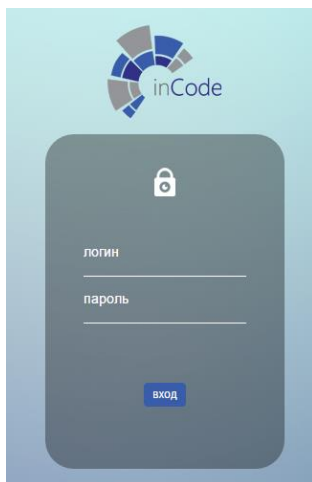


## Как работать с Solar inCode: пошаговая инструкция

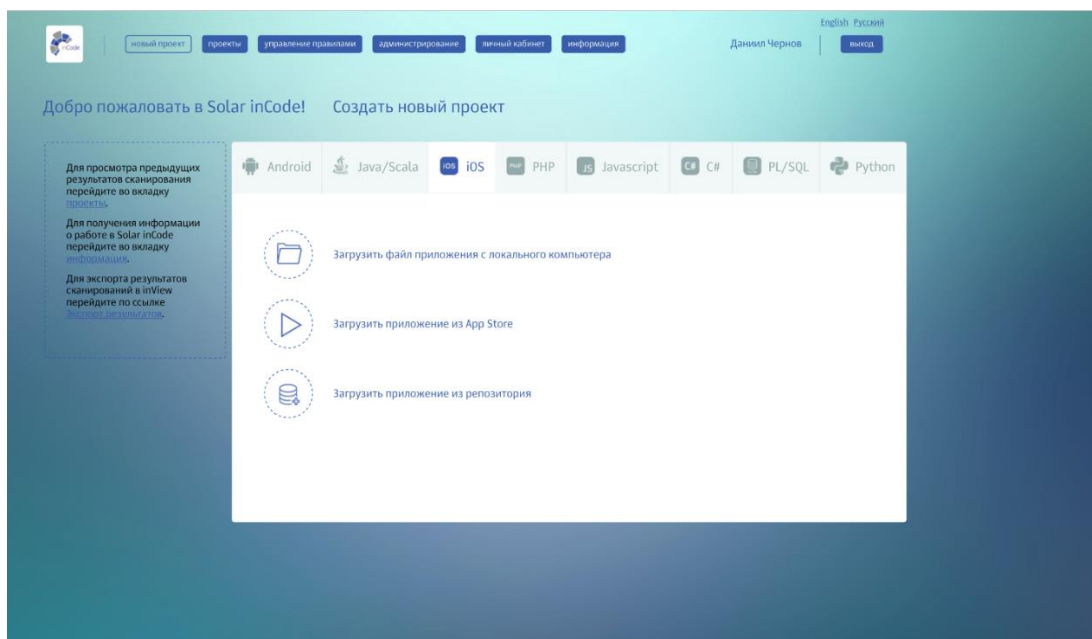
### Шаг 1.

Авторизуйтесь в веб-интерфейсе (<https://incode-demo.solarsecurity.ru/incode/>), используя учётную запись, полученную от менеджера по электронной почте.

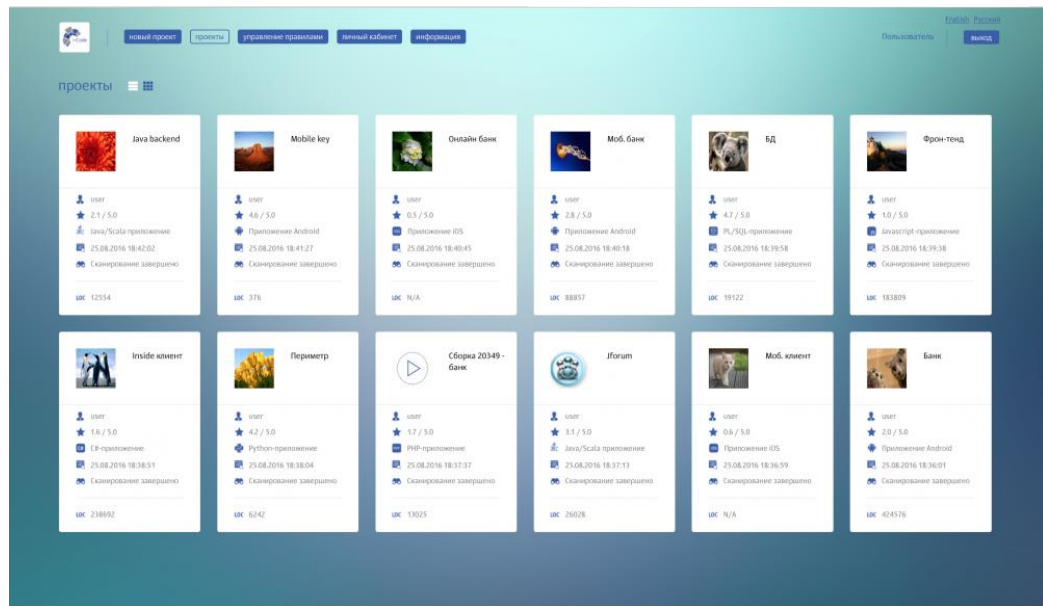


### Шаг 2.

Выберите способ загрузки проекта и начните сканирование.



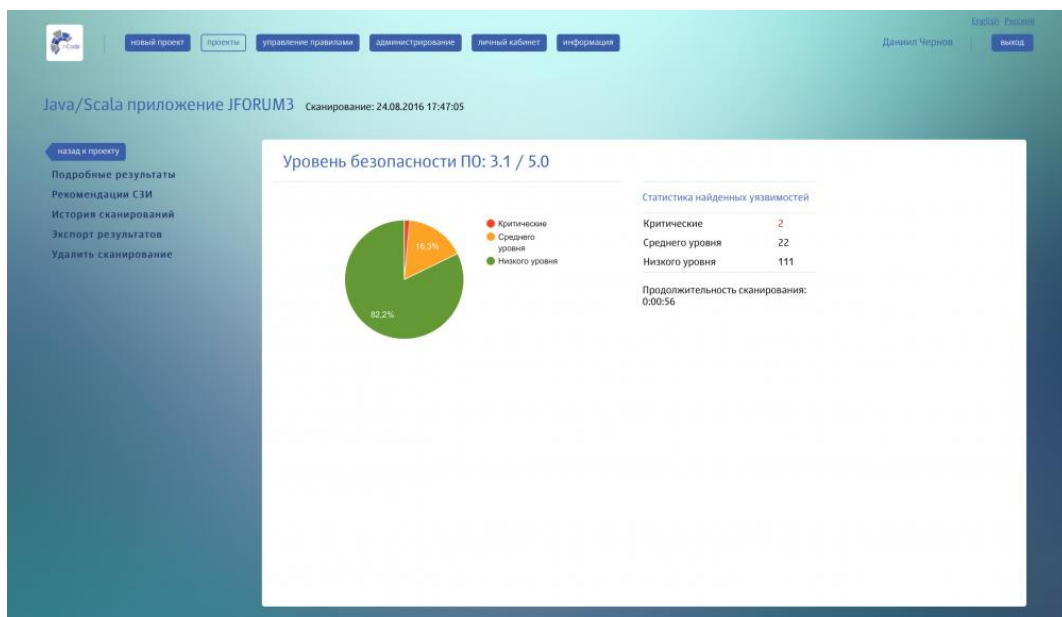
Вы можете выбрать удобный способ сортировки проектов, например, плиткой. Здесь вы сразу увидите основную информацию по проекту: название, рейтинг, вид приложения и т.д.



Вид сортировки можно поменять в любое время.

### Шаг 3.

После сканирования приложения вы сможете увидеть сводную диаграмму, в которой будет указано количество уязвимостей и их критичность.



После завершения сканирования вы сможете не только увидеть количество уязвимостей в коде приложения и оценку их критичности, но и рассмотреть каждую уязвимость более подробно и получить развёрнутые рекомендации по её устранению с указанием конкретной строки кода и описанием обнаруженной проблемы.

**Уязвимости**

Уязвимость	Количество
Критические	2
Среднего уровня	22
Низкого уровня	111
Удаленные	0

**Файлы**

- Слабый алгоритм хеширования: 1
- net/forum/uttl/MDS.java:39
- Атака по времени при сравнении строк: 1
- Куки с слишком общим параметром path: 1
- Куки с неограниченным сроком действия: 1
- Использование незащищенного протокола HTTP: 1
- Подмена пути: 1
- Внедрение внешней сущности в XML: 1
- Подмена пути: 14
- Совместное хранение доверенных и недоверенных данных: 3

**JFORUM3**

```

net/forum/uttl/MDS.java:39
38.     try {
39.         MessageDigest md = MessageDigest.getInstance("MD5");
40.         md.update(str.getBytes());
41.         byte[] hash = md.digest();
42.
43.         for (byte element : hash) {
44.             if ((0xFF & element) < 0x10) {
45.                 hexString.append('0').append(Integer.toHexString(0xFF & element));
46.             }
47.             else {
48.                 hexString.append(Integer.toHexString(0xFF & element));
49.             }
50.         }
51.     }
52. }
    
```

**Описание уязвимости**

Используемая хеш-функция небезопасна. Ее использование может привести к утрате конфиденциальности данных.

Хеш-функции MD2, MD5, SHA-1 обладают известными уязвимостями. Нахождение коллизий для функций MD2 и MD5 не требует существенных ресурсов: аналогичная задача для SHA-1, вероятно, будет решена в ближайшем будущем. Если эти функции применяются для хранения ценной информации (например, паролей), ее конфиденциальность может быть нарушена.

Хеш-функция, применяемая для хранения паролей, кроме устойчивости к коллизиям, должна быть не слишком быстрой. Это осложняет атаку путем полного перебора. Для этой цели разработаны специализированные хеш-функции: PBKDF2, bcrypt, scrypt.

Пусть пароли пользователей хранятся на сервере в зашифрованном виде с использованием небезопасной хеш-функции (например, MD5). Возможный сценарий атаки:

- Злоумышленник получает доступ к базе зашифрованных паролей.
- Злоумышленник, используя уязвимость алгоритма хеширования, вычисляет строку, для которой алгоритм хеширования даёт то же значение, что и для **каждого** пользователя.

Solar InCode полезен как при разработке новых приложений, так и при доработке уже существующих. С помощью сканера можно отслеживать количество уязвимостей в процессе разработки приложения и своевременно их устранять. Статистика Solar InCode позволяет отслеживать динамику изменений.

**информация**

- Подробнее результаты последнего сканирования
- Рекомендации СЗИ по последнему сканированию
- Экспорт результатов
- История сканирований
- Настройки проекта
- Удалить проект

**Java/Scala приложение JFORUM3**

Сканирование завершено

История сканирований

Дата	Уровень безопасности
24.08.2016 17:47:05	3.1
24.08.2016 16:24:21	3.6

**График уровня безопасности ПО**

Сканирование	Уровень безопасности
24.08.2016 16:24:21	3.6
24.08.2016 17:47:05	3.1

**График количества уязвимостей**

Сканирование	Критическое	Среднего уровня	Низкого уровня	Всего
24.08.2016 16:24:21	2	22	111	135
24.08.2016 17:47:05	2	22	111	135