



Подводные камни регулярных выражений: катастрофический возврат, ReDoS-атаки и выявление уязвимостей



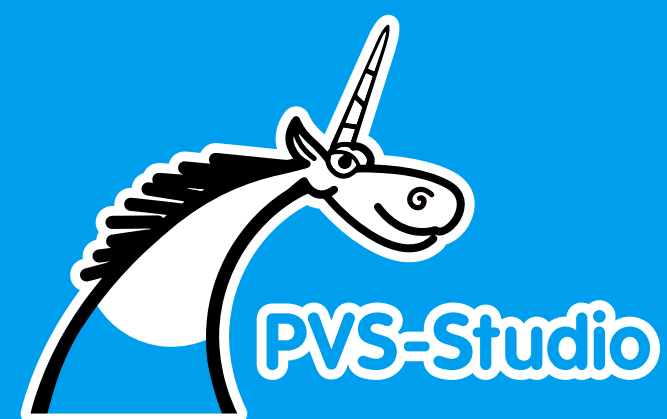
Глеб
Асламов



Георгий
Тормозов



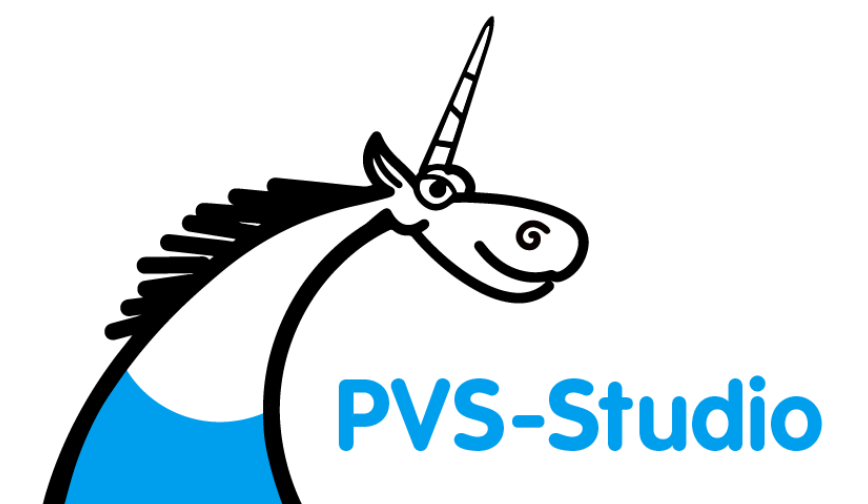
Спикеры вебинара



Георгий Тормозов

Middle C# Developer

- Разрабатывает C# и Go анализатор
- Пишет технические статьи про проверку Open-Source проектов
- Любит анекдоты и котов

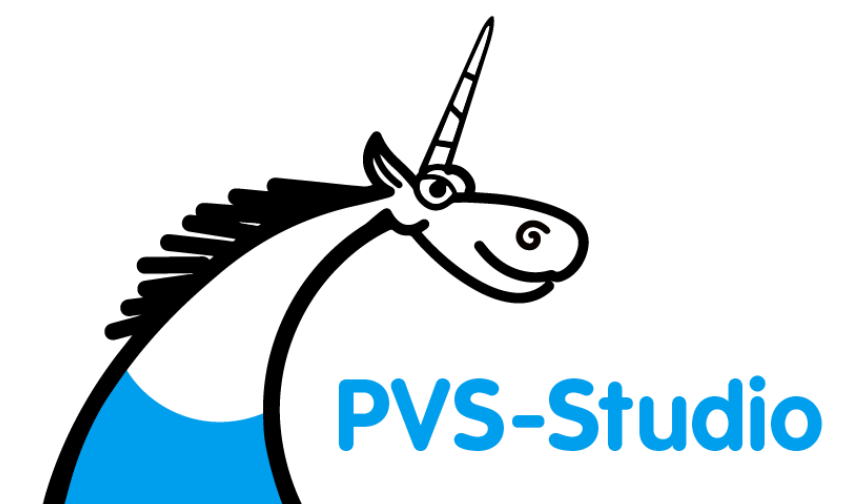


Георгий Тормозов

Middle C# Developer

Доклад: «Подводные камни регулярных выражений»

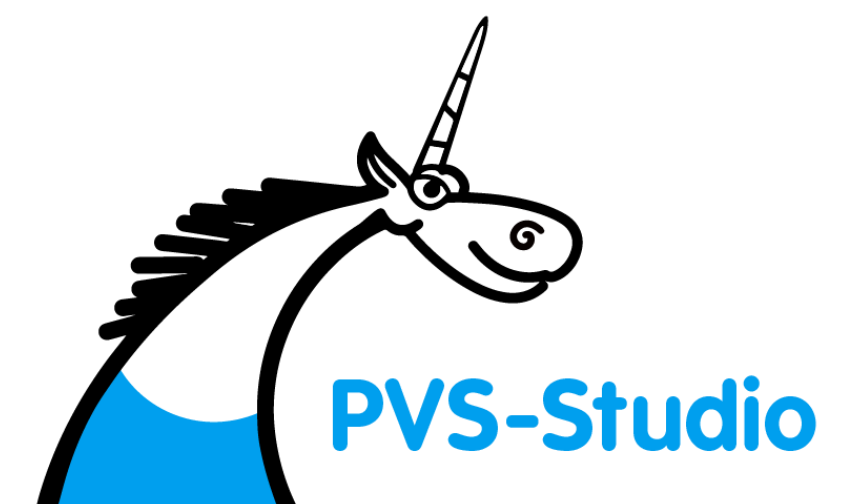
- Так ли безопасно использовать регулярные выражения на регулярной основе?
- Какие есть слабые места и каким образом они приводят к уязвимостям?
- ReDoS-атаки и не только



Глеб Асламов

Senior Developer Advocate, C# Developer

- C# и Go разработчик в PVS-Studio.
- Выступаю на конференциях, рассказываю про качество кода и безопасную разработку
- Иногда бываю ведущим вебинаров (вот это совпадение)



Глеб Асламов

Senior Developer Advocate, C# Developer

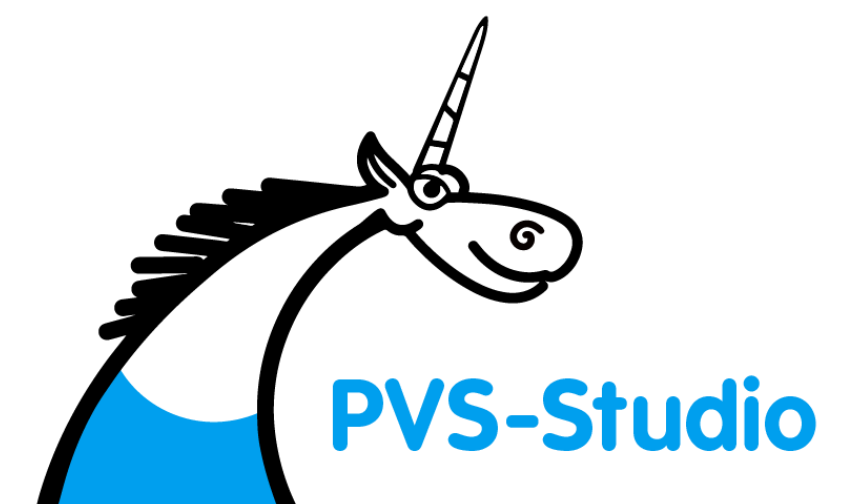
Доклад: «**Ищем уязвимый код**»

Копнем поглубже в мир проблем в коде

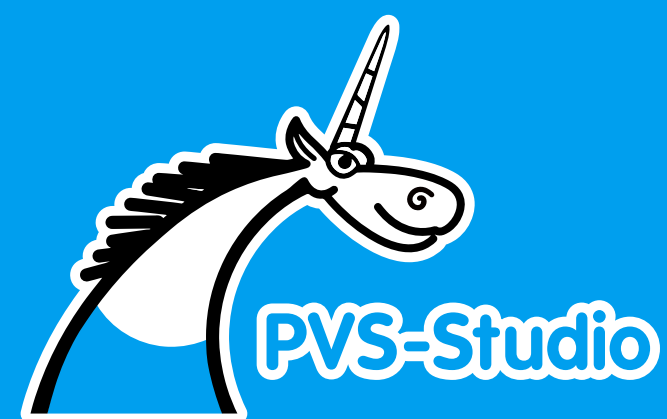
И немного в мир безопасной разработки :)

Узнаем ответы на разные вопросы:

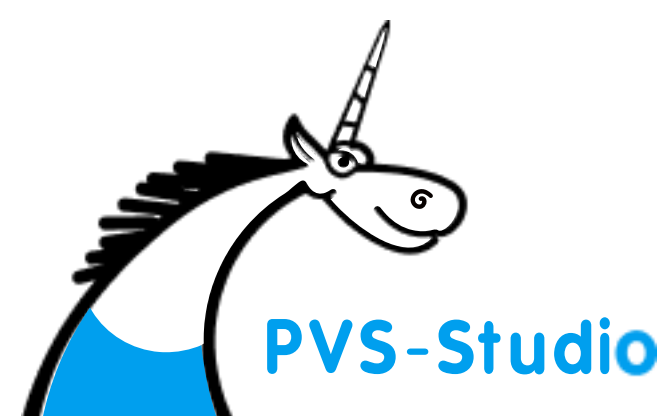
- Какие бывают уязвимости
- Как их найти?
- И причем тут ГОСТ...



Переходим к докладам



Ищем уязвимый КОД



Глеб Асламов
C# Developer Advocate



Глеб Асламов

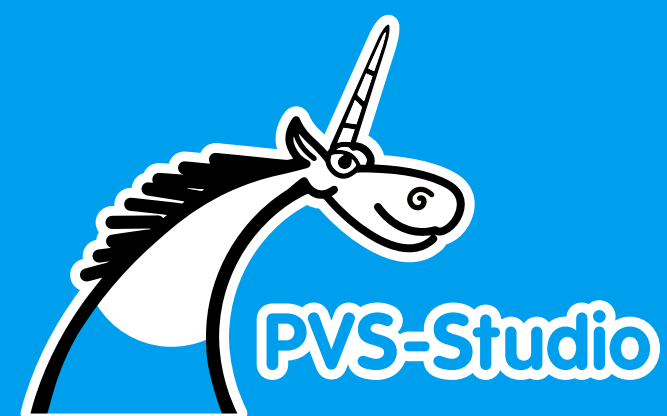
Senior Developer Advocate

- C# \ Go разработчик
- Выступаю на конференциях:
SQA Days , Стачка, DotNext, UIC Dev,
DevFest , etc.
- Один из создателей и ведущих курса по
ГОСТ Р 56939—2024



PVS-Studio

Бывают другие уязвимости?



Пример уязвимости в проекте FastReport

12

```
ParagraphFormat paragraphFormat;
```

```
...
```

```
public ParagraphFormat ParagraphFormat  
{  
    get { return paragraphFormat; }  
    set { ParagraphFormat = value; }  
}
```

Пример уязвимости в проекте FastReport

13

```
ParagraphFormat paragraphFormat;
```

```
...
```

```
public ParagraphFormat ParagraphFormat  
{  
    get { return paragraphFormat; }  
    set { ParagraphFormat = value; }  
}
```

Пример уязвимости в проекте FastReport

14

```
ParagraphFormat paragraphFormat;
```

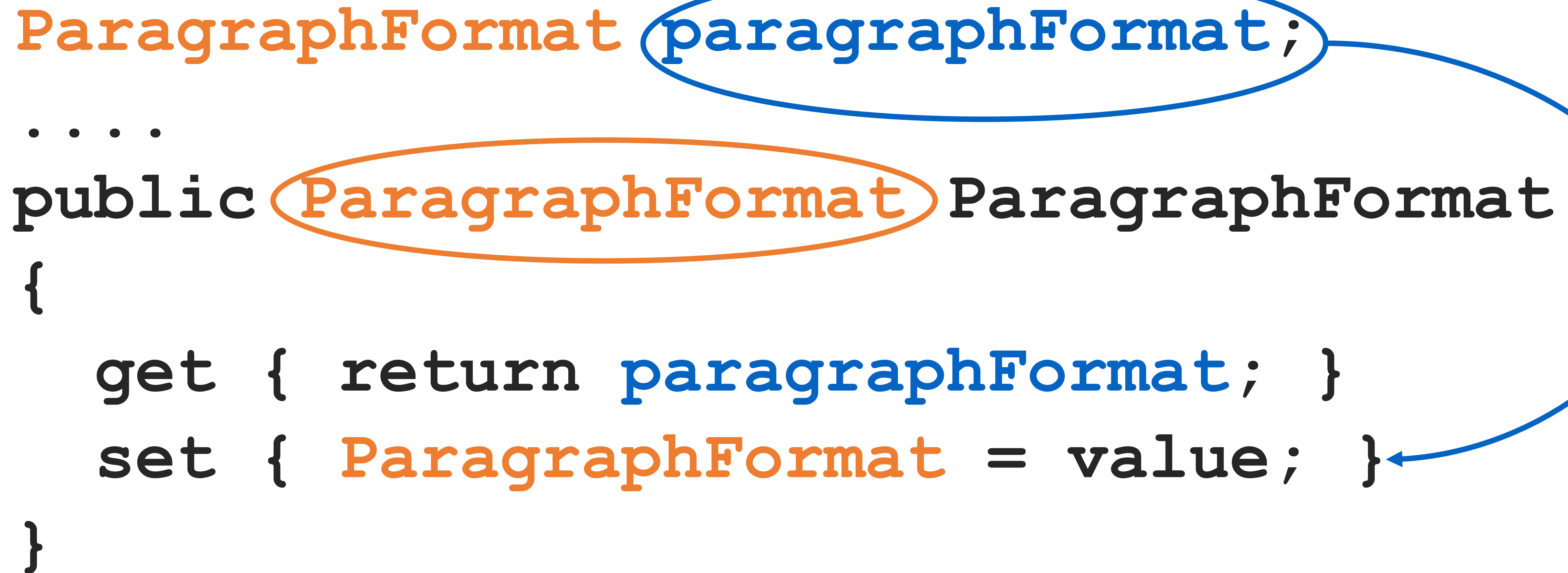
```
...
```

```
public ParagraphFormat ParagraphFormat  
{  
    get { return paragraphFormat; }  
    set { ParagraphFormat = value; }  
}
```


Пример уязвимости в проекте FastReport

15

```
ParagraphFormat paragraphFormat;  
...  
public ParagraphFormat ParagraphFormat  
{  
    get { return paragraphFormat; }  
    set { ParagraphFormat = value; }  
}
```



Пример уязвимости в проекте FastReport

16

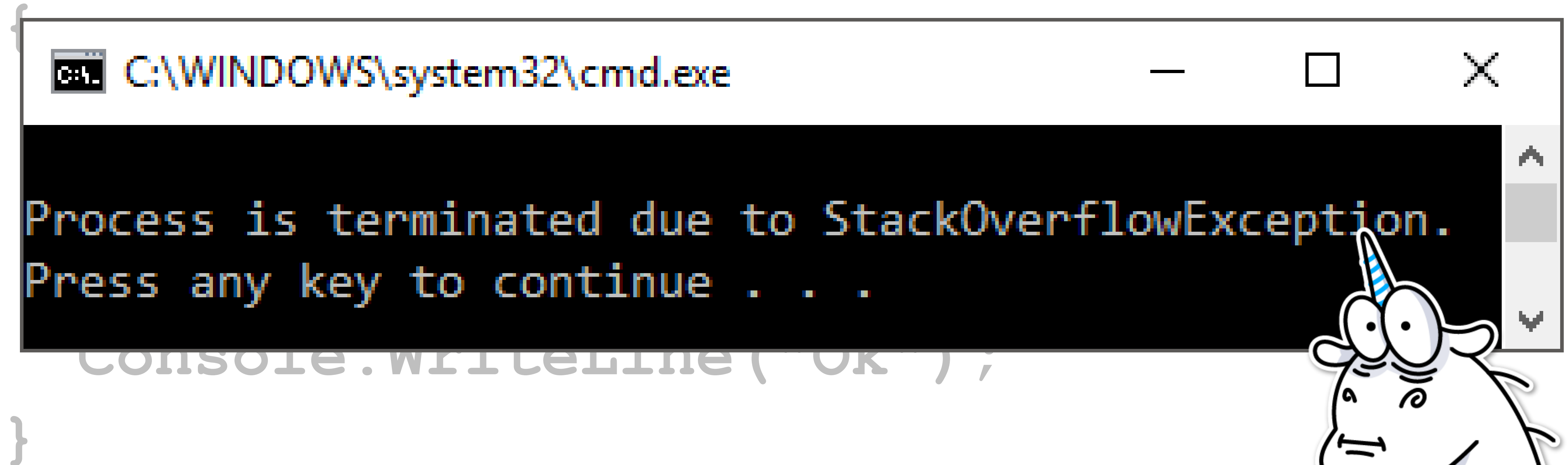
```
static void Main(string[] args)
{
    TextObject textObj = new TextObject();
    textObj.ParagraphFormat = null;

    Console.WriteLine("Ok");
}
```

Пример уязвимости в проекте FastReport

17

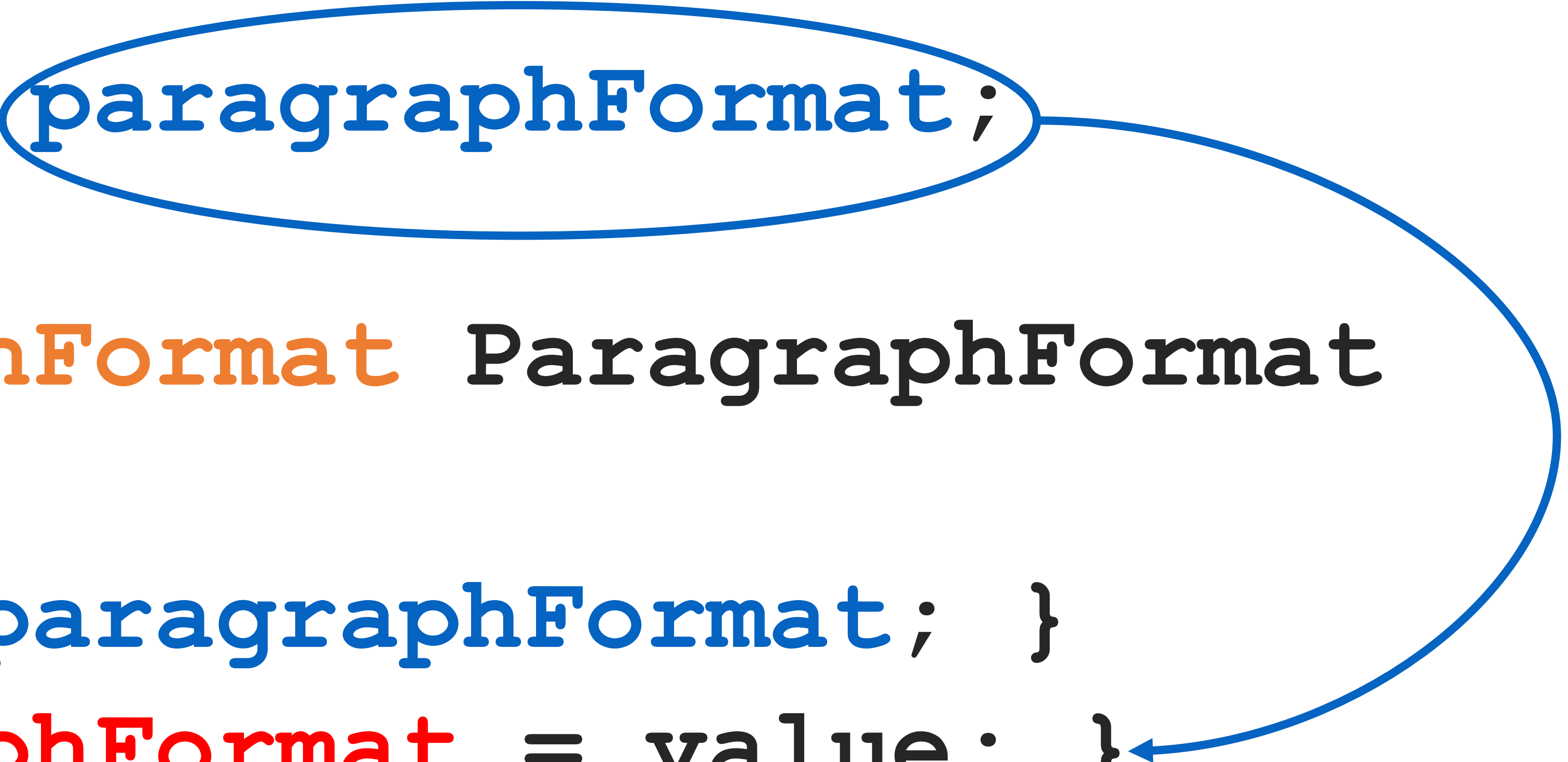
```
static void Main(string[] args)
```



Пример уязвимости в проекте FastReport

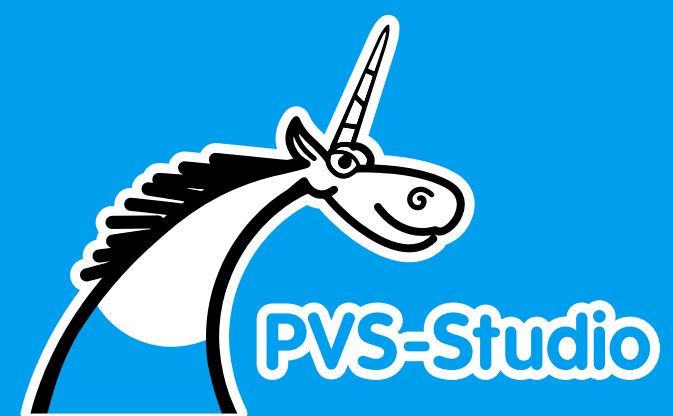
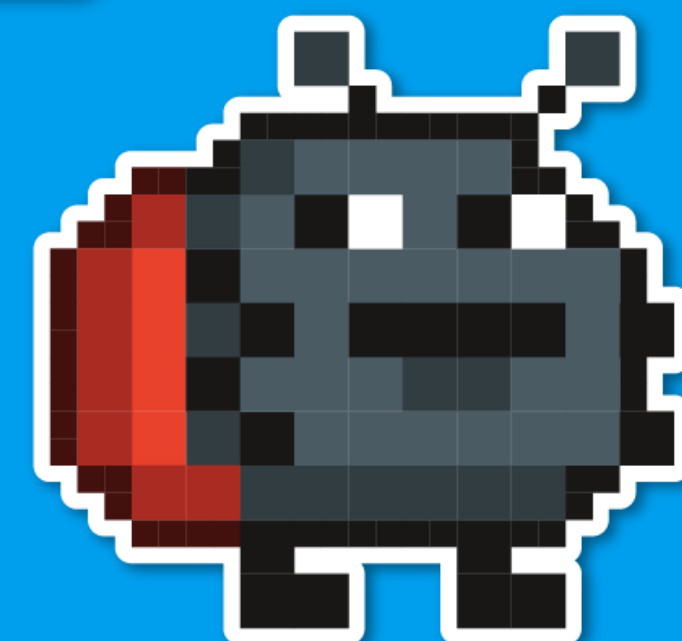
18

```
ParagraphFormat paragraphFormat;  
...  
public ParagraphFormat ParagraphFormat  
{  
    get { return paragraphFormat; }  
    set { ParagraphFormat = value; }  
}
```



V3010 [CWE-674] Possible infinite recursion inside 'ParagraphFormat' property.

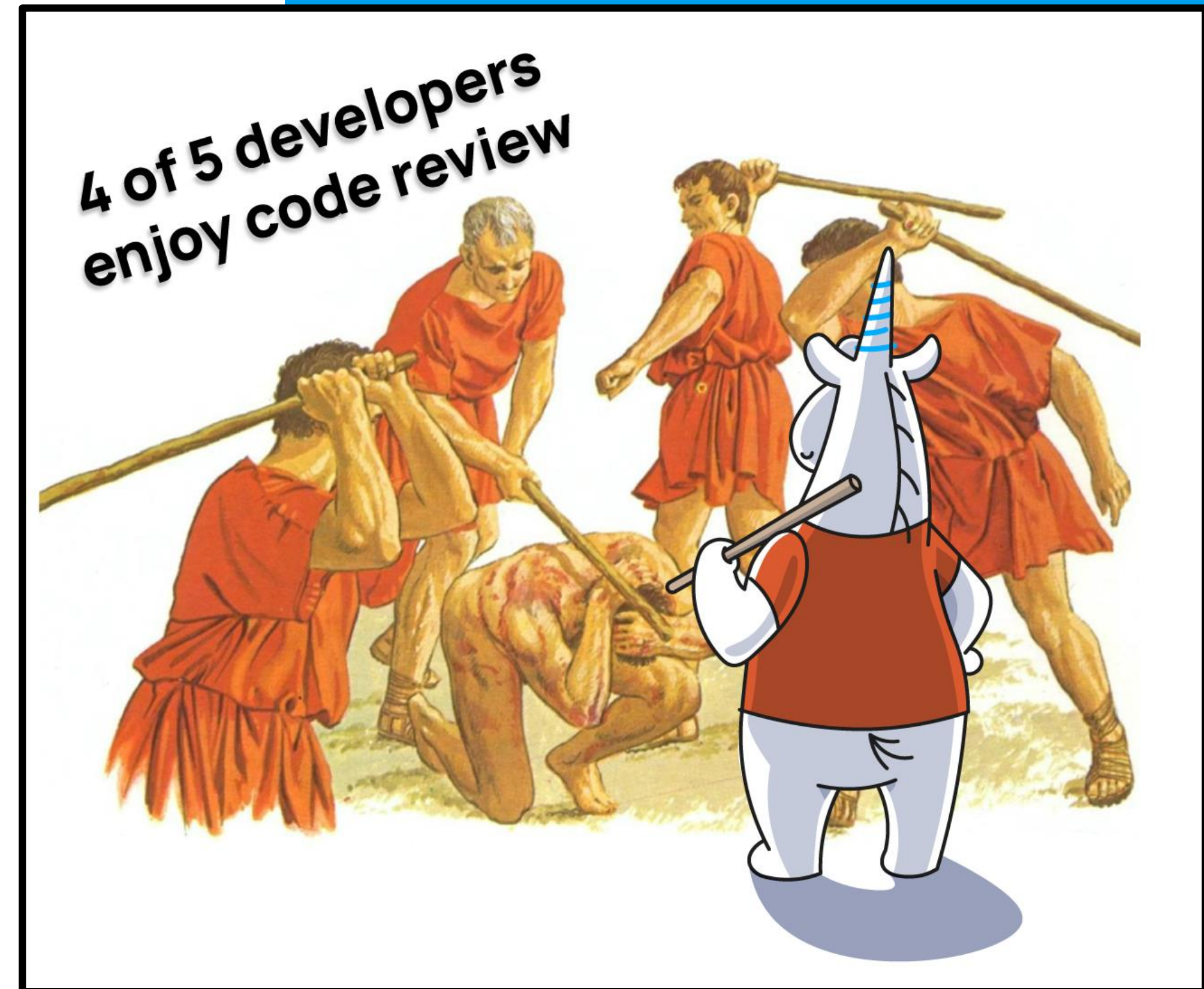
Но как их найти?



Статический анализ



- «*Автоматический код-ревью*»
- Нужен только код
- Полное покрытие
- Раннее обнаружение ошибок
- Ошибки исправляются на этапе разработки





Пример уязвимости SQL injection

23

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

Пример уязвимости SQL injection

24

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

Пример уязвимости SQL injection

25

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```


Пример уязвимости SQL injection

26

```
using (SqlConnection connection = new SqlConnection(...))  
{
```

```
    ...
```

```
    string query = "SELECT * FROM Users WHERE UserName = 'Иван'";  
    SqlCommand command = new SqlCommand(query, connection);  
    connection.Open();  
    SqlDataReader reader = command.ExecuteReader();  
    while (reader.Read())  
    {  
        ...  
    }  
}
```

Пример уязвимости SQL injection

27

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

Пример уязвимости SQL injection

28

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

Пример уязвимости SQL injection

29

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

Пример уязвимости SQL injection

30

```
using (SqlConnection connection = new SqlConnection(...))  
{  
    ....  
    String userName = Request.QueryString["name"];  
    using (var command = new SqlCommand(  
        Connection = connection,  
        CommandText =  
            "SELECT * FROM Users WHERE name = '" + userName + "'",  
        CommandType = System.Data.CommandType.Text ))  
    ....  
}
```

Иван



Пример уязвимости SQL injection

31

```
using (SqlConnection connection = new SqlConnection(...))  
{  
    ....  
    String userName = Request.Form["userName"];  
    using (var command = new SqlCommand("SELECT * FROM users WHERE userName = @userName", connection))  
    {  
        command.CommandType = CommandType.Text;  
        command.Parameters.AddWithValue("@userName", userName);  
        ....  
    }  
}
```

' OR '1'='1'



Пример уязвимости SQL injection

32

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (SqlCommand command = new SqlCommand(
        "SELECT * FROM Users WHERE UserName = 'Иван'",
        connection))
    {
        CommandType = CommandType.Text;
        ....
    }
}
```

Пример уязвимости SQL injection

33

```
using (SqlConnection connection = new SqlConnection(...))  
{
```

```
....
```

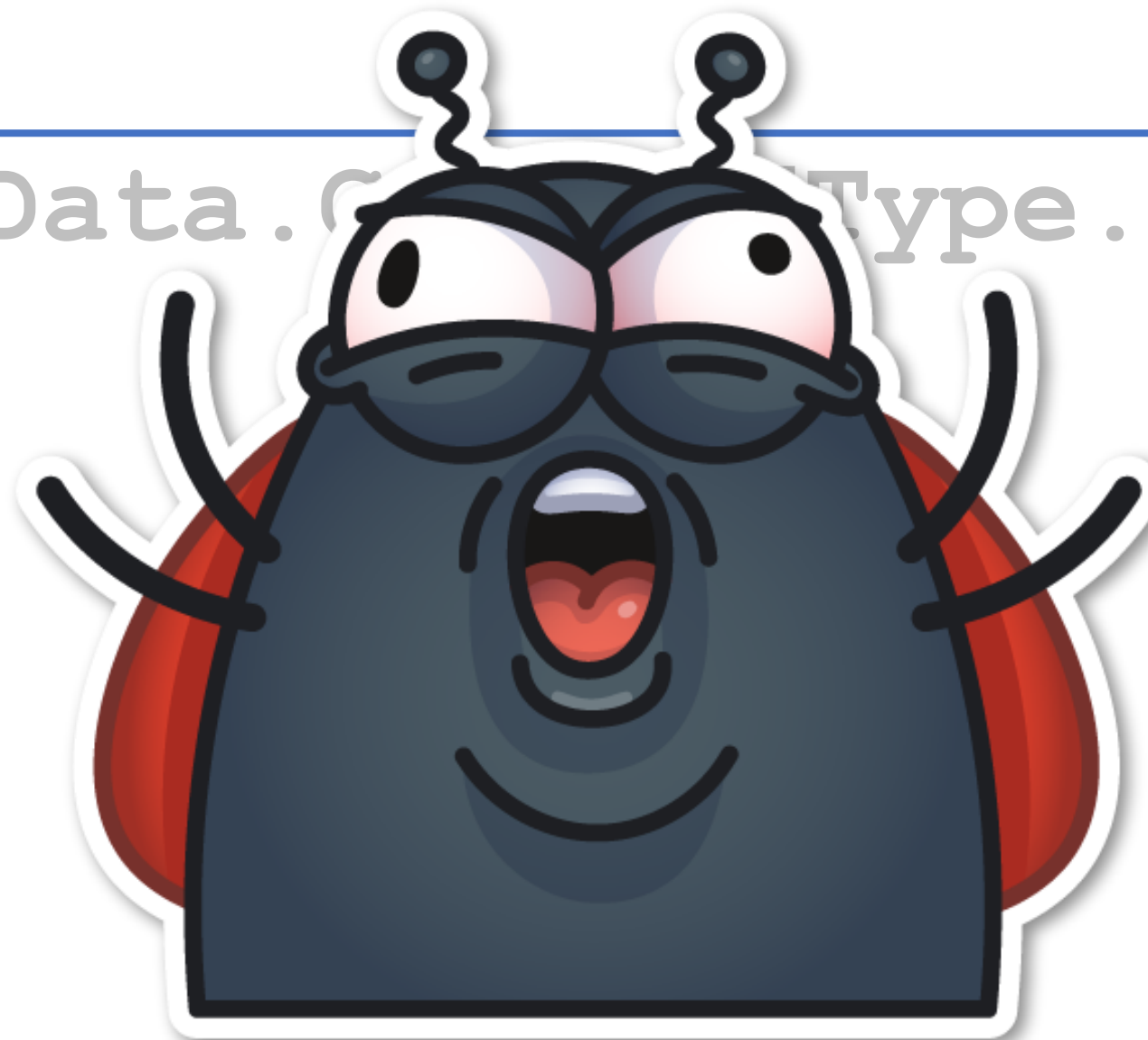
```
String userName = Request.Form["userName"];
```

```
SELECT * FROM Users WHERE UserName = ' ' OR '1'='1'
```

```
CommandType = System.Data.CommandType.Text } )
```

```
....
```

```
}
```



Пример уязвимости SQL injection

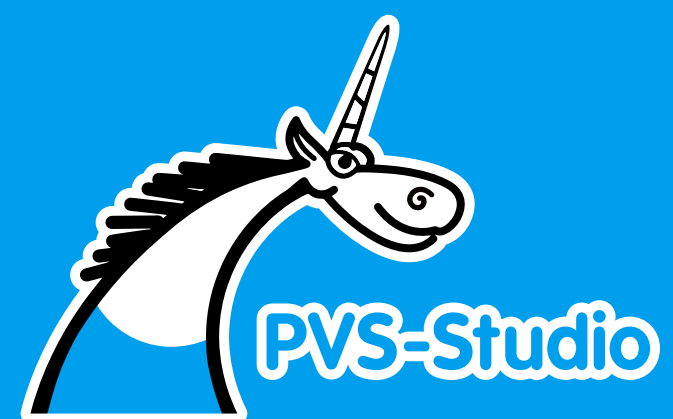
34

```
using (SqlConnection connection = new SqlConnection(...))
{
    ....
    String userName = Request.Form["userName"];
    using (var command = new SqlCommand() {
        Connection = connection,
        CommandText =
            "SELECT * FROM Users WHERE UserName = '" + userName + "'",
        CommandType = System.Data.CommandType.Text })
    ....
}
```

V5608 Possible SQL injection.

Potentially tainted data in the 'userName' variable is used to create SQL command.

Но для кого это?



РБПО



ГОСТ Р 56939-2024

Защита информации

РАЗРАБОТКА БЕЗОПАСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Общие требования

- Дата введения 2024-12-20.
- Взамен ГОСТ Р 56939-2016 *(о нем чуть позже)*



«Настоящий стандарт устанавливает общие требования к содержанию и порядку выполнения работ, связанных с созданием безопасного программного обеспечения (ПО) и устранением выявленных недостатков, в том числе уязвимостей, ПО.»



- Выявление ошибок в ПО;
- Снижение количества ошибок в ПО;
- Снижение ущерба от невыявленных уязвимостей ПО;
- Оперативное устранение выявляемых уязвимостей в ПО.

- Выявление ошибок в ПО;
- Снижение количества ошибок в ПО;
- Снижение ущерба от невыявленных уязвимостей ПО;
- Оперативное устранение выявляемых уязвимостей в ПО.

- Выявление ошибок в ПО;
- Снижение количества ошибок в ПО;
- Снижение ущерба от невыявленных уязвимостей ПО;
- Оперативное устранение выявляемых уязвимостей в ПО.

- Выявление ошибок в ПО;
- Снижение количества ошибок в ПО;
- Снижение ущерба от невыявленных уязвимостей ПО;
- Оперативное устранение выявляемых уязвимостей в ПО.

- Направлены на предотвращение и устранение уязвимостей программ
- Меры по разработке безопасного ПО:
 - ...
 - Динамический анализ кода
 - **Статический анализ кода**
 - ...



- Разработчик ПО должен проводить регулярный поиск уязвимостей на этапе эксплуатации жизненного цикла ПО.
- В состав применяемых инструментальных средств разработчик ПО должен включать статический анализатор.
- Рекомендуется настраивать конфигурацию статического анализатора и применять специализированные методы статического анализа для более глубокого поиска ошибок.

ГОСТ Р 71207-2024



Защита информации

РАЗРАБОТКА БЕЗОПАСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Статический анализ программного обеспечения

Общие требования

Введён в действие 01.04.2024

Введён впервые

«ГОСТ»

«ФСТЭК»

«Стандарты»



- ГОСТ написан понятно
- Полезная информация и рекомендации
- Не оторван от реальности!
Всё по делу

Из нового:

термины, требования,
критические ошибки и т.д



ГОСТ 71207-2024



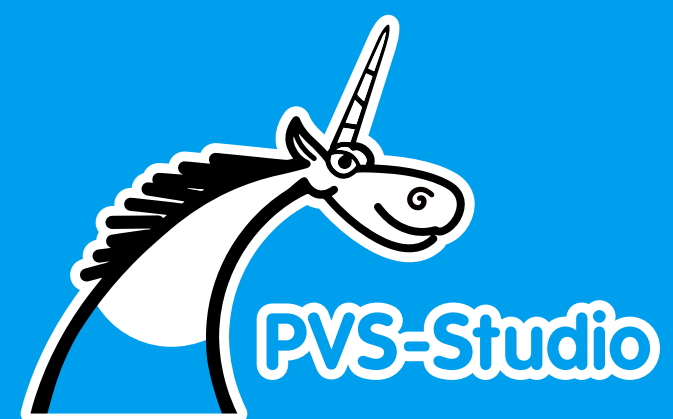
Про инструменты

- Линтеры
- Путаница
- Релизы 9 лет назад

Про использование

- Регулярно на сервере
- Отчеты не смотрят
- KPI \ Имена переменных

Технологии и методы анализа



Анализ потока данных [Data Flow]

51

- Определяет предположительное значение переменных и констант
- Примеры артефактов:
 - диапазон значений
 - точное значение
 - множество значений

```
int number = random.Next(1, 10)  
if (number == 10) ←
```



PVS-Studio

Анализ потока данных (RavenDB)

52

```
public override void VisitMethod(MethodExpression expr)
{
    if (    expr.Name.Value == "id"
        && expr.Arguments.Count == 0)
    {
        . . .
    }
}
```


Анализ потока данных (RavenDB)

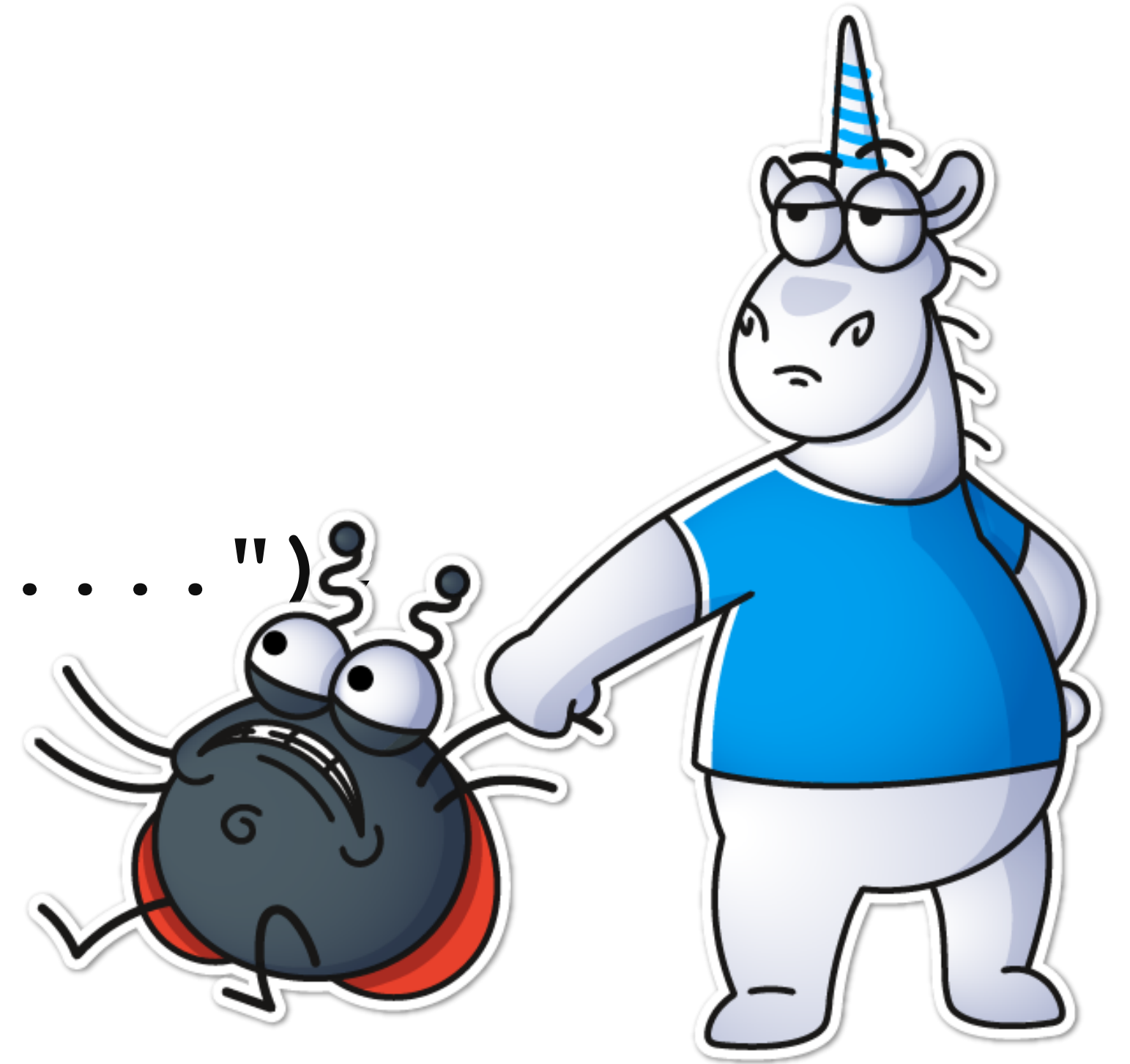
53

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

Анализ потока данных (RavenDB)

54

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

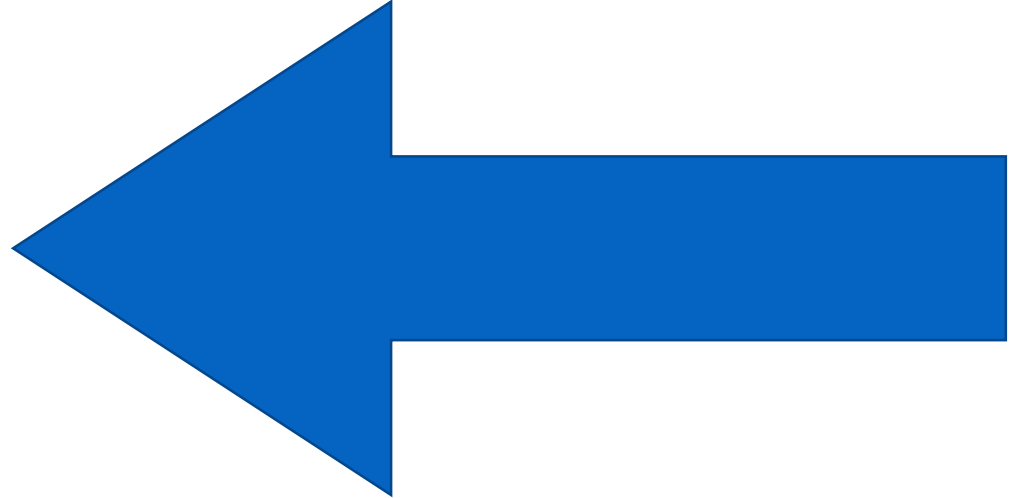


V3022 Expression 'expr.Arguments.Count != 1' is always true.

Межмодульный анализ (Barotrauma)

55

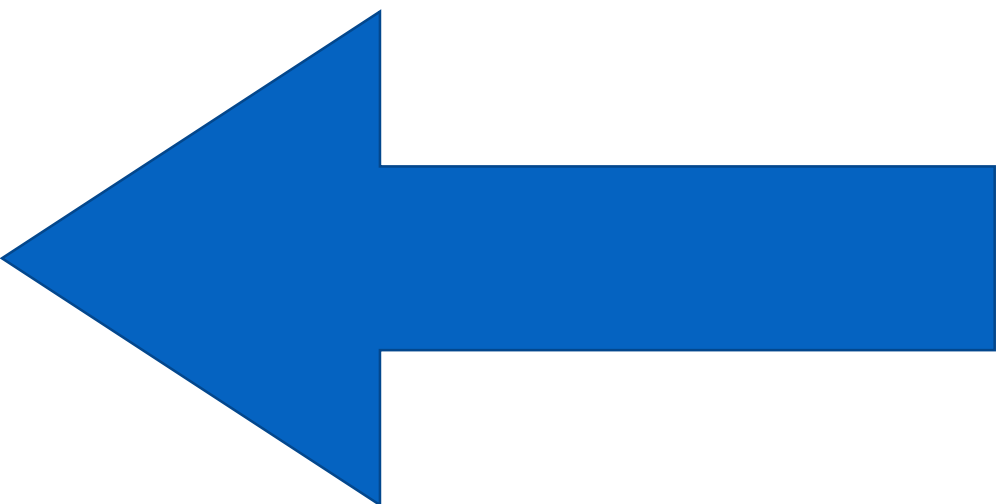
```
// Функция Remove() делает ссылку Sprite нулевой
partial class DecorativeSprite : ISerializableEntity
{
    public Sprite Sprite { get; private set; }
    . . .
    public void Remove()
    {
        Sprite?.Remove() ;
        Sprite = null;
        . . .
    }
}
```



Межмодульный анализ (Barotrauma)

56

```
// Функция Remove() делает ссылку Sprite нулевой
partial class DecorativeSprite : ISerializableEntity
{
    public Sprite Sprite { get; private set; }
    . . .
    public void Remove()
    {
        Sprite?.Remove();
        Sprite = null;
        . . .
    }
}
```



Межмодульный анализ (Barotrauma)

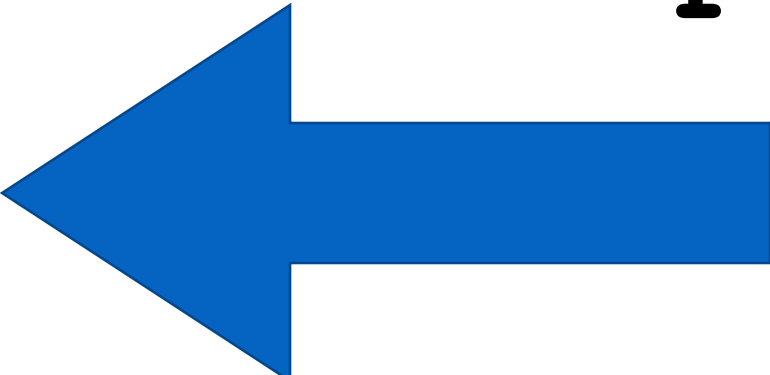
57

```
public void RecreateSprites()  
{  
    for (int i = 0; i < DecorativeSprites.Count; i++)  
    {  
        var decorativeSprite = DecorativeSprites[i];  
        decorativeSprite.Remove();  
        var source =  
            decorativeSprite.Sprite.SourceElement;  
        . . .  
    }  
}
```

Межмодульный анализ (Barotrauma)

58

```
public void RecreateSprites()  
{  
    for (int i = 0; i < DecorativeSprites.Count; i++)  
    {  
        var decorativeSprite = DecorativeSprites[i];  
        decorativeSprite.Remove();  
        var source =  
            decorativeSprite.Sprite.SourceElement;  
        . . .  
    }  
}
```



Межмодульный анализ (Barotrauma)

59

```
public void RecreateSprites()  
{  
    for (int i = 0; i < DecorativeSprites.Count; i++)  
    {  
        var decorativeSprite = DecorativeSprites[i];  
        decorativeSprite.Remove();  
        var source =  
            decorativeSprite.Sprite.SourceElement;  
        ....  
    }  
}
```



V3080. Possible null dereference. Consider inspecting 'decorativeSprite.Sprite'.

Критическая ошибка



Пример критической ошибки

61

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```

Пример критической ошибки

62

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```

Пример критической ошибки

63

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```

Пример критической ошибки

64

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```


Пример критической ошибки

65

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```

Пример критической ошибки

66

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```

Пример критической ошибки

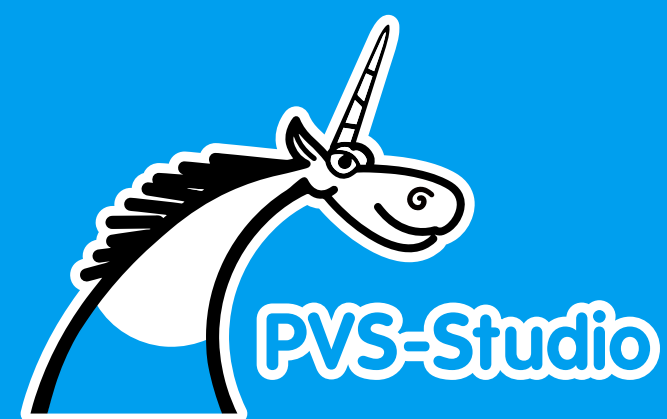
67

```
public void SetCredentials
(string userName, string password, string domain) {
    if (string.IsNullOrEmpty(userName)) {
        throw new ArgumentException
            ("Empty user name.", "userName");
    }
    if (string.IsNullOrEmpty("password")) {
        throw new ArgumentException
            ("Empty password.", "password");
    }
    CredentialsUserName = userName;
    CredentialsUserPassword = password;
    CredentialsDomain = domain;
}
```



V3022 Expression 'string.IsNullOrEmpty("password")' is always false.

Ну и много чего еще...



V772. Calling a 'delete' operator for a void pointer will cause undefined behavior.

Анализатор обнаружил потенциально возможную ошибку в коде, связанную с тем, что оператор 'delete' или 'delete[]' применяется для нетипизированного указателя (void*). Согласно стандарту C++20 (п. п. [§7.6.2.8/3](#)) такое применение ведет к неопределенному поведению.

Рассмотрим пример такого кода:

```
class Example
{
    int *buf;

public:
    Example(size_t n = 1024) { buf = new int[n]; }
    ~Example() { delete[] buf; }
};

....
void *ptr = new Example();
....
delete ptr;
....
```

Подобный пример опасен тем, что компилятор в реальности не знает, к какому типу относится указатель 'ptr'. Поэтому, при удалении такого нетипизированного указателя могут произойти различные неприятности, например, может возникнуть утечка памяти: оператор 'delete' не вызовет деструктор объекта типа 'Example', на который ссылается указатель 'ptr'.

Если подразумевалась именно работа с нетипизированным указателем, то перед применением оператора 'delete' ('delete[]') его необходимо привести к изначальному типу, например так:

```
....
void *ptr = new Example();
....
delete (Example*)ptr;
....
```

Иначе, во избежание ошибок, рекомендуется использовать только типизированные указатели совместно с оператором 'delete' ('delete[]'):

```
....
Example *ptr = new Example();
....
delete ptr;
....
```

Данная диагностика классифицируется как:

- CWE-758
- CERT-MSC15-C

Описание

Пример
ошибочного кода

Примеры
исправления

Сопоставление с SEI
CERT, OWASP, CWE

Для всего кода проводить анализ не реже, чем раз в 10 дней после внесения изменений.

Для нового кода проводить анализ сразу после внесения изменений.

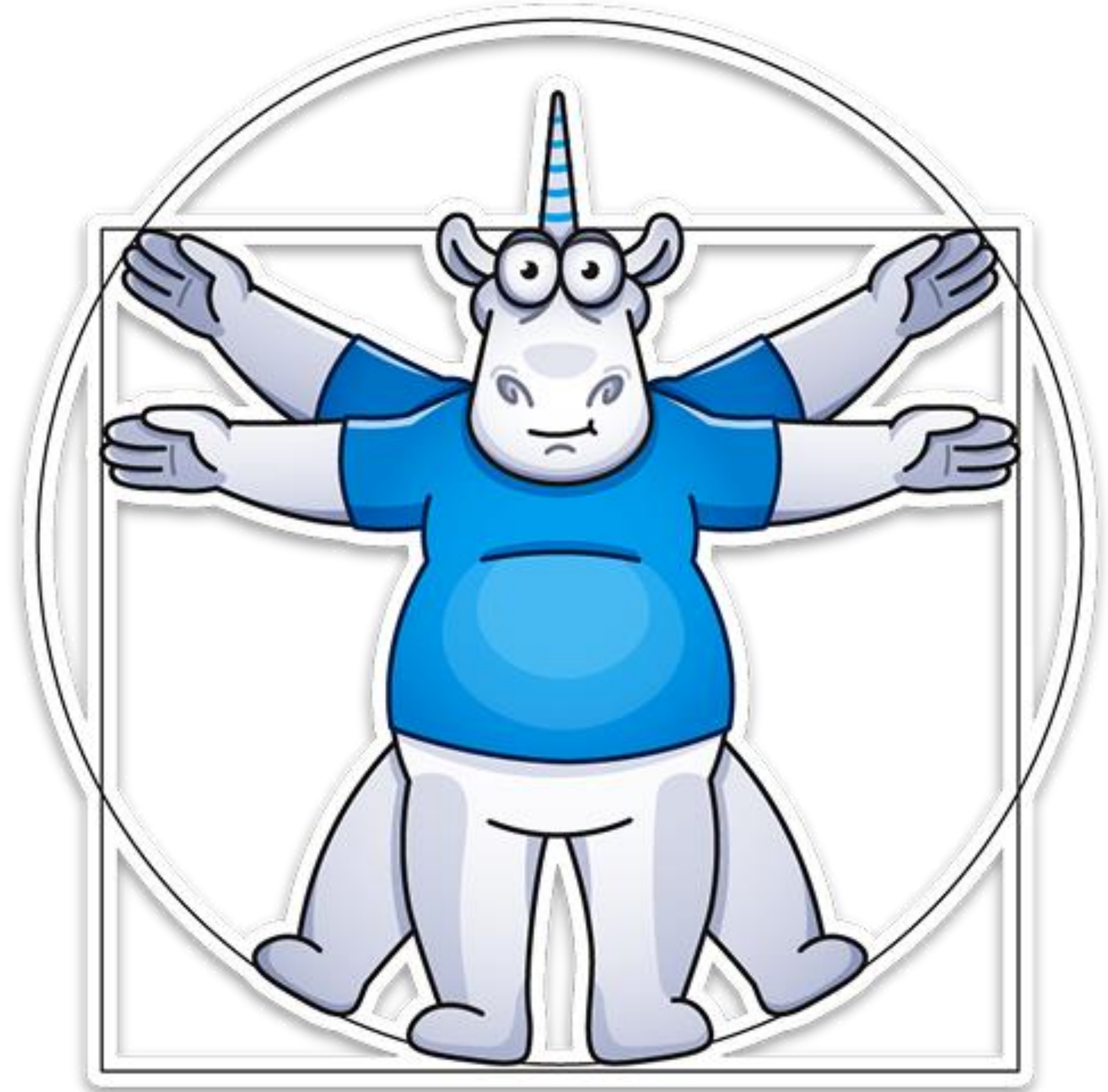
Просмотр и разметка предупреждений:

- Для нового кода не позднее, чем через 3 дня
- Для всего кода не позднее, чем через 10 дней



Для кого:

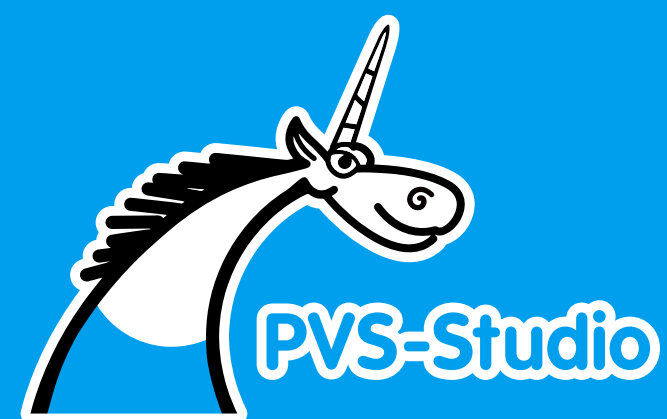
- Разработчикам надёжного ПО
 - Специалистам по внедрению
 - Руководителям и экспертам
-
- 30 вебинаров
 - Эксперты области



- Курс по ГОСТ Р 56939-2024
- Открытый
- Материал от инфобез компаний
- По каждому процессу ГОСТа



А что-то кроме ГОСТа?

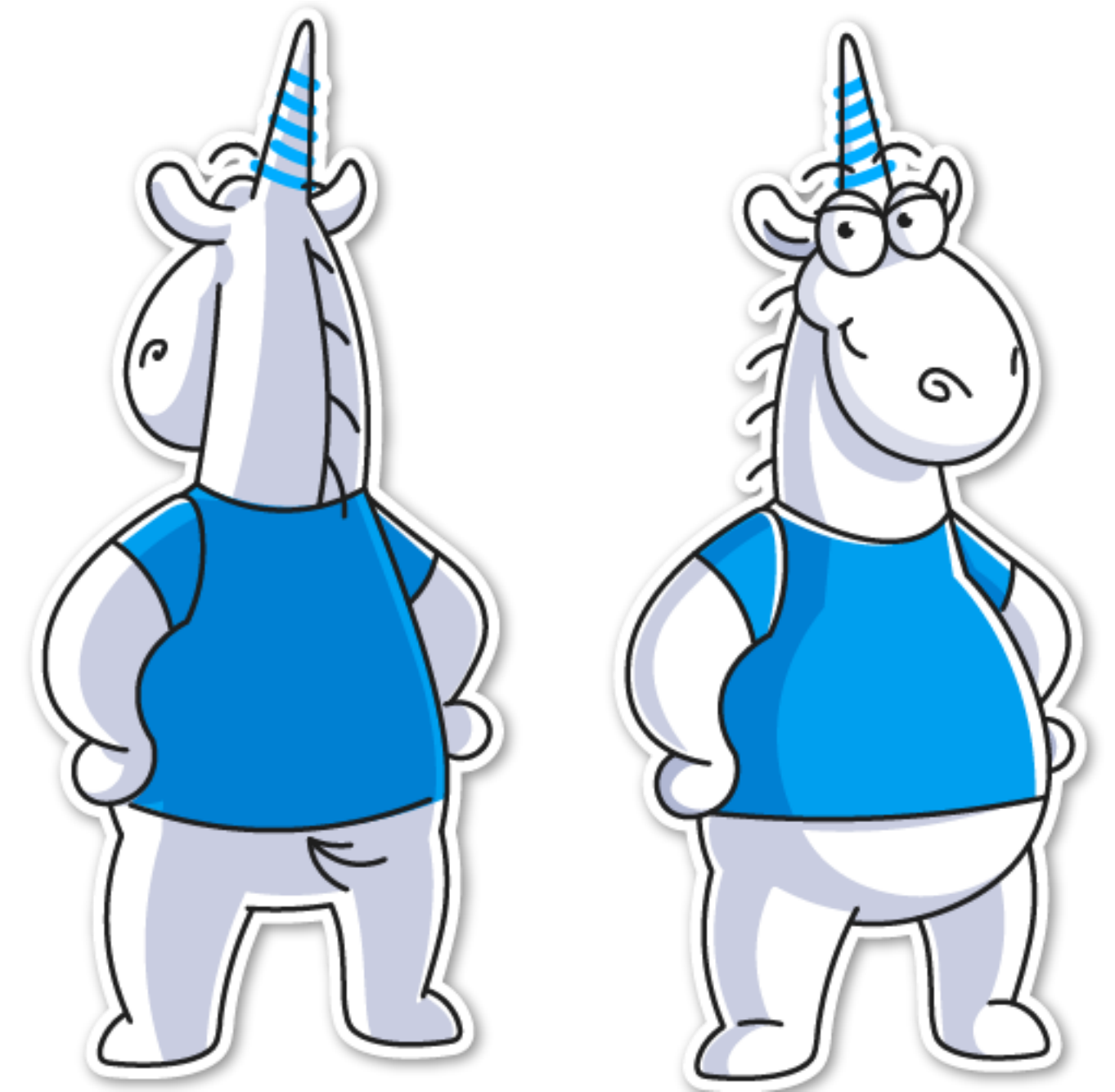


Safety (безопасность) / security (защищённость)

Безопасность

Надёжная работа приложения в любых условиях, без вмешательства извне

- MISRA C/C++
- AUTOSAR C++

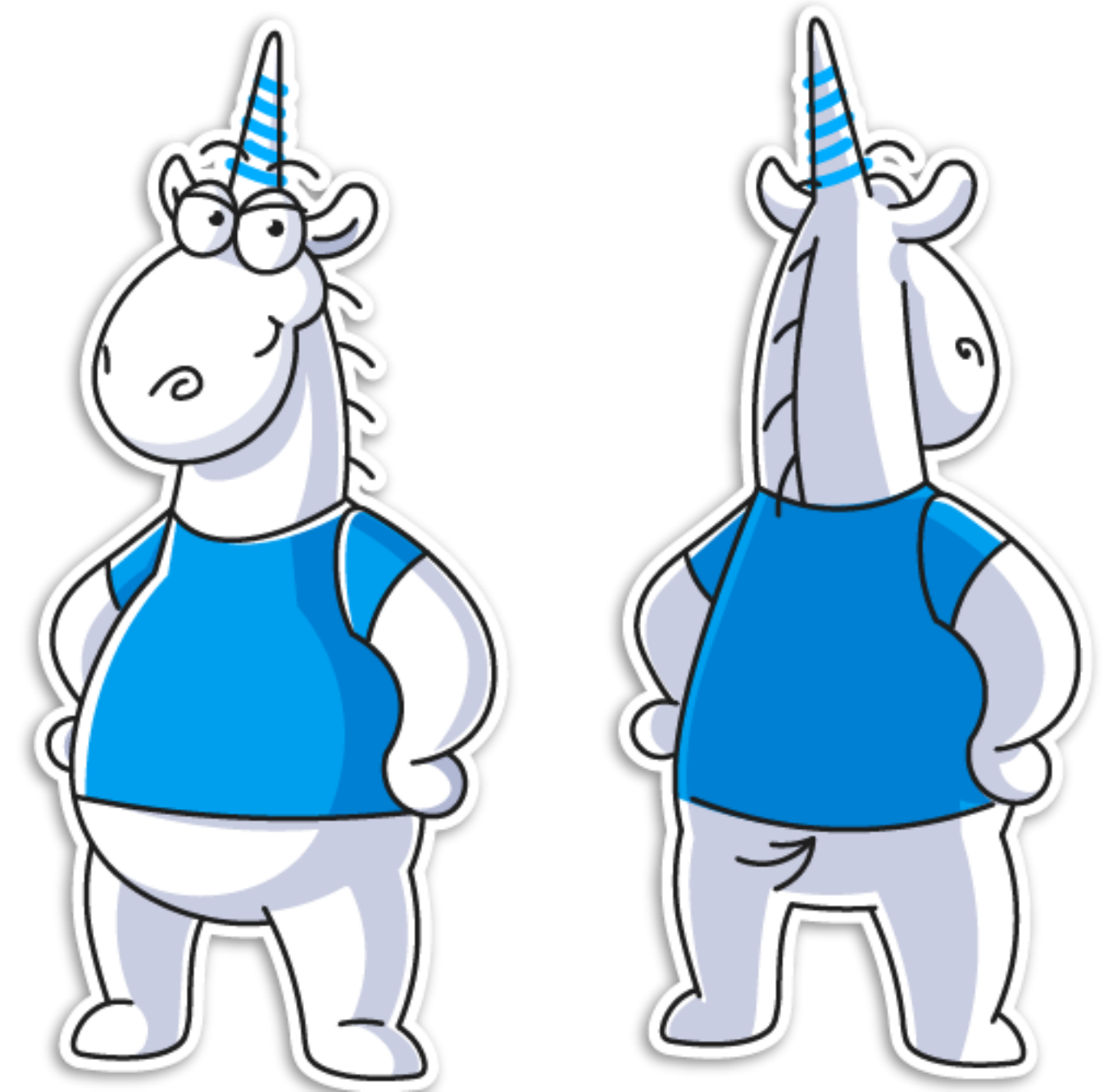


Safety (безопасность) / security (защищённость)

Защищённость

Стойкость ко внешним воздействиям, попыткам
вмешательства извне

- OWASP ASVS
- SEI CERT

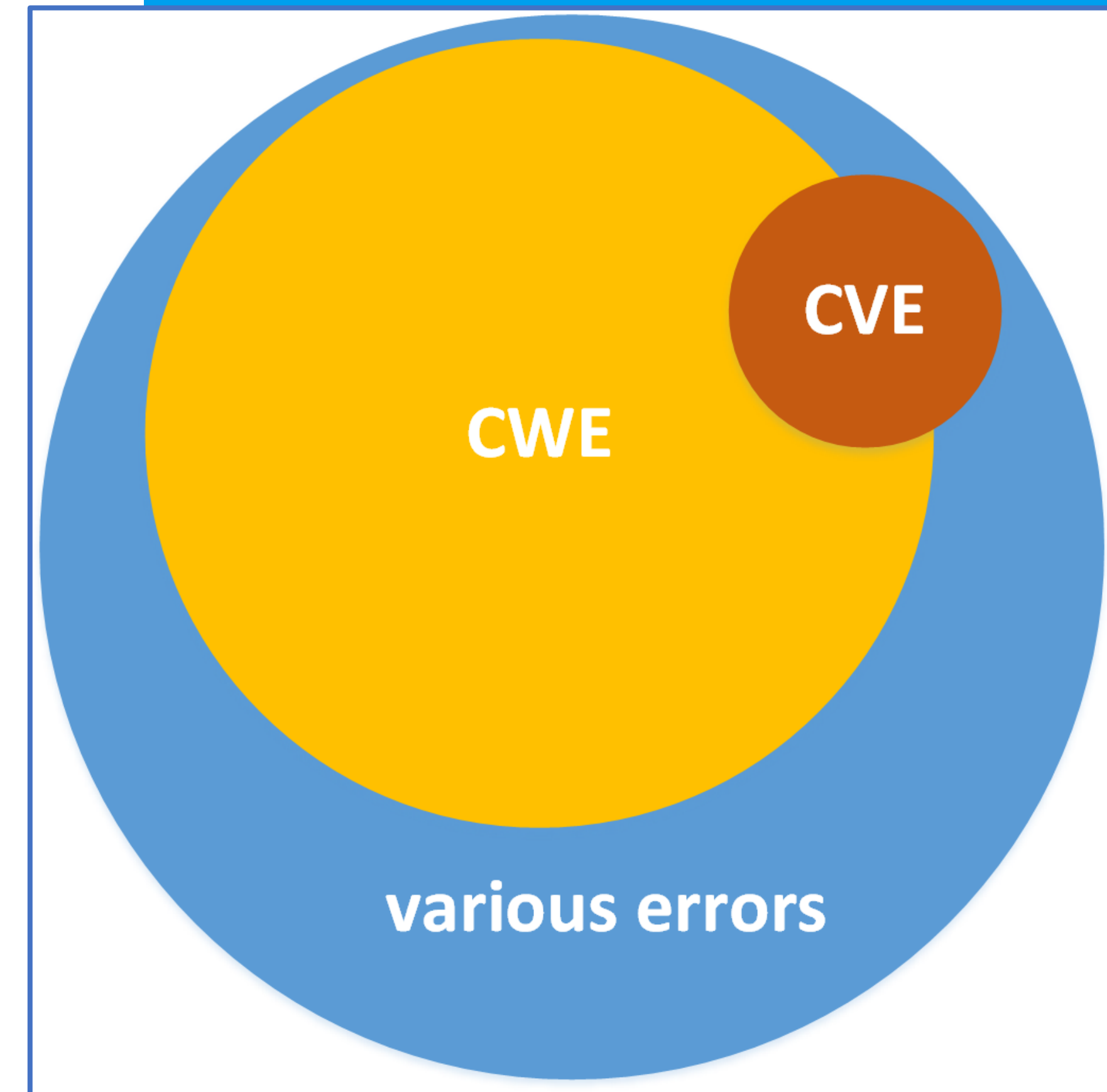


- **CWE** – Потенциальные уязвимости

Common Weakness Enumeration

- **CVE** – Существующие уязвимости

Common Vulnerabilities and Exposures

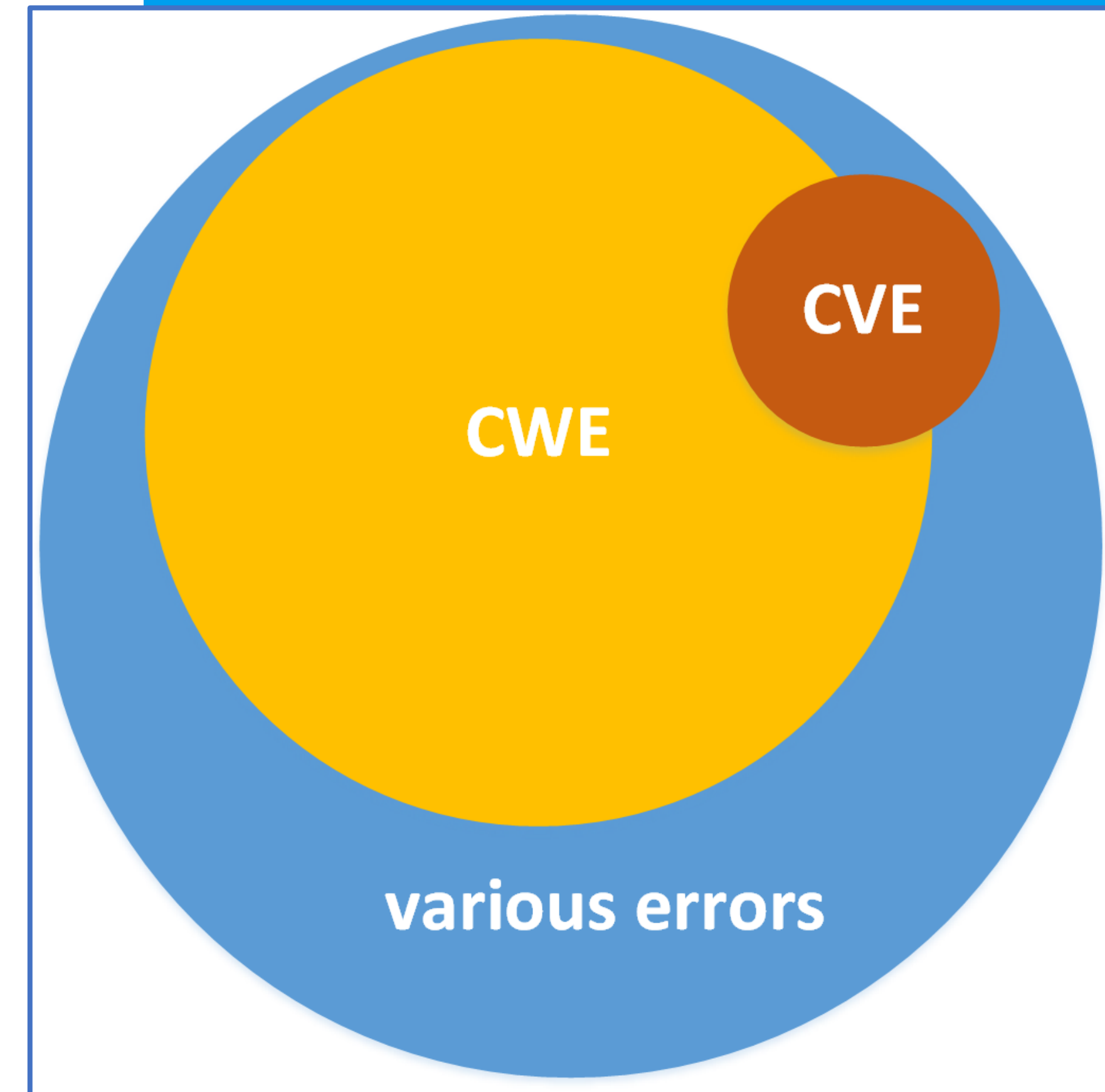


- **CWE** – Потенциальные уязвимости

Common Weakness Enumeration

- **CVE** – Существующие уязвимости

Common Vulnerabilities and Exposures

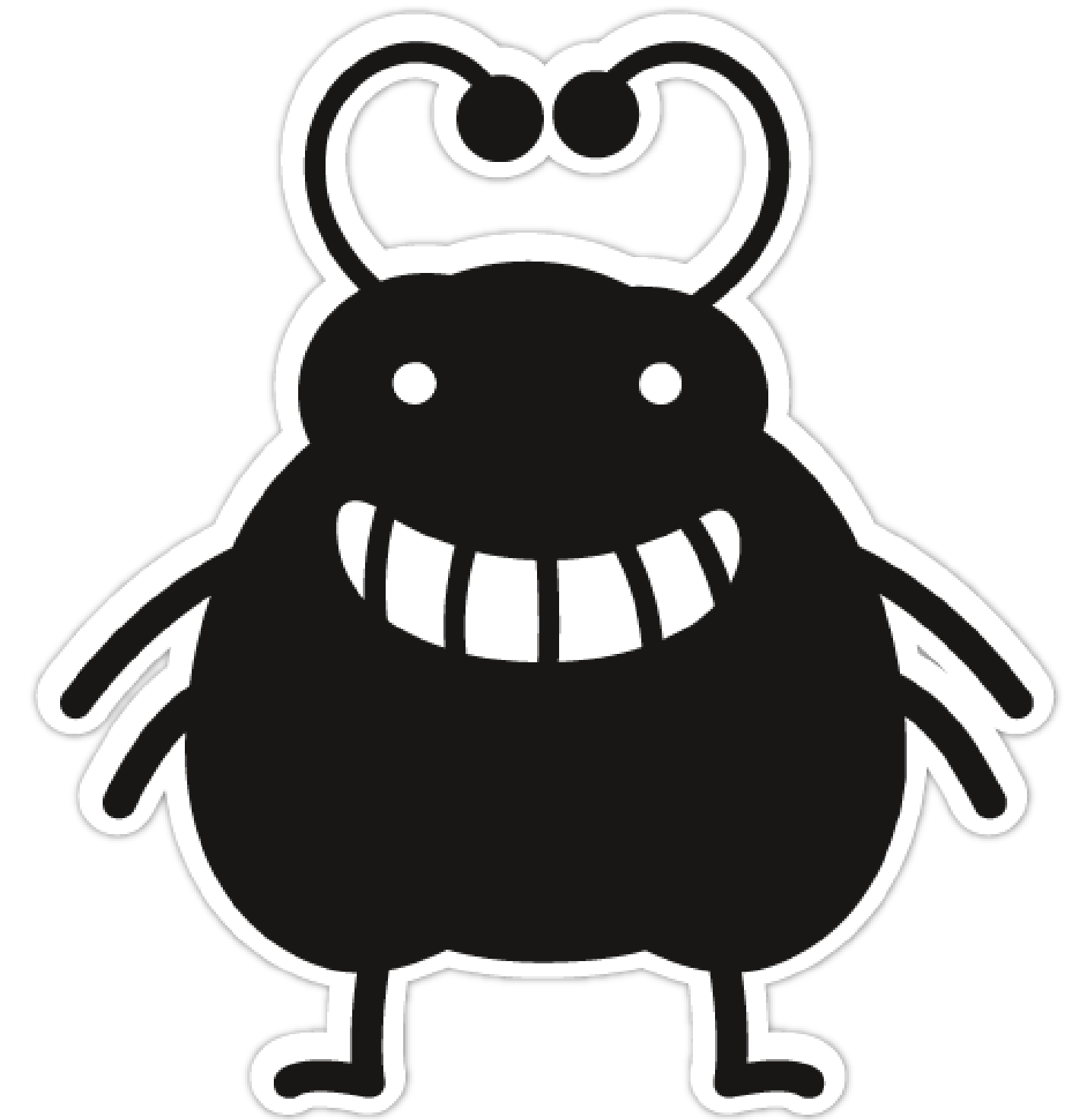


```
public struct BlobSasBuilder : IEquatable<BlobSasBuilder> {  
    ....  
    public bool Equals(BlobSasBuilder other) =>  
        BlobName == other.BlobName &&  
        CacheControl == other.CacheControl &&  
        BlobContainerName == other.BlobContainerName &&  
        ContentDisposition == other.ContentDisposition &&  
        ContentEncoding == other.ContentEncoding &&  
        ContentLanguage == other.ContentEncoding &&  
        ContentType == other.ContentType &&  
        ExpiryTime == other.ExpiryTime &&  
        Identifier == other.Identifier &&  
        IPRange == other.IPRange &&  
        Permissions == other.Permissions &&  
        Protocol == other.Protocol &&  
        StartTime == other.StartTime &&  
        Version == other.Version;  
}
```

```
public struct BlobSasBuilder : IEquatable<BlobSasBuilder> {  
    ....  
    public bool Equals(BlobSasBuilder other) =>  
        BlobName == other.BlobName &&  
        CacheControl == other.CacheControl &&  
        BlobContainerName == other.BlobContainerName &&  
        ContentDisposition == other.ContentDisposition &&  
        ContentEncoding == other.ContentEncoding &&  
        ContentLanguage == other.ContentEncoding &&  
        ContentType == other.ContentType &&  
        ExpiryTime == other.ExpiryTime &&  
        Identifier == other.Identifier &&  
        IPRange == other.IPRange &&  
        Permissions == other.Permissions &&  
        Protocol == other.Protocol &&  
        StartTime == other.StartTime &&  
        Version == other.Version;  
}
```



```
public struct BlobSasBuilder : IEquatable<BlobSasBuilder> {  
    ....  
    public bool Equals(BlobSasBuilder other) =>  
        BlobName == other.BlobName &&  
        CacheControl == other.CacheControl &&  
        BlobContainerName == other.BlobContainerName &&  
        ContentDisposition == other.ContentDisposition &&  
        ContentEncoding == other.ContentEncoding &&  
        ContentLanguage == other.ContentEncoding &&  
        ContentType == other.ContentType &&  
        ExpiryTime == other.ExpiryTime &&  
        Identifier == other.Identifier &&  
        IPRange == other.IPRange &&  
}
```



CWE-697: Incorrect Comparison

V3112 An abnormality within similar comparisons. It is possible that a typo is present inside the expression '`ContentLanguage == other.ContentEncoding`'


```
public struct FileSasBuilder : IEquatable<FileSasBuilder> {  
    ....  
    public bool Equals(FileSasBuilder other) =>  
        CacheControl == other.CacheControl  
        && ContentDisposition == other.ContentDisposition  
        && ContentEncoding == other.ContentEncoding  
        && ContentLanguage == other.ContentEncoding  
        && ContentType == other.ContentType  
        && ExpiryTime == other.ExpiryTime  
        && FilePath == other.FilePath  
        && Identifier == other.Identifier  
        && IPRange == other.IPRange  
        && Permissions == other.Permissions  
        && Protocol == other.Protocol  
        && ShareName == other.ShareName  
        && StartTime == other.StartTime  
        && Version == other.Version  
}
```

```
public struct FileSasBuilder : IEquatable<FileSasBuilder> {  
    ....  
    public bool Equals(FileSasBuilder other) =>  
        CacheControl == other.CacheControl  
        && ContentDisposition == other.ContentDisposition  
        && ContentEncoding == other.ContentEncoding  
        && ContentLanguage == other.ContentEncoding  
        && ContentType == other.ContentType  
        && ExpiryTime == other.ExpiryTime  
        && FilePath == other.FilePath  
        && Identifier == other.Identifier  
        && IPRange == other.IPRange  
        && Permissions == other.Permissions  
        && Protocol == other.Protocol  
        && ShareName == other.ShareName  
        && StartTime == other.StartTime  
        && Version == other.Version  
}
```



PVS-Studio

83

Статический анализатор кода (SAST)

- Языки C, C++, C#, Java
- Поддержка стандартов безопасности и защищенности
- Интегрируется в большинство популярных IDE и CI/CD
- B2B, 17 лет на рынке



Q/A

