Автоматический поиск ошибок

Владимир Кошелев

Институт системного программирования РАН

vedun@ispras.ru

21 февраля 2017 г.

План

Введение

Об ИСП РАН

Чем мы занимаемся:

- Анализ исходного кода программ для поиска ошибок:
  - C/C++
  - Java
  - C#

Об ИСП РАН

Чем мы занимаемся:

- Анализ исходного кода программ для поиска ошибок:
  - C/C++
  - Java
  - C#
- Анализ бинарного кода

Об ИСП РАН

Чем мы занимаемся:

- Анализ исходного кода программ для поиска ошибок:
  - C/C++
  - Java
  - C#
- Анализ бинарного кода
- Разработка GCC

Об ИСП РАН

Чем мы занимаемся:

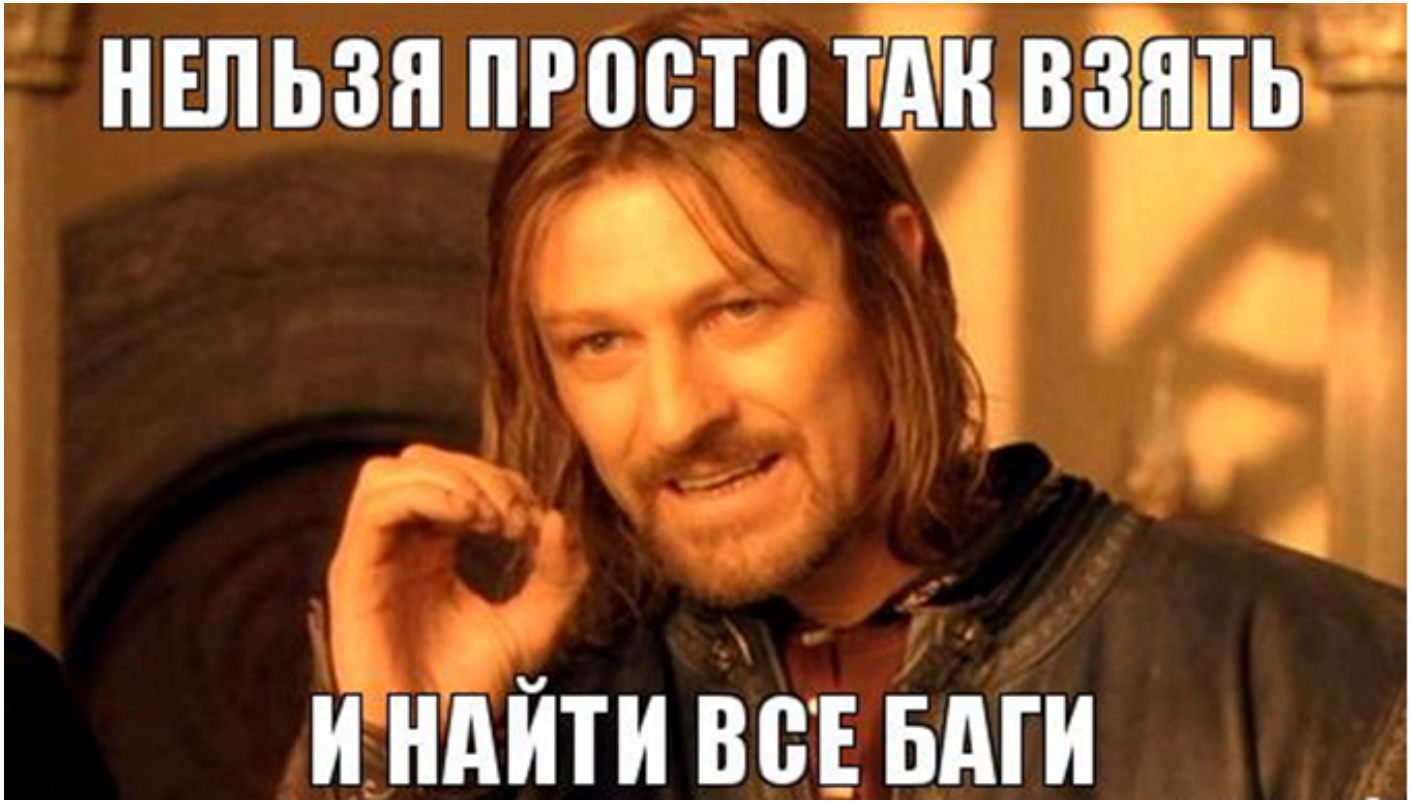
- Анализ исходного кода программ для поиска ошибок:
  - C/C++
  - Java
  - C#
- Анализ бинарного кода
- Разработка GCC
- Разработка QEMU
- ...

Теорема Райса

В чем заключается основная сложность анализа программ?

Теорема Райса

В чем заключается основная сложность анализа программ?



Постановки задачи при поиске багов

- Находим ***все*** баги и имеем ложные срабатывания:
  - Верификация

Постановки задачи при поиске багов

- Находим ***все*** баги и имеем ложные срабатывания:
  - Верификация
- Находим ***не все*** баги и имеем ложные срабатывания:
  - Статический анализ

Постановки задачи при поиске багов

- Находим ***все*** баги и имеем ложные срабатывания:
  - Верификация
- Находим ***не все*** баги и имеем ложные срабатывания:
  - Статический анализ
- Находим ***только*** баги, но найдем явно ***не все***:
  - Динамический анализ
  - Смешанный анализ

Верификация

Верификация: поиск неинициализированных переменных

```
public string Foo(string a, string b) {
    string min;
    int cmpResult = a.CompareTo(b);

    if (cmpResult < 0)
        min = a;

    if (cmpResult >= 0)
        min = b;

    return min;
}
```

Верификация: поиск неинициализированных переменных

```
public string Foo(string a, string b) {
    string min;
    int cmpResult = a.CompareTo(b);

    if (cmpResult < 0) // False
        min = a;

    if (cmpResult >= 0) // False
        min = b;

    return min;
}
```

"Верификация"в CodeContracts

Пример контрактов

```
public string Greet(string[] args)
{
    Contract.Requires(args != null);
    Contract.Requires(args.Length > 0);
    Contract.Requires(args[0] != null);
    Contract.Ensures(Contract.Result<string>().Length
        > 5);

    Console.WriteLine("Hello, {0}!", args[0]);
    return "abacaba" + args[0];
}
```

Roslyn: используем на практике

Пару слов о Roslyn

Практическое применение Roslyn: боремся с boilerplate

Entitas - The Entity Component System Framework for C# and Unity

Задача: автоматически генерировать boilerplate код для работы с компонентами.

Практическое применение Roslyn: боремся с boilerplate

Entitas - The Entity Component System Framework for C# and Unity

Задача: автоматически генерировать boilerplate код для работы с компонентами.

```
public class PositionComponent : IComponent
{
    public float X;
    public float Y;
}
```

Практическое применение Roslyn: боремся с boilerplate

Entitas - The Entity Component System Framework for C# and Unity

Задача: автоматически генерировать boilerplate код для работы с компонентами.

```
public class PositionComponent : IComponent
{
    public float X;
    public float Y;
}

public void AddPosition(float newX, float newY) {
    var component =
        CreateComponent<PositionComponent>
            (BattleComponentsLookup.Position);
    component.X = newX;
    component.Y = newY;
    AddComponent(BattleComponentsLookup.Position,
        component);
}
```

Генерация через Reflection

```
var types = Assembly
            .GetAssembly(typeof(MainClass))
            .GetTypes();

foreach (var type in types)
{
    Process(type);
}
```

Генерация через Reflection

```
var types = Assembly
            .GetAssembly(typeof(MainClass))
            .GetTypes();

foreach (var type in types)
{
    Process(type);
}
```

Главная проблема: генерация работает **только** если проект компилируется.

Если вы хотите рефакторить, нужно исправлять все ошибки в сгенерированном коде.

Генерация через Roslyn

```
using (var workspace = MSBuildWorkspace.Create())
{
    var solution =
      workspace.OpenSolutionAsync(path).Result;

    foreach (var project in solution.Projects){
      var compilation =
          project.GetCompilationAsync().Result;

      var allTypes =
          compilation.GetSymbolsWithName(x => true,
              SymbolFilter.Type).OfType<ITypeSymbol>();

      foreach (var type in types)
          Process(type)
    }
}
```

Исключение на Mac/Mono

```
Unhandled Exception:
System.Reflection.ReflectionTypeLoadException: The classes in the module cannot be
      loaded.
  at (wrapper managed-to-native) System.Reflection.Assembly:GetTypes
       (System.Reflection.Assembly,bool)
  at System.Reflection.Assembly.GetTypes () [0x00000] in
       <8f2c484307284b51944a1a13a14c0266>:0
  at System.Reflection.Assembly+<>c__Iterator0.MoveNext () [0x00021] in
       <8f2c484307284b51944a1a13a14c0266>:0
  at System.Linq.Enumerable+WhereSelectEnumerableIterator`2[TSource,TResult].MoveNext ()
       [0x00078] in <63992662b765477a898ef49cdcc99ee2>:0
  at System.Linq.Enumerable+<SelectManyIterator>c__Iterator2`2[TSource,TResult].MoveNext
       () [0x000bc] in <63992662b765477a898ef49cdcc99ee2>:0
  at System.Composition.TypedParts.TypedPartExportDescriptorProvider..ctor
       (System.Collections.Generic.IEnumerable`1[T] types,
       System.Composition.Convention.AttributedModelProvider attributeContext) [0x00049]
       in <c091afde214c4b8e8efbbeb9d44062d4>:0
  at System.Composition.Hosting.ContainerConfiguration.CreateContainer () [0x00042] in
       <c091afde214c4b8e8efbbeb9d44062d4>:0
  at Microsoft.CodeAnalysis.Host.Mef.MefHostServices.Create
       (System.Collections.Generic.IEnumerable`1[T] assemblies) [0x00019] in
       <947408acdcbc4eba93cdb6a19d92fcd1>:0
  at Microsoft.CodeAnalysis.Host.Mef.DesktopMefHostServices.get_DefaultServices ()
       [0x00011] in <2ba3a08d4fe0467c8c54c8bc37c61987>:0
  at Microsoft.CodeAnalysis.MSBuild.MSBuildWorkspace.Create
       (System.Collections.Generic.IDictionary`2[TKey,TValue] properties) [0x00000] in
       <2ba3a08d4fe0467c8c54c8bc37c61987>:0
  at Microsoft.CodeAnalysis.MSBuild.MSBuildWorkspace.Create () [0x00000] in
       <2ba3a08d4fe0467c8c54c8bc37c61987>:0
  ...
```

Статический анализ

Зачем разработчикам нужен статический анализ

- Статический анализ используется как для поиска ошибок до тестирования, так и для поиска ошибок, пропущенных при тестировании

Зачем разработчикам нужен статический анализ

- Статический анализ используется как для поиска ошибок до тестирования, так и для поиска ошибок, пропущенных при тестировании
- Чем больше размер команды, тем эффективнее работает статический анализ

Зачем разработчикам нужен статический анализ

- Статический анализ используется как для поиска ошибок до тестирования, так и для поиска ошибок, пропущенных при тестировании
- Чем больше размер команды, тем эффективнее работает статический анализ
- Применение сложного статического анализа предполагает высокую цену ошибки

Зачем разработчикам нужен статический анализ

- Статический анализ используется как для поиска ошибок до тестирования, так и для поиска ошибок, пропущенных при тестировании
- Чем больше размер команды, тем эффективнее работает статический анализ
- Применение сложного статического анализа предполагает высокую цену ошибки
- Статический анализ хорошо сочетается с CodeReview

Требования к статическим анализаторам

- Время работы

Требования к статическим анализаторам

- Время работы
- Соотношение "выдано предупреждений/внесено исправлений"

Требования к статическим анализаторам

- Время работы
- Соотношение "выдано предупреждений/внесено исправлений"
- Поддерживаемые классы ошибок, включая подклассы

Требования к статическим анализаторам

- Время работы
- Соотношение "выдано предупреждений/внесено исправлений"
- Поддерживаемые классы ошибок, включая подклассы
- Понятность сообщений об ошибках

Требования к статическим анализаторам

- Время работы
- Соотношение "выдано предупреждений/внесено исправлений"
- Поддерживаемые классы ошибок, включая подклассы
- Понятность сообщений об ошибках
- Интеграция с CI

Инструменты статического анализа

- Coverity
- SharpChecker (наш инструмент)
- SonarLint
- ReSharper
- PVS-Studio

Coverity

- Де факто стандарт качества для инструментов статического анализа
- Продается только за большие деньги
- Бесплатен для OpenSource на https://scan.coverity.com
- Все виды анализа: AST, внутрипроцедруный, межпроцедурынй

SharpChecker

- Статический анализ на основе Roslyn, разработанный в ИСП РАН.
- Цель проекта: разработать инструмент, соответствующий уровню лучших современных статических анализаторов.
- Все виды анализа: AST, внутрипроцедруный, межпроцедурынй.
- http://sharpchecker.ispras.ru

SonarLint for VisualStudio

- Бесплатная коллекция анализаторов для Roslyn
- Интеграция с SonarQube
- Интеграция с сборкой msbuild
- Более 100 AST анализаторов
- Интеграция с Visual Studio

Сравнение ReSharper и PVS-Studio

	ReSharper	PVS-Studio

Сравнение ReSharper и PVS-Studio

	ReSharper	PVS-Studio
Количество статей на харбрахабре за февраль	0	8

Сравнение ReSharper и PVS-Studio

	ReSharper	**PVS-Studio**
Количество статей на харбрахабре за февраль	0	8
Наличие AST-детекторов	Да	Да

Сравнение ReSharper и PVS-Studio

	ReSharper	**PVS-Studio**
Количество статей на харбрахабре за февраль	0	8
Наличие AST-детекторов	Да	Да
Наличие внутрипроцедурного анализа потоков данных	Да	Нет

Сравнение ReSharper и PVS-Studio

	ReSharper	**PVS-Studio**
Количество статей на харбрахабре за февраль	0	8
Наличие AST-детекторов	Да	Да
Наличие внутрипроцедурного анализа потоков данных	Да	Нет
Анализ во время разработки	Да	Нет

Сравнение ReSharper и PVS-Studio

	ReSharper	**PVS-Studio**
Количество статей на харбрахабре за февраль	0	8
Наличие AST-детекторов	Да	Да
Наличие внутрипроцедурного анализа потоков данных	Да	Нет
Анализ во время разработки	Да	Нет
Интеграция с CI	Да	Да

Сравнение ReSharper и PVS-Studio

	ReSharper	**PVS-Studio**
Количество статей на харбрахабре за февраль	0	8
Наличие AST-детекторов	Да	Да
Наличие внутрипроцедурного анализа потоков данных	Да	Нет
Анализ во время разработки	Да	Нет
Интеграция с CI	Да	Да
Хорошая документация	Нет	Да

Динамический анализ

PEX / IntelliTest

http://www.pexforfun.com/

Идея работы IntelliTest

Алгоритм работы:

- запустим метод с какими-нибудь параметрами;

Идея работы IntelliTest

Алгоритм работы:

- запустим метод с какими-нибудь параметрами;
- построим формулу, которая описывает его выполнение;

Идея работы IntelliTest

Алгоритм работы:

- запустим метод с какими-нибудь параметрами;
- построим формулу, которая описывает его выполнение;
- решим задачу: какими должны быть параметры, чтобы выполнение прошло по другой ветке и увеличило покрытие;

Идея работы IntelliTest

Алгоритм работы:

- запустим метод с какими-нибудь параметрами;
- построим формулу, которая описывает его выполнение;
- решим задачу: какими должны быть параметры, чтобы выполнение прошло по другой ветке и увеличило покрытие;
- запустим с новыми данными.

Спасибо за внимание!

Владимир Кошелев
vedun@ispras.ru

Ссылки:
SharpChecker: http://sharpchecker.ispras.ru
Coverity: https://scan.coverity.com
SonarLint : http://www.sonarlint.org/visualstudio/