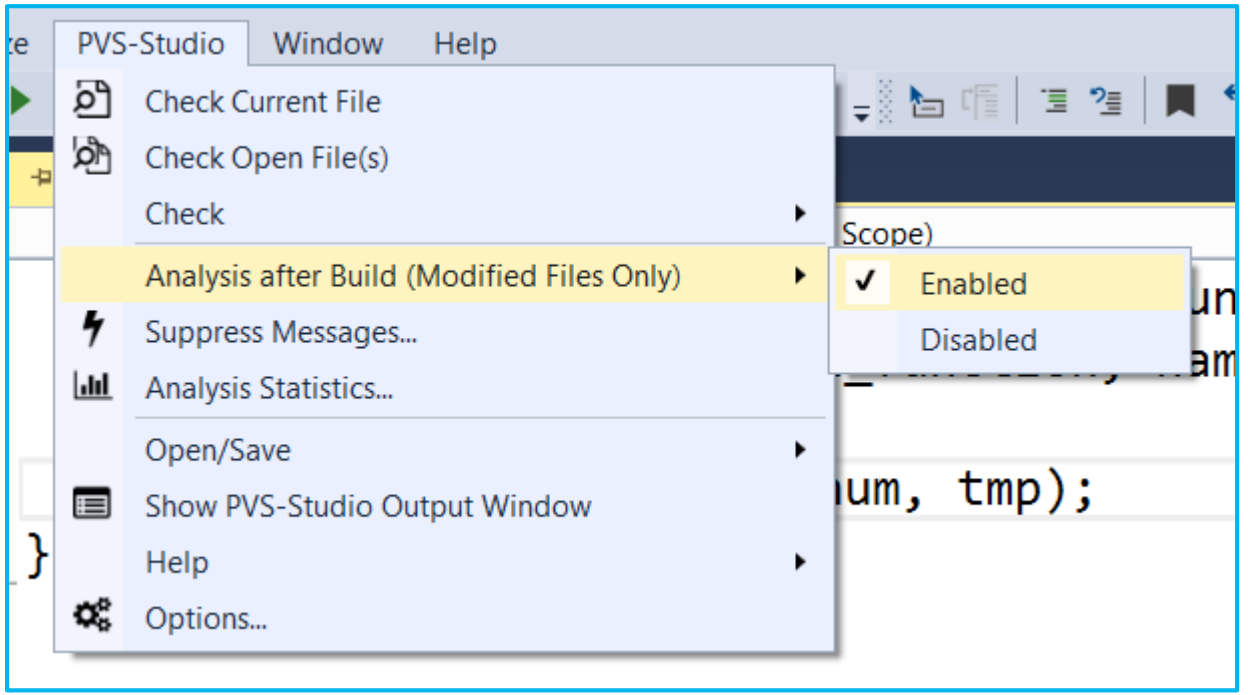
- Various ways to suppress false positives in specific lines of code
- Suppression of false positives in macros
- Suppression of false positives using *pvsconfig* diagnostics configuration files
- Read more: http://www.viva64.com/en/m/0017/

Using PVS-Studio: excluding from analysis

- Possibility to exclude files from analysis by their name, directory or mask
- Interactive filtration of analysis results (log) in PVS-Studio window:
  - by diagnostic code
  - by the file name
  - by including the word in the text of a diagnostic

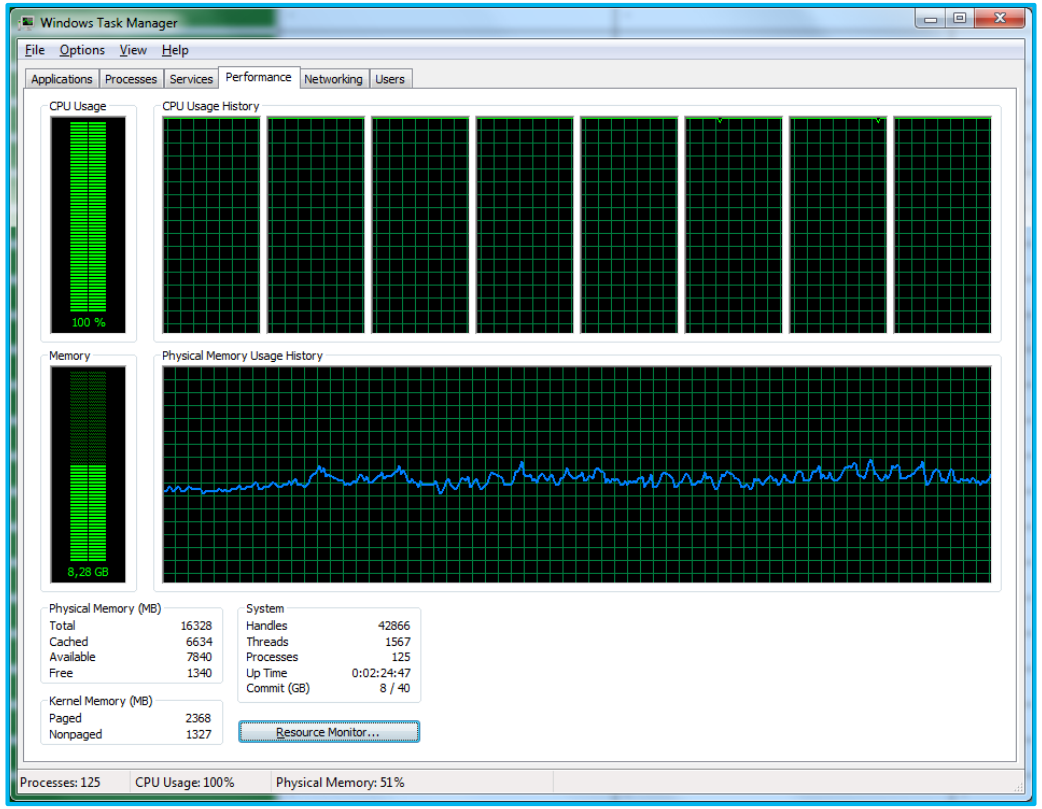
Using PVS-Studio: automatic analysis of files after their recompilation

- The most efficient way of fixing an error is to do it right after it appeared in code



Using PVS-Studio: scalability

- Support of multicore and multiprocessor systems with configuration of the number of utilized cores
- IncrediBuild support



Using PVS-Studio: continuous integration

- Running analysis from command line for checking the whole solution: allows to integrate PVS-Studio into night builds to receive a fresh log in the morning
- Saving and loading of analysis results: you can check the code at night, save the results, and load them for review in the morning
- BlameNotifier utility: a tool that allows to distribute mail notifications to developers about their errors, which PVS-Studio found during the nightly run
- Using of relative paths in report files

Using PVS-Studio: other features

- Convenient online reference on all diagnostics, which is available both from a program and on our web site. Documentation in .pdf as a single file is also available.
- Interactive filtration of analysis (log) results in the PVS-Studio window
- Automatic check on new versions of PVS-Studio

Using PVS-Studio: other features

- PVS-Studio can be easily used under Linux/macOS
- Please, get acquainted with the instruction so that you haven't been confused by the configurations and command line keys
- How to run PVS-Studio in Linux/macOS: http://www.viva64.com/en/m/0036/

- I know that we all don't like reading the instructions. But, believe me, this is the case when everything is simple and short, and it will save your time!

Using PVS-Studio: quick start

- Particular attention should be given to the ability to quickly try PVS-Studio on any project
- For this you can track compiler invocations and gather all needed information for the analysis
- Windows:
  - C and C++ Compiler Monitoring UI tool
  - Instruction: http://www.viva64.com/en/m/0033/
- Linux/macOS
  - pvs-studio-analyzer utility
  - Instruction : see «Quick start» in the document

http://www.viva64.com/en/m/0036/

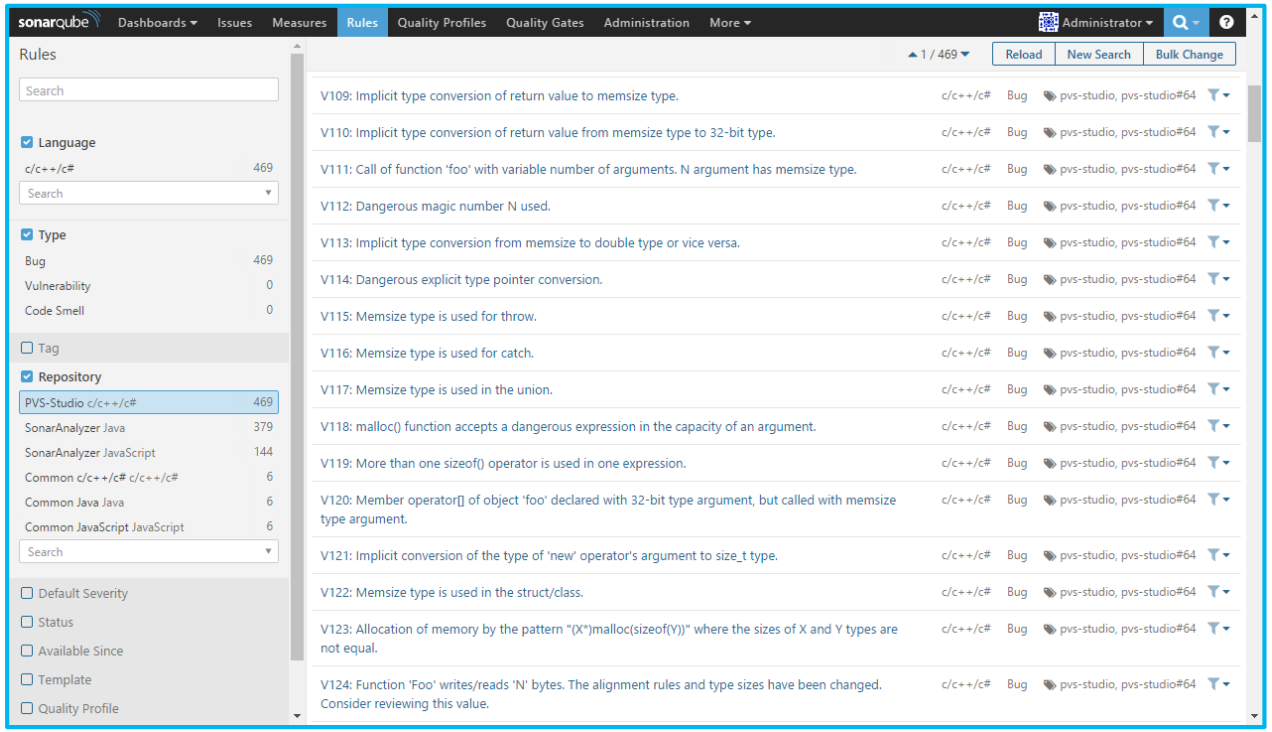
Using PVS-Studio: SonarQube

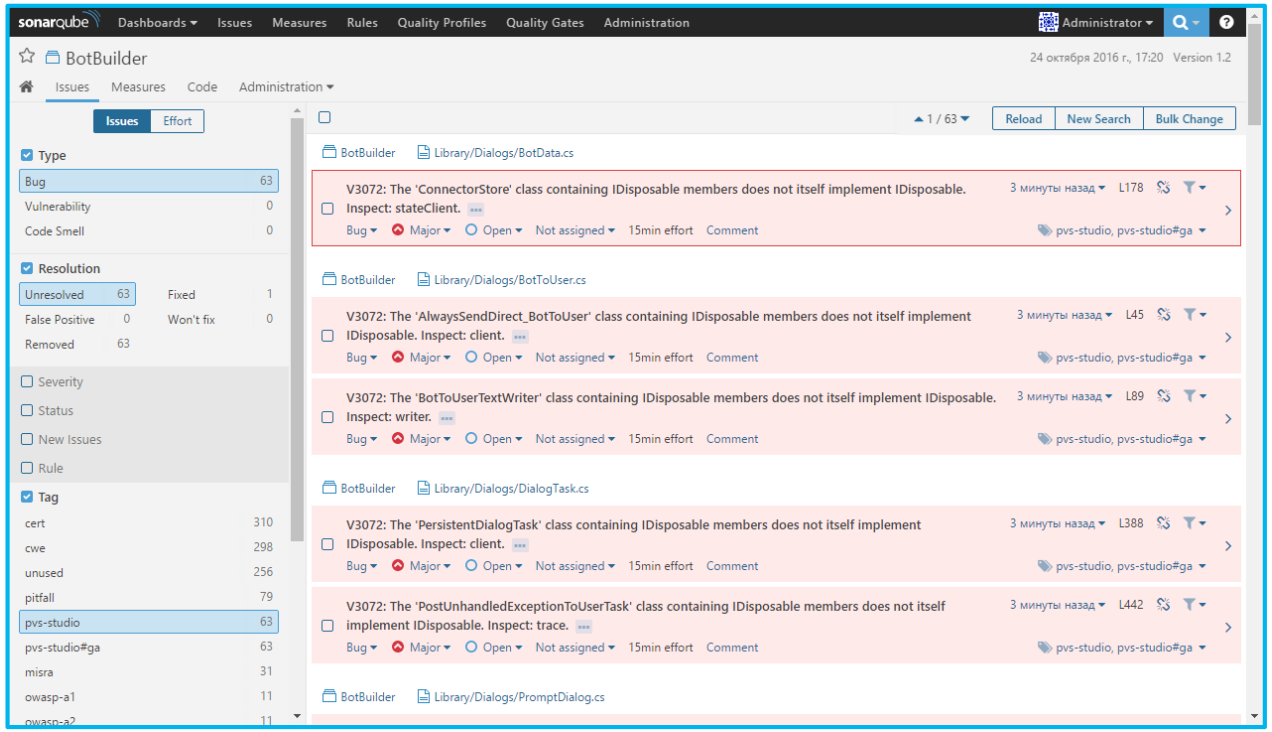
- We developed a plugin for importing analysis results into SonarQube
- Using of this plugin allows to add warnings found by PVS-Studio analyzer to the warnings base of SonarQube server

sonarqube

Using PVS-Studio: SonarQube

- Details are given in the article «Control source code quality using the SonarQube platform»
  http://www.viva64.com/en/b/0452/





Using PVS-Studio: HTML report

PVS-Studio Analysis Results

Date:	Tue Sep 26 17:53:28 2017
PVS-Studio Version:	6.18.23071.1
Command Line:	./plog-converter -a GA\;OP -t html -o /home/svyatoslav/test -r /home/svyatoslav/Projects/ClickHouse/ /home/svyatoslav/Projects/ClickHouse/ClickHouse.log
Total Warnings (GA):	382
Total Warnings (OP):	435

Group	Location	Level	Code	Message
General Analysis	Exception.h:49	Low	V690	The 'ErrnoException' class implements a copy constructor, but lacks the '=' operator. It is dangerous to use such a class.
General Analysis	KeeperException.h:24	Low	V690	The 'KeeperException' class implements a copy constructor, but lacks the '=' operator. It is dangerous to use such a class.
General Analysis	main.cpp:110	Medium	V506	Pointer to local variable 'zookeeper_' is stored outside the scope of this variable. Such a pointer will become invalid.
General Analysis	WriteBufferFromString.h:25	High	V783	Dereferencing of the invalid iterator 's.end()' might take place.
General Analysis	WriteHelpers.h:200	Low	V560	A part of conditional expression is always true: 0x00 <= c. Unsigned type value is always >= 0.
General Analysis	WriteHelpers.h:210	Low	V560	A part of conditional expression is always true: 0 <= lower_half. Unsigned type value is always >= 0.
General Analysis	HashTable.h:220	Medium	V730	Not all members of a class are initialized inside the compiler generated constructor. Consider inspecting: zero_value_storage.
General Analysis	HashTable.h:456	Medium	V730	Not all members of a class are initialized inside the constructor. Consider inspecting: size.
General Analysis	HashTable.h:510	Medium	V730	Not all members of a class are initialized inside the constructor. Consider inspecting: container, ptr.

```
69      struct TryResult
70      {
71          TryResult() = default;
72
73          explicit TryResult(Entry entry_)
74              : entry(std::move(entry))
                ↑ V546 Member of a class is initialized by itself: 'entry(entry)'.
75              , is_usable(true)
76              , is_up_to_date(true)
77          {
78          }
```

Download and try PVS-Studio

Download and try PVS-Studio

- You can download the trial version and request the trial key for the full licence for Windows, Linux or macOS
  http://www.viva64.com/en/pvs-studio-download/
- Read about limitations of the trial version:
  http://www.viva64.com/en/m/0009/

Clients


ARQA TECHNOLOGIES
TRIPWIRE
SAFARI GAMES
ITIVITI
OPTEL
SIG SUSQUEHANNA INTERNATIONAL GROUP, LLP
GlobalView
NEXON GT
GlobalLogic
TAX COM
TOOLS FOR BROKERS enjoy technical perfection
DAVASO
FRAGLAB
Kulicke & Soffa
Crésus
nanometrics
NG Branch Technology
Valassis
Sogitec
THE COALITION
Acolada Content Language Data Management
oculus
MOAI GAMES
TANIUS TECHNOLOGY
mpdv The MES Experts!
rocksteady
softthinks TECHNOLOGIES
AutoChips
AOT Advanced Osteotomy Tools
Ford
nDOORS INTERACTIVE
vmware
medidata
ePLAN
BRUKER
SAIC
DIGITAL EXTREMES
codemasters
TARGEM GAMES
FUJIFILM Value from Innovation
DAQRI
WENZEL
HOTHEAD
exness
SIEMENS
DR.WEB
RARE
SIEMENS Healthineers
Modulbank
TELESTE
AnchorFree
SIGNON MEMBER OF TÜV SÜD GROUP
Mentor Graphics
ΟΜΙΛΟΣ ΕΛΛΗΝΙΚΑ ΧΡΗΜΑΤΙΣΤΗΡΙΑ HELLENIC EXCHANGES GROUP
FORESIGHT
Bluehole
epochlabs
IAI
PROGNOZ
Artec 3D
EMC²


Veeam
BAE SYSTEMS
INSPIRED WORK
SIGMA
OctoNus
TEOCO
THE EMPLOYEE OWNED COMPANY
F-Secure
innovmetric
Volkswagen
PHILIPS
KASPERSKY lab
Centrify
Gaijin entertainment
NVIDIA
Cerner
WAVES
align
microOLAP
NETRONOME
NAT GAMES
IBM
NEXSTREAMING
Adobe
hp
EPIC GAMES
INEOQUEST
DR Systems
INFORM
TERADYNE
ESPRIT
c-core
ncsoft
Hex-Ray
DALET
DAYBREAK
RASTR
NATIONAL INSTRUMENTS
CSIR
VISPIRON
NEC
SUPERCELL
RSI
create, evaluate, operate
Schneider Electric
CADMATIC
jade
WARGAMING.NET
LET'S BATTLE
WE MAKE WONDERS!
neople
CRÉDIT AGRICOLE
a2ia
Prescient Technologies
FUJITSU
MAIN CONCEPT
JEPPESEN
EPSON
EXCEED YOUR VISION
EMERSON
Process Management
MERGE
An IBM Company
GEDIS
A Rohde&Schwarz Company
ZeniMax
MEDIA INC.
Viber
KS
SPORTVISION
CHANGING THE GAME
OpenConnect
adya SYSTEMS

Buy PVS-Studio

Types of licenses

Team License	Enterprise License
Best suits a small department, usually it's the first experience of using analyzers	**Suitable for multiple departments within the company**
Only for one platform: Windows, Linux or macOS	All platforms: Windows, Linux and macOS
	Support of continuous integration systems
	Integration with SonarQube
	BlameNotifier tool
	IncrediBuild support
	Ability to request custom diagnostics
The license can be purchased for 1, 2 or 3 years	The license can be purchased for 1, 2 or 3 years
E-mail reply within 48 hours	E-mail reply within 24 hours
Typically used in teams up to 9 people	Mainly used in teams of more than 10 people

Individual licenses

- We promote our product as B2B solution and we don't offer individual licenses
- Read in detail about the reasons: http://www.viva64.com/en/b/0320/
- Individual developers can use an option of **free license**
- Ways to Get a Free PVS-Studio License: https://www.viva64.com/en/b/0614/

Buy PVS-Studio

- More about licensing: https://www.viva64.com/en/order
- To order a license and get information about prices, write to us: support@viva64.com

Thank you for attention!

- Write to us: support@viva64.com
- Subscribe:
  - Twitter: @Code_Analysis
  - RSS: http://feeds.feedburner.com/viva64-blog-en
  - Facebook: https://www.facebook.com/StaticCodeAnalyzer
  - Telegram: https://t.me/pvsstudio_en
- Download PVS-Studio:

https://www.viva64.com/en/pvs-studio-download