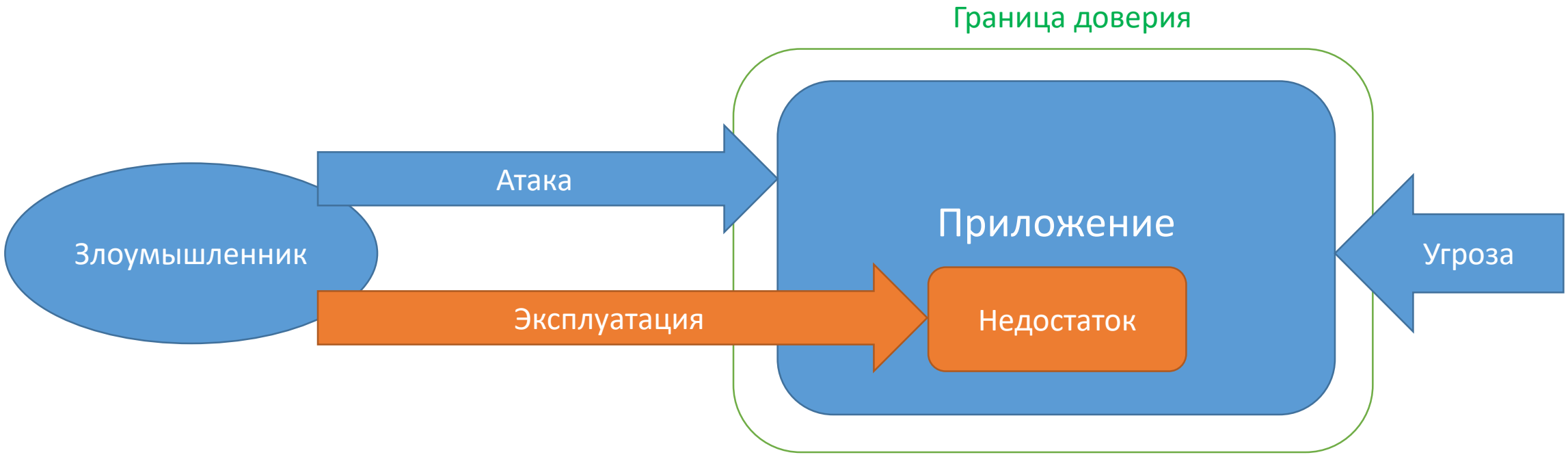
Инъекции

Самосадный Кирилл
https://t.me/kirsamosad
samosad@seclab.cs.msu.su

МГУ 2018

ЛикБез про ИнфоБез

Определения



Угроза: что атакующий хочет сделать с информацией или процессом
Атака: то, как он может это сделать
Эксплуатация: успешная атака
Граница доверия - изменение уровня доверия к данным

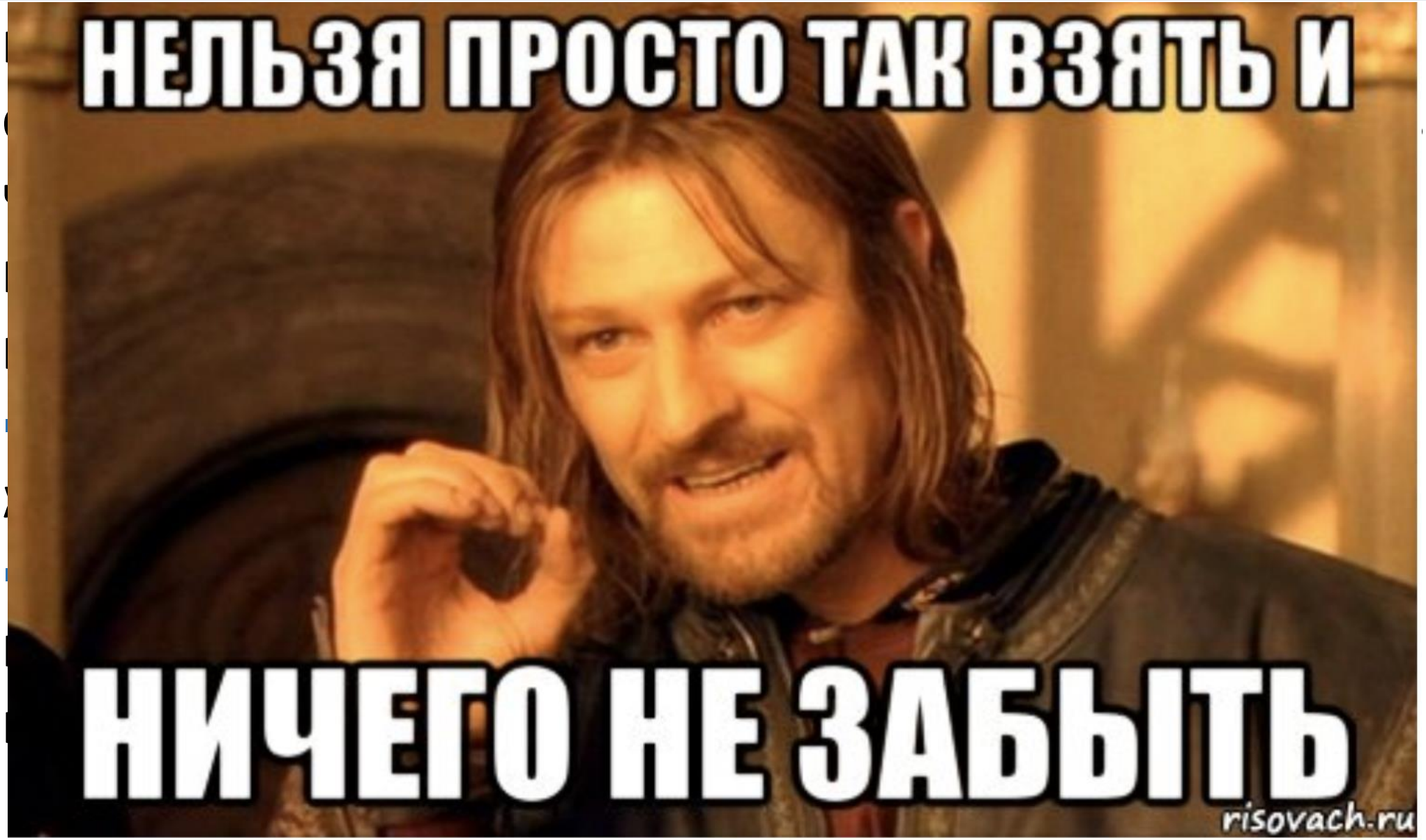
Вечная путаница

- Недостаток vs Уязвимость (недостаток плюс возможность эксплуатации)
- Безопасность (~~состояние защищенности~~ нет угроз) vs Защищенность (защищены от угроз): безопасная программа vs защищенная программа
- Авторизация vs Аутентификация
- Авторизация – проверка (предоставление) прав на действие
- Аутентификация – проверка подлинности
- Идентификация – распознавание по некоторому набору признаков (идентификаторов)

Недостатки в ПО

- Недостатки проектирования
- Недостатки реализации
- Недостатки внедрения (конфигурации)
- Недостатки эксплуатации
- На каждом этапе появляются свои типы недостатков
- На каждом этапе можно побороться за защищенность приложения соответствующими методами
  - Идея: на этапе X адресовать недостатки последующих этапов

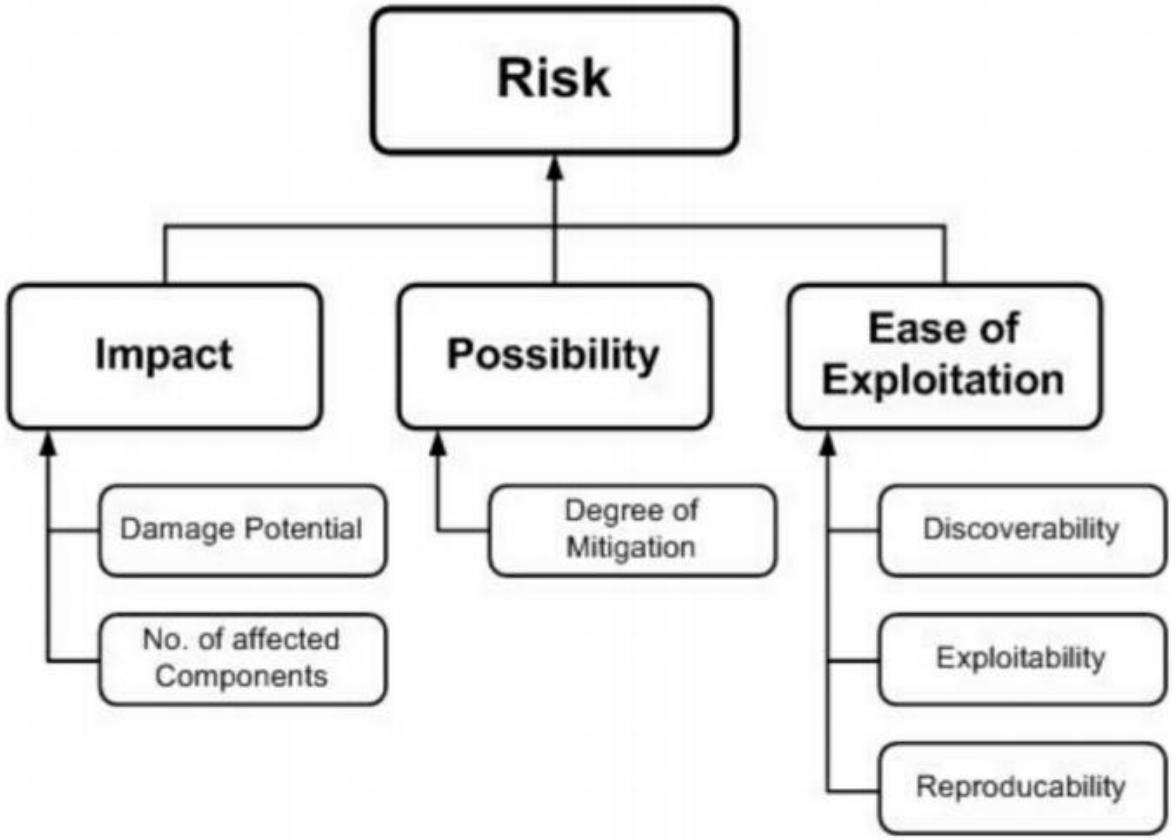
Причины появления недостатков



Методы обеспечения "состояния защищенности"

- Не допустить недостаток
  - Обучение
  - Контроль качества
- Скомпенсировать недостаток средствами защиты
- Типичная безопасность:
  - Мониторинг и реагирование (недостаток в коде – тоже инцидент)
  - Уменьшение поверхности атаки

Оценка рисков



OWASP Top 10 Web

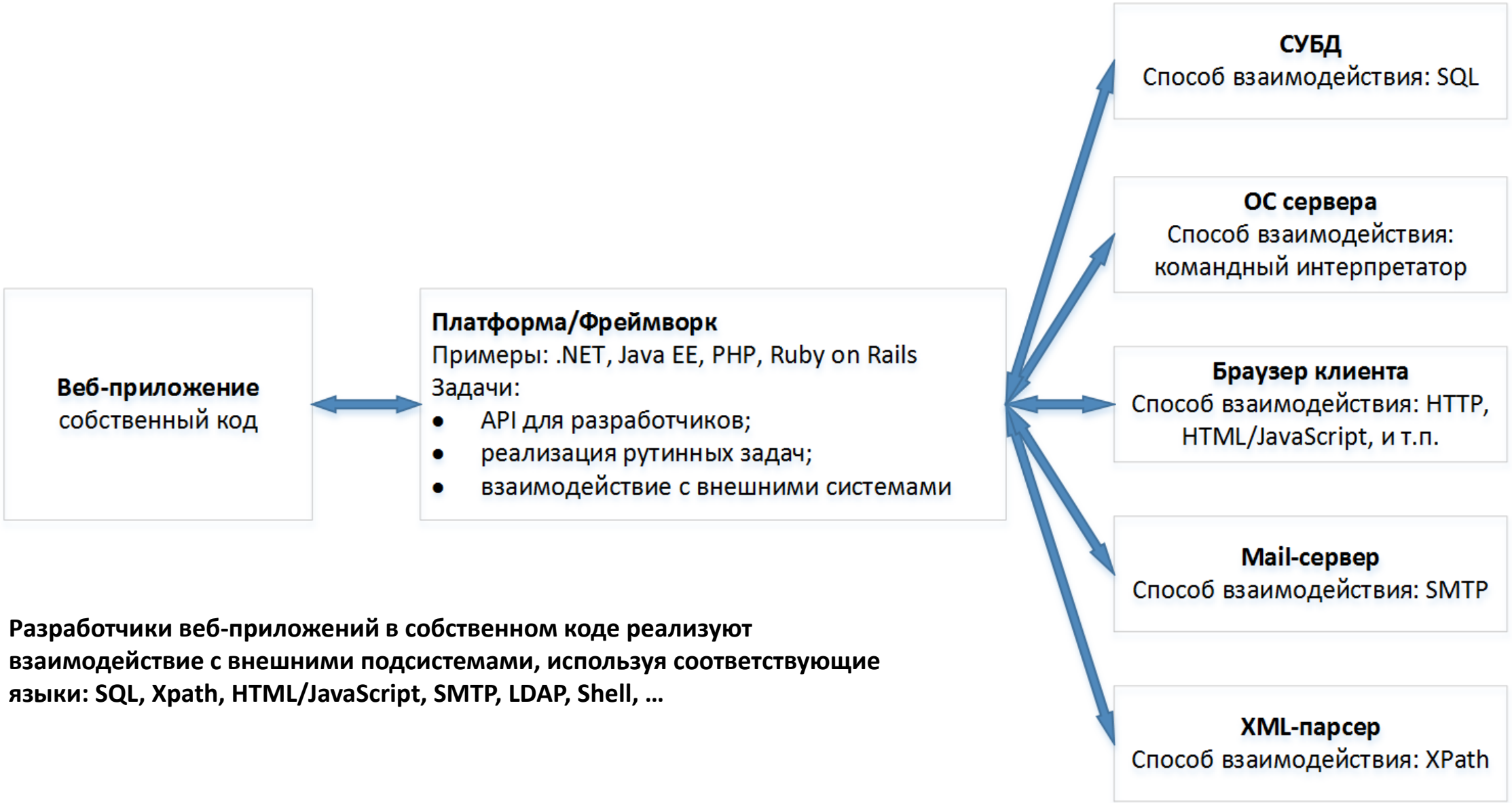
- A1:2017 – Injection
- A2:2017 – Broken Authentication
- A3:2017 – Sensitive Data Exposure
- A4:2017 – XML External Entities (XXE)
- A5:2017 – Broken Access Control
- A6:2017 – Security Misconfiguration
- A7:2017 – Cross-Site Scripting (XSS)
- A8:2017 – Insecure Deserialization
- A9:2017 – Using Components with Known Vulnerabilities
- A10:2017 – Insufficient Logging & Monitoring

OWASP Top 10 Mobile

- M1:2016 – Improper Platform Usage
- M2:2016 – Insecure Data Storage
- M3:2016 – Insecure Communication
- M4:2016 – Insecure Authentication
- M5:2016 – Insufficient Cryptography
- M6:2016 – Insecure Authorization
- M7:2016 – Poor Code Quality
- M8:2016 – Code Tampering
- M9:2016 – Reverse Engineering
- M10:2016 – Extraneous Functionality

Инъекции

Устройство типичного веб-приложения



Разработчики веб-приложений в собственном коде реализуют взаимодействие с внешними подсистемами, используя соответствующие языки: SQL, Xpath, HTML/JavaScript, SMTP, LDAP, Shell, …

Downstream Components

- Приложения взаимодействует с различными внешними подсистемами
  - СУБД, LDAP, стандартный интерпретатор ОС, почтовый демон, браузер, файловая система и т.п.
- У большинства подсистем есть свой язык
  - SQL у СУБД, Shell/CMD у стандартного интерпретатора ОС, HTML и Javascript у браузера и т.п.
- Обращения приложения во внешние подсистемы параметризуются
- Есть константная часть, есть переменная
- Переменная часть формируется динамически с участием данных от пользователя

Суть инъекции

- В языке есть ключевые слова, разделители, правила оформления секций данных:
  - SELECT * FROM news WHERE author = '**admin**'
  - View first
  - grep -R "**search string**" *
- Injection-атаки – выход из контекста данных в контекст команд:
  - SELECT * FROM news WHERE author = '**admin' or sleep(5) --** '
  - <a href="/show?id="**1"><script>alert(1)</script>View first
  - grep -R "**search string" 1; echo "p0wned" #**" *

Примеры Injection-контекстов

- SQL-операторы
- HTML-разметка
- Javascript-операторы
- CSS-директивы
- HTTP-заголовки
- XPath-операторы
- LDAP-операторы
- Shell/cmd-оператор
- Конструкции языка приложения (Python/PHP)
- SMTP-команды

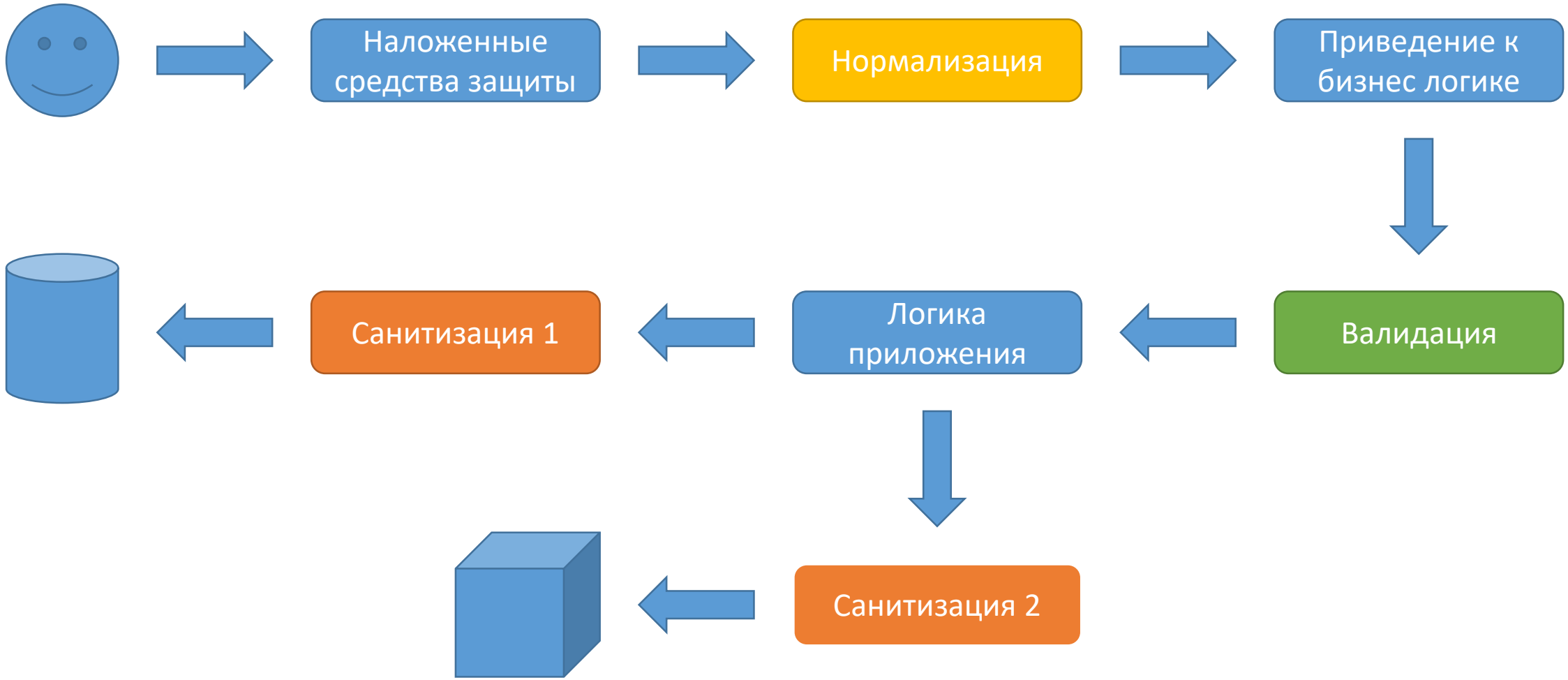
Модель невмешательства

- Пусть программа получает данные из high-level(доверенных) и low-level (недоверенных) каналов
- Программа генерирует вывод в high-level(критические) и low-level (обычные) каналы
- Вывод low-level (недоверенных) данных в high-level (критический) канал считается нарушением принципа невмешательства

Общая методика предотвращения Injection-атак

- Валидируем все входные данные
  - если есть тип – приводим к нему
  - если строка – регулярное выражение (черным спискам отказать)
- Валидация входных данных относительно белого списка не всегда может гарантировать безопасность
  - входные данные после валидации могут содержать служебные символы и разделители языка служебного компонента
  - => необходимо осуществлять экранирование или кодирование в зависимости **от контекста вывода**
  - например, O'Neal в MySQL-операторе должен стать O\'Neal
  - (a + b) < c в HTML должно быть (a + b) < c (**в теле тега**)

Схема валидации и санитизации



DEMO: Command Injection

SQL & NoSQL

Простой пример

```php
<?php
    $sql = "SELECT id, firstname, lastname FROM users WHERE id=$_GET['id']
        or username=:username"
    $sth = $dbh->prepare($sql)
    $sth->execute(array(':username', $_GET["username"]))
    $result = $sth->fetchAll();
?>
```

- NoSQL – non SQL, не реляционная СУБД
- У них тоже есть свои языки запросов

```
db.inventory.find( {
                    status: "A",
                    $or: [ { qty: { $lt: 30 } }, { item: /^p/ } ]
        }, {
                     {"item": 1, "status": 1}
        })
```

SELECT _id, item, status **FROM** inventory **WHERE** status = "A" **AND** (qty < 30 **OR** item **LIKE** "p%")

Литература

- Dorothy E. Denning. Lattice Model of Secure Information Flow, 1976
  - http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.5776&rep=rep1&type=pdf
- Michal Zalewski. The Tangled Web: A Guide to Securing Modern Web Applications Book
  - https://repo.zenk-security.com/Techniques%20d.attaques%20%20.%20%20Failles/The%20Tagled%20Web%20A%20Guide%20to%20Securing%20Modern%20Web%20Applications.pdf
- XSS without HTML: Client-Side Template Injection with AngularJS
  - https://portswigger.net/blog/xss-without-html-client-side-template-injection-with-angularjs
- OWASP XSS (Cross Site Scripting) Prevention Cheat Sheet
  - https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet
- HackerOne
  - Лекции по "взлому" – https://www.hacker101.com/
  - Список топовых опубликованных недостатков – https://hackerone.com/hacktivity?sort_type=popular&filter=type:all&page=1&range=forever