AKTIV.
CONSULTING

Вебинар, август 2023

Отраслевая специфика и нюансы внедрения безопасной разработки ПО **в промышленных предприятиях**

Александр Моисеев

Ведущий консультант по информационной безопасности AKTIV.CONSULTING

Организационный слайд

Всем участникам будет выслана **презентация и запись видео**

Задавайте вопросы на вкладке «**Вопросы**»

Автор самого лучшего вопроса **получит подарок** от AKTIV.CONSULTING

О чем сегодня пойдет речь

1

Обеспечение безопасной разработки ПО в промышленных организациях

2

Реализация проверок безопасности для «унаследованного» ПО

Новая регуляторика
и методические рекомендации

Национальные стандарты и НПА

- 239 приказ ФСТЭК
- ГОСТ Р 56939 +проекты
- 76 приказ ФСТЭК и Методика ВУ и НДВ *(ограниченного распространения)*
- Профили защиты *(МЭ, АВЗ, ОС, …)*
- Требования безопасности *(виртуализация, контейнеризация, СУБД, …)*
- Указы Президента № 166, 250

Новые методические рекомендации

- Методика тестирования обновлений (2022)
- Методика оценки уровня критичности уязвимостей (2022)
- Руководство по управлению уязвимостями (2023)
- В разработке Регламент сертификации процедур безопасной разработки (2023-2024)

Обеспечение безопасной разработки ПО в промышленных организациях

01

Особенности разработки ПО в промышленных организациях

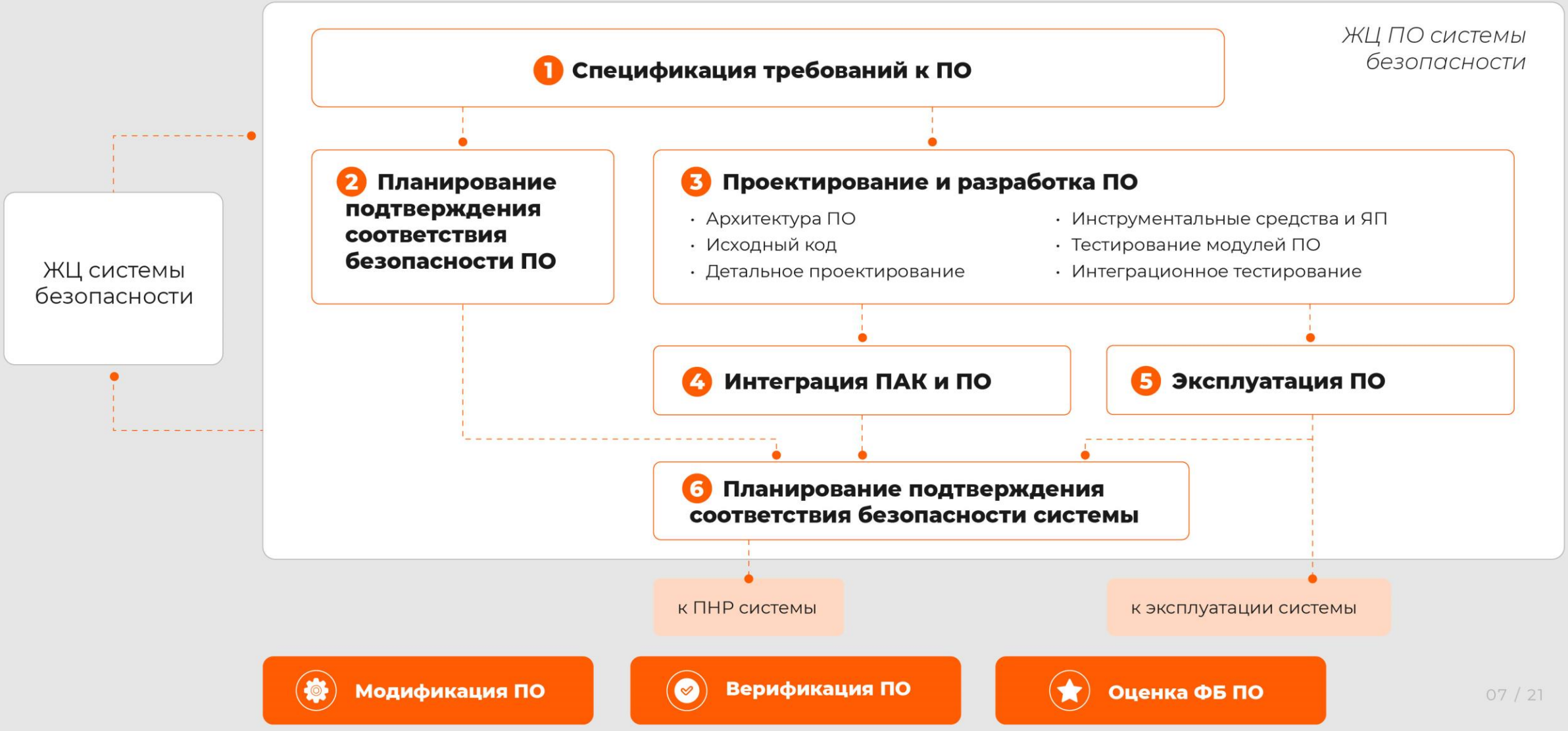
Бизнес аспекты:

1. Прикладное **ПО управляет** технологическими, производственными процессами
2. **Длительный ЖЦ**
3. Оказывает **влияние на здоровье** человека, экологию, ущерб инфраструктуре
4. Меры ИБ **должны предотвращать ухудшение** эффективности мер ФБ *(ГОСТ Р 59506)*

Технические аспекты:

1. **Специфические** ЯП и протоколы
2. **Повышенные требования** устойчивости к отказам
3. **Долгий цикл** предварительных и приемочных испытаний, оценки соответствия
4. **Возможна связь с безопасностью:**
   - промышленной
   - функциональной безопасностью *(МЭК 61508)*
   - зданий и сооружений *(ГОСТ 34332)*
   - атомной *(МЭК 63096)*
   - машин и механизмов *(Machinery Directive)*
   - и т. п.

Эталонная модель ЖЦ ПО ФБ (МЭК 61508)



Методы и средства обеспечения безопасности

МЕТОДЫ

- Структурные методы *(CORE, JSD, ...)*
- Прослеживаемость *(прямая и обратная)*
- Стандарты кодирования
- Программирование с защитой
- Формальное доказательство *(методы CCS, CSP, VDM, Z)*

СРЕДСТВА

- Универсальный язык программирования *(UML)*
- Статический анализ
- Динамический анализ и тестирование
- Сертифицированные инструментальные средства
- Функциональное тестирование и тестирование методом «черного ящика»
- Стресс-тестирование

Рекомендуемый набор мер для обеспечения безопасности при разработке

Управление доступом к среде разработки

Формирование рекомендаций по безопасному программированию

Установление требований ИБ в отношении сторонних компонентов и коллективов разработки

Управление уязвимостями в сегменте разработки и продуктовой среде

Тестирование на проникновение

Фаззинг

Реализация проверок безопасности для «унаследованного» ПО

02

Проблематика

Основные проблемы:

Неподдерживаемый код

Нет документации к ПО

Используются устаревшие версии ЯП, библиотек

Не используются технологии защиты кода (DEP, ASLR, safe stack и т.п.)

Проблематика

Основные угрозы:

Выполнение произвольного кода

Раскрытие конфиденциальной информации

Отказ в обслуживании

Недекларированный функционал

Отладочная информация

Основные уязвимости:

Переполнение буфера

Use after free

Состояние гонки

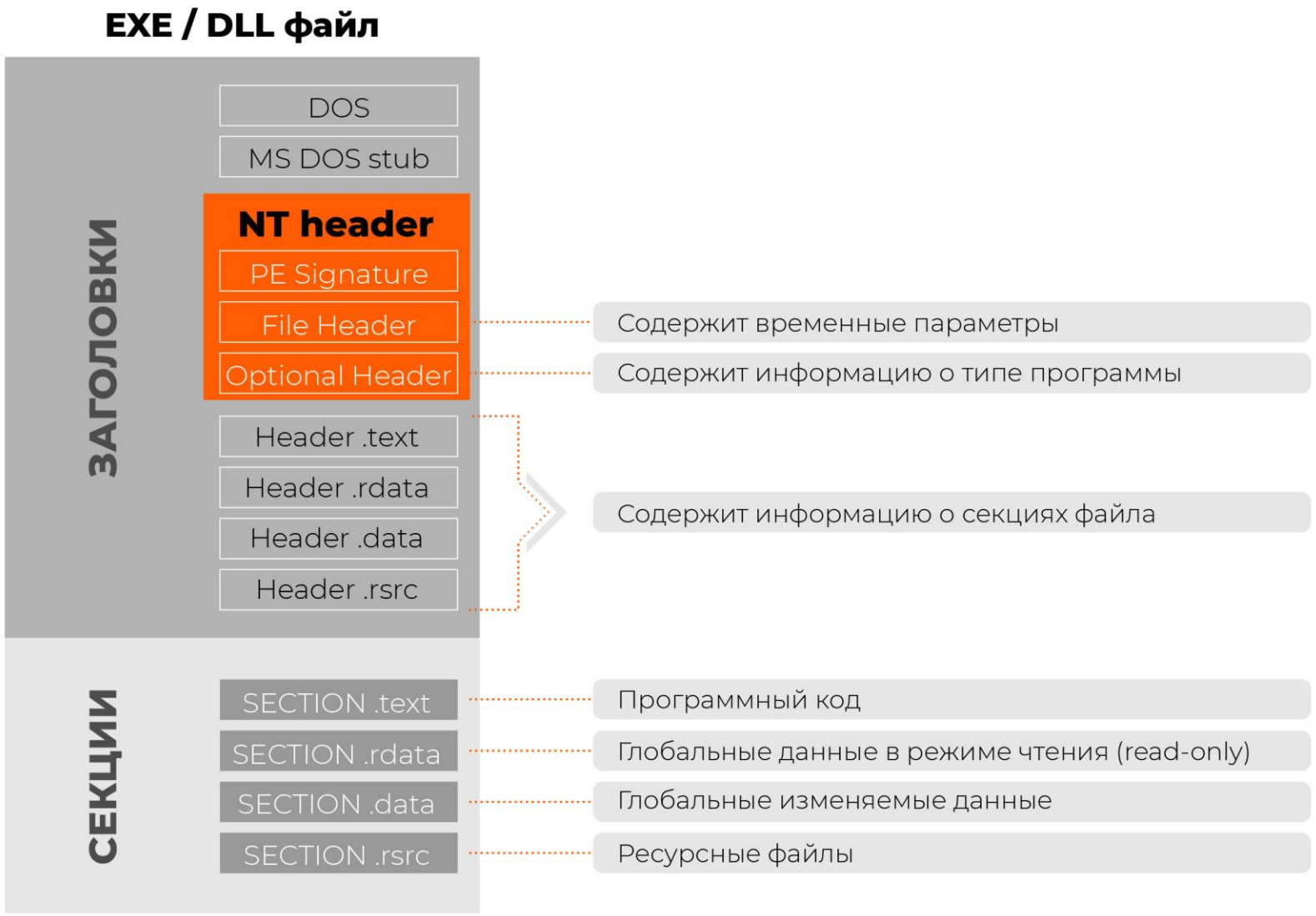
Исследование исполняемого файла методом «серого ящика»

1	Статический анализ «1/2» Динамический анализ «1/2»	Поиск нетипичных артефактов или поведения
2	Фаззинг	Локализация ошибок
3	Статический анализ «2/2» Динамический анализ «2/2»	Дизассемблирование и отладка

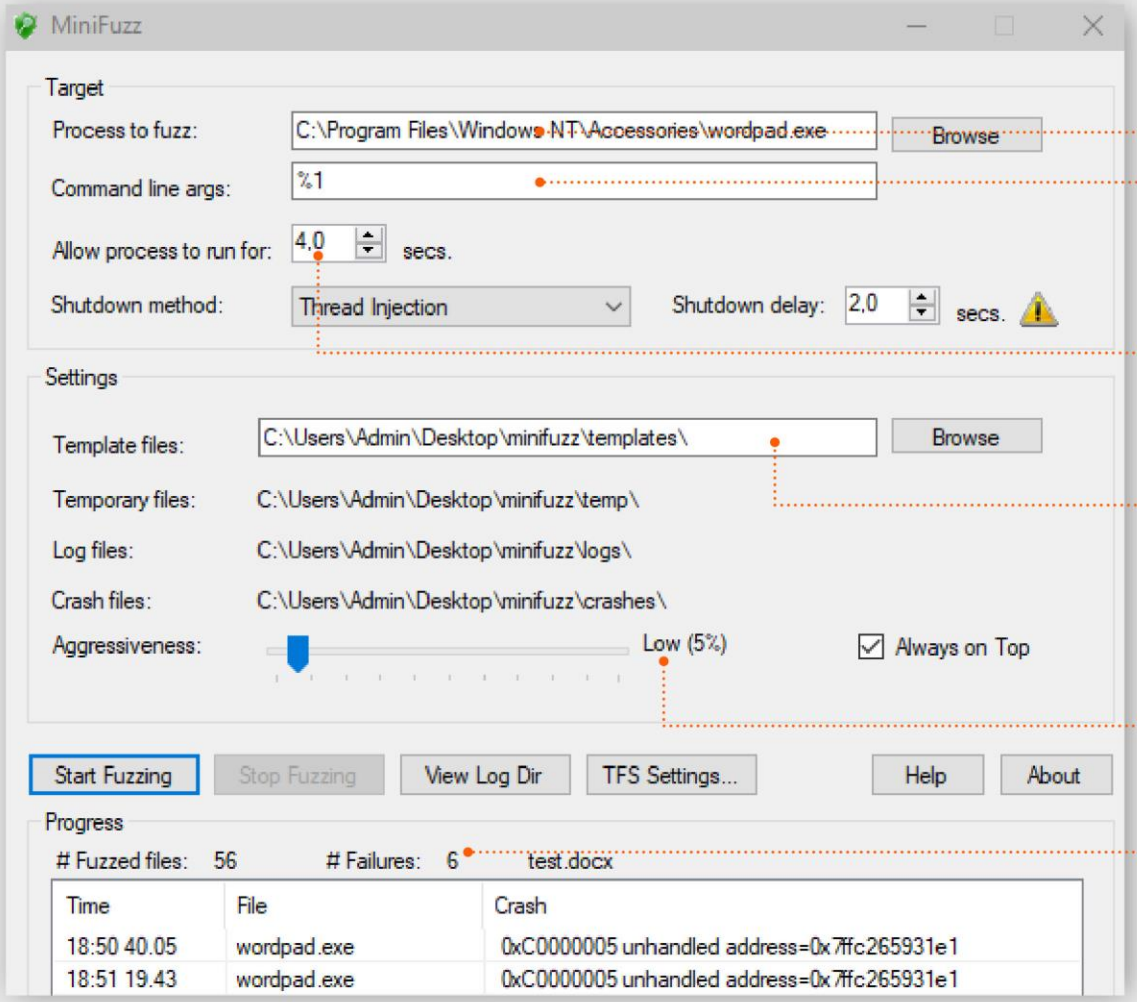
Поиск нетипичных артефактов или поведения

Метод	Статический анализ «1/2» Без запуска исполняемого файла	Динамический анализ «1/2» С запуском исполняемого файла
Особенности метода	Исследуем свойства и метаданные исполняемого файла	Запуск в стендовом окружении
Исследования исполняемого файла	Секции-PE заголовка Значения Strings Экспорт и импорт функций Подключаемые библиотеки Аномальные объекты или данные	Создание процессов Изменения в файловой системе Сетевая активность Изменения в палнировщике задач

Примеры: исследование PE-заголовка



Попытка вызвать ошибку (на примере фаззера файлов MiniFuzz)



Полный путь до приложения

Передаваемые тестируемому приложению опции командной строки

Время функционирования приложения до принудительного завершения

Путь до папки с файлами шаблонами

Степень искажения файла шаблона

Счетчик ошибок

Дизассемблирование и отладка

Метод	Статический анализ «2/2»	Динамический анализ «2 / 2»
Особенности метода	Без запуска исполняемого файла Исследуем ассемблерный листинг кода или псеводокод	С запуском исполняемого файла Запуск в стендовом окружении
Исследования исполняемого файла	Порядок вызова функций Логику работы кода	Состояние программы Порядок и последовательность работы интересующего участка кода Состояние памяти и стэка

Пример:
листинг дизассемблированного кода

CPU - main thread, module

```
00DF157B    CC              INT3
00DF157C  $ 8BFF            MOV EDI,EDI
00DF157E  . 55              PUSH EBP
00DF157F  . 8BEC            MOV EBP,ESP
00DF1581  . 83EC 14         SUB ESP,14
00DF1584  . A1 0080DF00     MOV EAX,DWORD PTR DS:[DF8000]
00DF1589  . 33C5            XOR EAX,EBP

00DF15B2  . 68 A410DF00     PUSH Seldon1_.00DF10A4          ProcNameOrOrdinal = "CheckTokenMembership"
00DF15B7  . 56              PUSH ESI                        hModule
00DF15B8  . FF15 84A0DF00   CALL DWORD PTR DS:[<&KERNEL32.GetProcAd  GetProcAddress
00DF15BE  . 8BF8            MOV EDI,EAX

00DF1608  > 8B4D FC         MOV ECX,DWORD PTR SS:[EBP-4]
00DF160B  . 8BC3            MOV EAX,EBX
00DF160D  . 5E              POP ESI
00DF160E  . 33CD            XOR ECX,EBP
00DF1610  . 5B              POP EBX
00DF1611  . E8 D7530000     CALL Seldon1_.00DF69ED
00DF1616  . 8BE5            MOV ESP,EBP
00DF1618  . 5D              POP EBP
00DF1619  . C3              RETN
00DF161A    CC              INT3
```

В качестве резюме

01

При внедрении БРПО для ЗО КИИ важно учитывать **отраслевую специфику и практики,** связанные с промышленной и функциональной безопасностью

02

Часть свидетельств безопасности ПО может содержаться в имеющейся **проектной документации**

03

Для проверок безопасности «унаследованного» ПО необходимо применять **методы** «**серого ящика**»

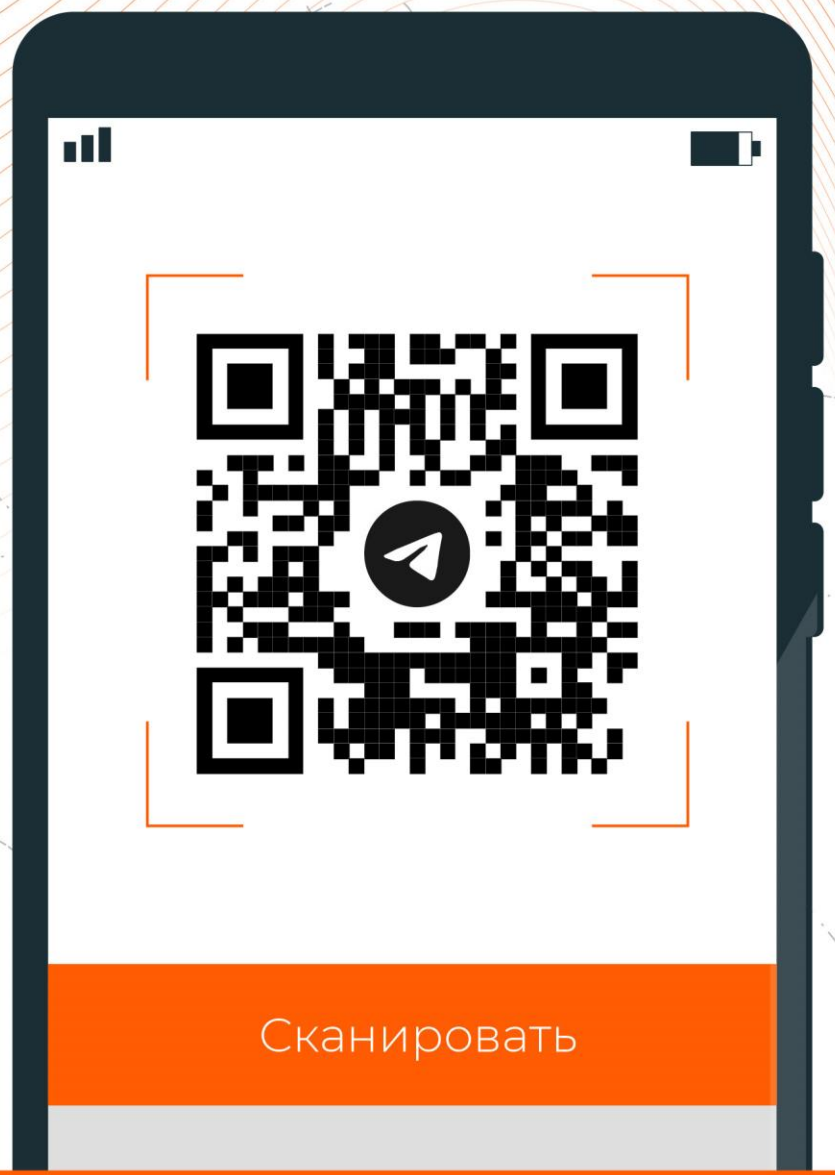
04

Для тренировки навыков написания безопасного кода разработчикам нужна постоянная **тренировка практик security**

05

Рекомендуем дополнить регуляторные требования **собственными проверками** безопасности *(управление секретами, SCA/OSA, SBOM, автоматизированные проверки безопасности в CI/CD и т.д.)*

AKTIV.
CONSULTING



AKTIV.
CONSULTING

Александр Моисеев

Ведущий консультант по информационной безопасности AKTIV.CONSULTING

moiseev@aktiv.consulting