VK Cloud

Бесплатные инструменты ИБ для разработчика. Есть ли польза?

Станислав Погоржельский,
Технологический евангелист
VK Cloud

tech

О чём поговорим?

127 уязвимостей за 30 минут

Компания не проверяла код годами.

DevOps запустил SonarQube Community.
Результат: 127 уязвимостей, 15 критических.

Исправить до релиза — $200.
Исправить на проде — $10 000+.

Соотношение: 1:50. Бесплатное ≠ бесполезное.

SonarQube Community Edition — что это?

это бесплатный инструмент автоматического анализа кода с открытым исходным кодом, который интегрируется в рабочий процесс разработки для проверки качества и безопасности кода

Основные возможности

1. Поиск проблем в коде:Баги (bugs) — ошибки, которые могут сломать приложениеУязвимости безопасности (vulnerabilities) — SQL-инъекции, XSS, CSRFCode smells — плохие практики, усложняющие поддержкуДублирование кода
2. Метрики качества:Технический долг — сколько времени нужно на исправление всех проблемПокрытие тестами (code coverage)Сложность кода (cyclomatic complexity)
3. Интеграция с инструментами разработки:Готовая интеграция с популярными CI/CD платформами, такими как Jenkins, TeamCity и CircleCI, а также с инструментами сборки Gradle и Maven SonarQube Free & Open Source Community Build | Sonar
4. Quality Gates (Ворота качества):Позволяет предотвратить попадание в production кода, который не соответствует установленным политикам качества

Что сегодня разберём

какие бесплатные инструменты реально работают

как их внедрить за 1 день

где бесплатное не спасёт

как понять, когда пора платить

Экосистема инструментов ИБ

Категория	Что проверяет	Примеры бесплатных
SAST	Исходный код	SonarQube Community, Semgrep
DAST	Запущенное приложение	OWASP ZAP, Nikto
SCA	Зависимости	Snyk Free, OWASP Dependency-Check
Secret Scanning	Утечки ключей	GitGuardian, Gitleaks
IaC Security	Terraform/K8s конфиги	Trivy

SAST = анализ кода без запуска приложения

Находит:
- SQL Injection
- XSS
- Path Traversal
- Hardcoded secrets
- Небезопасную криптографию

Где применять:
Pre-commit → Pull Request → Nightly scan → IDE

Инструменты:
Semgrep | SonarQube Community | Bandit

Покрытие: До 70 % уязвимостей на этапе разработки

SCA — Software Composition Analysis

SCA = анализ зависимостей

Проверяет:
- package.json / pom.xml / requirements.txt
- Сопоставление с базами CVE
- Устаревшие версии
- Лицензии

Зачем нужно:
80 % кода — чужие библиотеки
Уязвимость в одной = уязвимость во всём приложении

Инструменты:
Snyk Free | OWASP Dependency-Check | Trivy

Кейс Log4Shell:
1 библиотека → миллионы взломов (2021)

Кейс стартапа и не только

Исходные данные:

- Команда — 5 разработчиков
- Бюджет на ИБ — 0 ₽
- Стек — React + Node.js + PostgreSQL

Что сделали:

1. Зарегистрировались в Snyk Free
2. Подключили GitHub-репозиторий
3. Запустили сканирование — 43 уязвимости
4. 15 critical/high исправили за неделю

Где бесплатное не тянет

Когда начинаются проблемы:

1. Большие команды (> 10 человек)
   → нет ролей, приоритизации, отчётов
2. Комплаенс (PCI DSS, ISO 27001, 152-ФЗ)
   → нет аудит-логов и SLA
3. Бизнес-контекст и риски
   → инструмент не понимает, что важно для вас
4. Поддержка и скорость реакции
   → форумы вместо техподдержки

Если (стоимость инцидента × вероятность) > (цена подписки × 10) — пора платить.

Минимальный DevSecOps pipeline

CI/CD с проверками безопасности

```
# Пример GitHub Actions
jobs:
  build:
    steps:
      - run: semgrep --config=auto
      - run: trivy fs .
      - run: gitleaks detect
      - run: zap-baseline.py -t https://stage.example.com
```

Security by automation, не по вдохновению.

Этапы:
Build → Test → Security Scan → Deploy

Комбинируем инструменты

Тип	Инструмент	Где работает	Покрытие рисков
SAST	Semgrep / SonarQube Comm.	Код, PR	≈ 60 %
SCA	Snyk Free / Dep-Check	CI/CD	≈ 20 %
Secrets	Gitleaks / TruffleHog	Pre-commit / CI	≈ 10 %
DAST	OWASP ZAP / Nikto	Stage / Prod	≈ 10 %
IaC Security	Checkov / Trivy	Infra as Code	≈ 10 %

Совокупно: до 80 % типовых уязвимостей без лицензий

Реальная польза бесплатных инструментов

Что дают даже самые простые решения:

- ☑ Ранний фидбек — уязвимости ловятся до релиза
- ☑ Рост осведомлённости разработчиков
- ☑ Автоматизация рутинных проверок
- ☑ Экономия времени и бюджета

Не чем сканируешь — а что делаешь с результатами.

Полезные ресурсы и инструменты

GitHub-репозитории:

- OWASP Dependency-Check
- Semgrep Rules
- Gitleaks
- Checkov
- OWASP ZAP

Ресурсы и чеклисты:

- OWASP Top 10
- GitHub Security Lab
- DevSecOps Maturity Model
- cheat.sh — примеры CLI-команд

Top-5 ошибок при внедрении бесплатных инструментов

1️⃣ **Без владельца процесса**
→ Инструмент установлен, но никто не следит за результатами.
2️⃣ **Без приоритизации**
→ Ловят всё подряд, но не понимают, что критично.
3️⃣ **Без обновлений**
→ Старые базы CVE = ложная уверенность в безопасности.
4️⃣ **Без обучения команды**
→ Разработчики не знают, как исправлять находки.
5️⃣ **Без автоматизации**
→ Ручные проверки → забываются, ломаются, устаревают.

Без культуры безопасности даже лучшие инструменты бесполезны.

Выводы

Бесплатные инструменты — не панацея, но отличный старт.

- Дают 80 % покрытия без затрат
- Помогают строить культуру DevSecOps
- Доступны каждому разработчику

Если безопасность не встроена в процесс — она всегда будет опциональной.

Бесплатные инструменты ИБ — краткий обзор

Инструмент	Назначение	Этап (тип)	Основные ограничения
Semgrep	Статический анализ кода (поиск паттернов уязвимостей)	SAST	Ограниченные встроенные правила, ручная настройка YAML
SonarQube Community	Анализ качества и безопасности кода	SAST	Нет branch analysis, нет PR-комментариев, только main ветка
OWASP Dependency-Check	Анализ зависимостей на известные CVE	SCA	Медленная первая загрузка, устаревшие базы CVE без обновления
Snyk Free	Проверка зависимостей и контейнеров	SCA / Container	Лимит сканов в месяц, нет SLA
Gitleaks	Поиск секретов и токенов в коде	Secret Scan	Нет GUI, требует CLI и конфигурации
TruffleHog	Глубокий поиск утечек в истории Git	Secret Scan	Медленный анализ больших репозиториев
OWASP ZAP	Динамическое тестирование веб-приложений	DAST	Нет headless API-интеграций, требует ручной настройки правил
Nikto	Сканирование веб-серверов	DAST	Старый интерфейс, ограниченные шаблоны
Trivy	Проверка контейнеров и IaC на уязвимости	Container / IaC	Может пропускать кастомные образы, нет глубокой интеграции
Checkov	Анализ Terraform, Kubernetes, CloudFormation	IaC Security	False positives, нет отчётности для менеджмента
Terrascan	Анализ инфраструктурного кода	IaC Security	Небольшое комьюнити, слабая документация

VK Cloud

Станислав Погоржельский,
технологический евангелист
VK Cloud

Спасибо за внимание

tech