ГОСТ Р 71207–2024 — Статический анализ программного обеспечения. Терминология

Вебинар от команды PVS-Studio

Андрей Карпов

DevRel

Андрей Карпов

- Один из основателей проекта PVS-Studio — pvs-studio.ru
- Пишу и рассказываю про статический анализ и качество кода

Термины

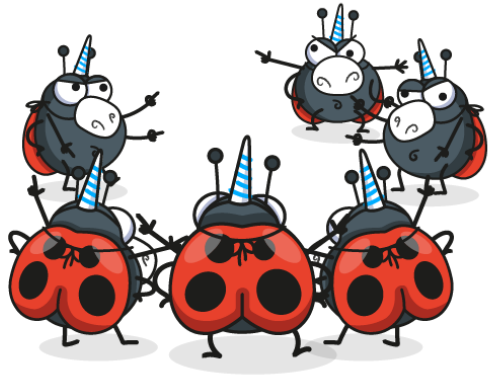
- Императивный язык программирования:
  Язык программирования, в котором используется парадигма программирования, описывающая действия над данными в терминах последовательности команд.
- Пропустим.
- Пройдёмся по интересным или полезным с практической точки зрения терминам.

Статический анализ

- Инструментальное исследование программы, основанное на анализе исходных кодов в режиме, не предусматривающем реального выполнения кода.
- Выполняется для определения свойств программы.
- В частности, статический анализ применяется для выявления ошибок в программе.

Ошибка в программе

- Конструкция или набор конструкций в исходном коде программы, наличие которой указывает на возможность реализации угроз безопасности информации и/или на ошибку реализованного в программе алгоритма.
- Ошибка в программе может:
  - быть уязвимостью программы;
  - быть программной закладкой;
  - быть ошибкой реализованного алгоритма;
  - приводить к утечке конфиденциальных данных;
  - приводить отказу программного обеспечения и пр.



Важный момент про ошибки

- В настоящем стандарте в задачи статического анализа **не входит разграничение ошибок в части последствий**, необходимо найти потенциальные места ошибок.

- В статьях мы про это время от времени пишем.
- Приятно видеть это и в ГОСТе.
- Нет смысла разбираться, к каким именно последствиям может привести та или иная ошибка.
- Её надо просто поправить!

Анализ потока данных

- Могут определяться возможные значения переменных и констант.
- Примеры артефактов:
  - переменная не инициализирована;
  - диапазон значений;
  - точное значение;
  - множество значений;
  - указатель:
    - нулевой;
    - неинициализированный;
    - указывает на буфер размера N байт;
    - память освобождена.

Анализ потока данных (RavenDB, C#)

```
public override void VisitMethod(MethodExpression expr)
{
  if (expr.Name.Value == "id" && expr.Arguments.Count == 0)
  {
    if (expr.Arguments.Count != 1)
    {
      throw new InvalidOperationException("....");
    }
```

PVS-Studio: V3022 Expression 'expr.Arguments.Count != 1' is always true. JavascriptCodeQueryVisitor.cs 188

Анализ потока управления (Far2l, C++)

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

PVS-Studio: V779 Unreachable code detected. 7z.cpp 203

Особенность PVS-Studio

- В PVS-Studio анализ потока данных и потока управления неразделимы.

- Нет повода рассматривать их по отдельности.
- Они работают вместе.

Пример (FreeCAD, C++)

```
QGVPage* QGIView::getQGVPage(TechDraw::DrawView* dView)
{
  ViewProviderPage* vpp = getViewProviderPage(dView);
  if (!vpp) {
    return vpp->getQGVPage();
  }
  return nullptr;
}
```

Анализ потока данных + анализ потока управления

PVS-Stidop: V522 Dereferencing of the null pointer 'vpp' might take place.

QGIView.cpp 592

Анализ помеченных данных

- Анализируется течение потока данных от источников до стоков.
- Основные цели обнаружить:
  - попадание данных из источников в стоки, что может означать утечку конфиденциальных данных;
  - небезопасное использование входных данных, которое может нарушить ход работы программы.

Анализ помеченных данных (NcFTP, C)

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

Воспроизведение: вводим строку с NUL в начале.

PVS-Studio: V1010 Unchecked tainted data is used in index.

https://pvs-studio.ru/ru/blog/posts/cpp/0599/

Индуктивная переменная

- Переменная цикла программы, значение которой может быть выражено в виде функции от счетчика цикла или других индуктивных переменных.
- Примечание — Индуктивные переменные, как правило, выражаются линейными функциями от счетчика цикла, изменяясь на каждой итерации цикла на фиксированное целое значение.

```
int A[10];
for (size_t i = 0; i < sizeof(A); i++)
  A[i] = 22;
```

Контролируемая сборка ПО

- Сборка ПО, при выполнении которой статическим анализатором фиксируются порядок запуска программных средств, их настройки и используемые конфигурации.

Ложноотрицательное срабатывание

- Отсутствие предупреждения об ошибке со стороны статического анализатора в ситуации наличия заведомо известной ошибки в программе, которая способна реализоваться в ходе выполнения программы.
  - В инструменте нет соответствующей диагностики.
  - Проблемы выявления высокоуровневых ошибок.
  - Вычислительная сложность.
  - Неразрешимость задачи (проблема останова).

https://pvs-studio.ru/ru/blog/terms/6460/

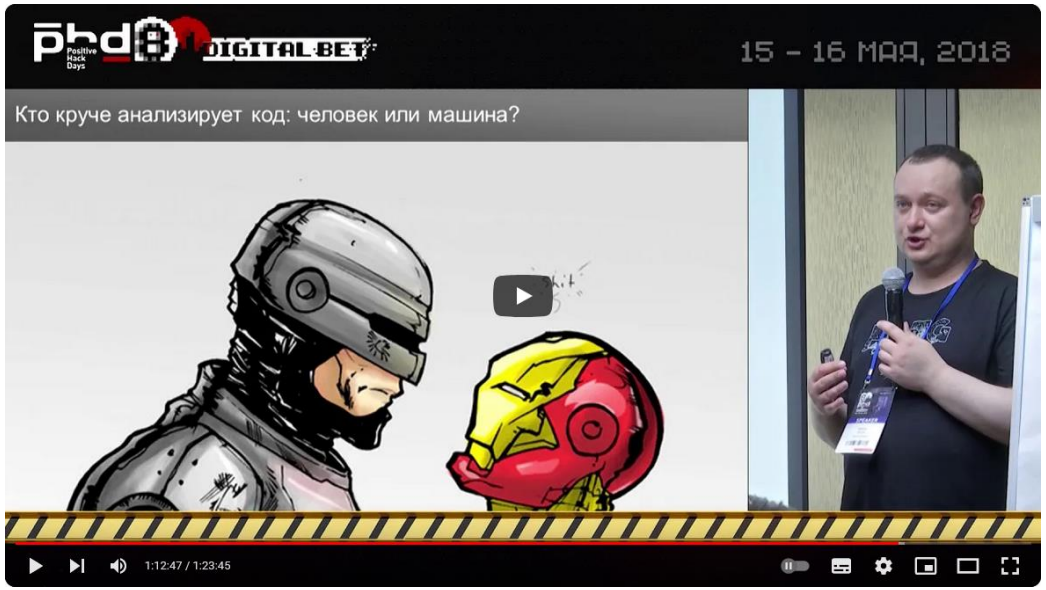
Ложноотрицательное срабатывание: высокий уровень ошибки

- Анализатор не знает, что и как должна делать программа.
- Невозможно угадать, что человек ошибся в формуле и вместо *sin(a)* нужно было использовать *cos(a)*.

Ложноотрицательное срабатывание: сложность

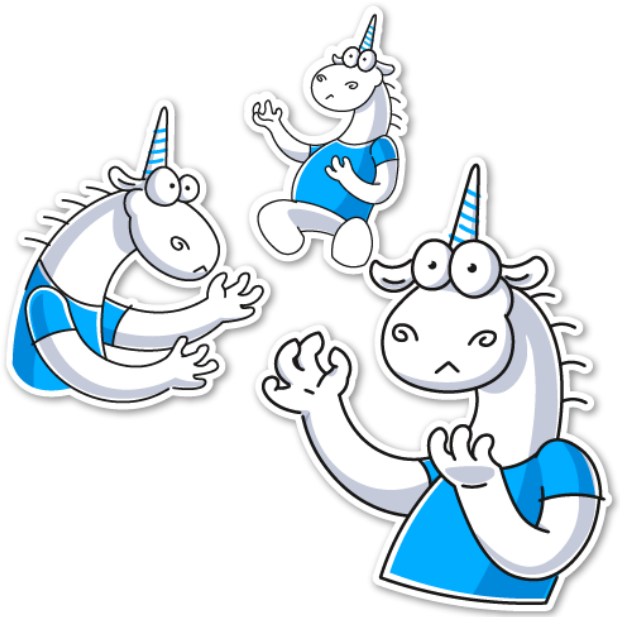
- Медленно или вообще неразрешимо (проблема останова).

- Владимир Кочетков. Идеальный статический анализ. https://youtu.be/RdeElwDJ3tg?si=lcO9jjiCA0grGXS2



Ложноположительное срабатывание

- Выданное статическим анализатором предупреждение, которое указывает на конструкцию или несколько конструкций в программе, не содержащих ошибку, которая может реализоваться в ходе выполнения программы.



https://pvs-studio.ru/ru/blog/terms/6461/

Сигнатурный анализ

- Мы в статьях называли это «поиск по шаблону».
- Определение наличия свойств программы при помощи поиска строк в исходном коде по некоторому образцу, в том числе заданному с помощью формального языка поиска.
- Например, при помощи регулярных выражений.

Статический анализ это что-то типа регулярок?

- **Нет!**

- Это очень маленькая его часть.
- Скорее всего, это не будут регулярные выражения в чистом виде (используется AST).

Сигнатурный анализ (Qemu, C)

```
target_ulong helper_mftc0_cause(CPUMIPSState *env)
{
  ....
  if (other_tc == other->current_tc) {
    tccause = other->CP0_Cause;
  } else {
    tccause = other->CP0_Cause;
  }
  ....
}
```

PVS-Studio: V523 The 'then' statement is equivalent to the 'else' statement. cp0_helper.c 383

Сигнатурный анализ (NetBeans, Java)

```
private RevisionInterval
getResortedRevisionInterval(SVNRevision revision1,
                            SVNRevision revision2) {
  RevisionInterval ret;
  if(revision1.equals(SVNRevision.HEAD) &&
     revision1.equals(SVNRevision.HEAD)) {
    ....
  }
```

PVS-Studio: V6001 There are identical sub-expressions 'revision1.equals(SVNRevision.HEAD)' to the left and to the right of the '&&' operator. RevertModifications.java(387)

Статистический анализ (Linux Kernel, C)

```
static const
struct XGI330_LCDDataDesStruct2  XGI_LVDSNoScalingDesData[] = {
  {0,  648,  448,  405,  96, 2}, /* 00 (320x200,320x400,
                                    640x200,640x400) */
  {0,  648,  448,  355,  96, 2}, /* 01 (320x350,640x350) */
  {0,  648,  448,  405,  96, 2}, /* 02 (360x400,720x400) */
  {0,  648,  448,  355,  96, 2}, /* 03 (720x350) */
  {0,  648,    1,  483,  96, 2}, /* 04 (640x480x60Hz) */
  {0,  840,  627,  600, 128, 4}, /* 05 (800x600x60Hz) */
  {0, 1048,  805,  770, 136, 6}, /* 06 (1024x768x60Hz) */
  {0, 1328,    0, 1025, 112, 3}, /* 07 (1280x1024x60Hz) */
  {0, 1438,    0, 1051, 112, 3}, /* 08 (1400x1050x60Hz)*/
  {0, 1664,    0, 1201, 192, 3}, /* 09 (1600x1200x60Hz) */
  {0, 1328,    0, 0771, 112, 6}  /* 0A (1280x768x60Hz) */
};
```

PVS-Studio: V536 Be advised that the utilized constant value is represented by an octal form. Oct: 0771, Dec: 505. vb_table.h 1379

Модельный вариант (ошибки)

- Подтип некоторого типа ошибки в программе, отнесение к которому осуществляется исходя из особенностей реализации ошибки данного типа и особенностей языка программирования.
- Разбиение типа ошибки на модельные варианты применяется из-за технологических ограничений статического анализа.
- Примеры:
  - поиск неопределённого поведения;
  - поиск опечаток;
  - поиск разыменований нулевого указателя.

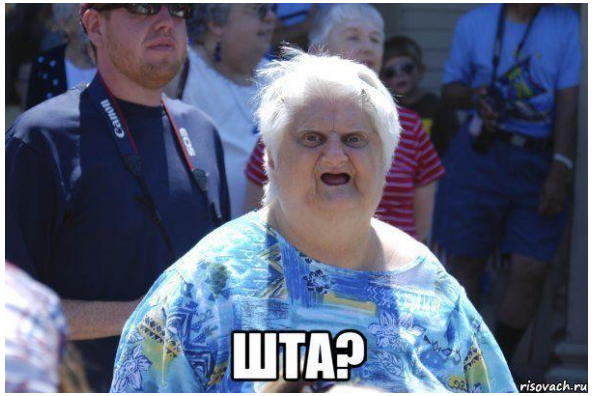
Модельный вариант ошибки (MuseScore, C++)

```
Ms::Segment* NotationSelectionRange::rangeStartSegment() const {
  Ms::Segment* startSegment = score()->selection().startSegment();

  startSegment->measure()->firstEnabled();

  if (!startSegment) {
    return nullptr;
  }
```

PVS-Studio: V595 The 'startSegment' pointer was utilized before it was verified against nullptr. Check lines: 129, 131. notationselectionrange.cpp 129

Есть отдельные термины, которые я не очень понял

- **Поточный вариант (ошибки)**: Способ выражения ошибки в исходном коде программы через конкретную структуру потоков управления и данных.
- Есть примечание. Поточные варианты являются общими для всех типов ошибок, так как характеризуют не саму ошибку, а способ её выражения в исходном коде программы. (Понятнее не стало :)



Критическая ошибка (важнейшее понятие)

- Ошибка, которая может привести к нарушению безопасности обрабатываемой информации.
- Мы в статьях называли такие ошибки потенциальными уязвимостями.

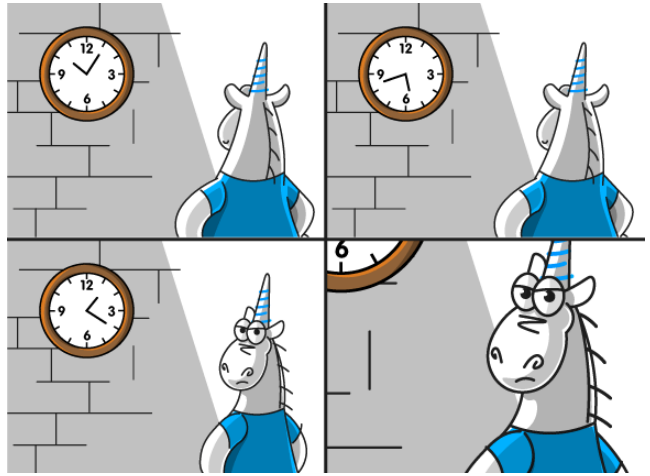
Чем полезно понятие «критическая ошибка»?

Классификация предупреждений

- Предупреждений от анализаторов всегда больше, чем хотелось бы.
- Проблематика, о которой слышал: сертификационная лаборатория присылает отчёт с большим количеством предупреждений.
- Как ответить, что не всё критично?
- Чем руководствоваться самой лаборатории?
- Теперь можно руководствоваться ГОСТом и понятием «критическая ошибка».
- Можно выделить соответствующее подмножество диагностических правил.

PVS-Studio и критические ошибки

- PVS-Studio умеет выявлять критические ошибки;
- Пока нет возможности выбрать подмножество критических ошибок;
- Сделаем.



Однако помним: «некритическая ошибка» тоже может быть опасной

- К классификации следует относиться вдумчиво.
- Всё условно и относительно.

- Формально верно найденная критическая ошибка может не представлять угрозы.
- Ошибка, не классифицированная как критическая, вполне может такой быть.

```
static bool samu_correct(struct samu *s1, struct samu *s2)
{
  ....
  } else if (s1_len != s1_len) {
    DEBUG(0, ("Password history not written correctly, "
              "lengths differ, want %d, got %d\n",
          s1_len, s2_len));
    ret = False;
```

Samba, C

- Ошибка найдена как **модельный вариант опечатки**.
  PVS-Studio: V501 There are identical sub-expressions to the left and to the right of the '!=' operator: s1_len != s1_len pdbtest.c 106
- Но фактически это может быть **ошибкой непроверенного использования чувствительных данных**.

Заключение

- Следующий вебинар как раз будет посвящен критическим ошибкам.

- Пишите нам, если хотите узнать про какие-то темы подробнее. Мы собираем идеи для будущих вебинаров.

- Спасибо за внимание и подписывайтесь:
  - Telegram канал: https://t.me/**pvsstudio_rus**
  - Ежемесячный дайджест наших статей: **https://pvs-studio.ru/ru/subscribe/**

Q&A

Андрей Карпов

DevRel