DevSecOps

весна 2023

Вопросы к экзамену/зачёту/собеседованию

CWE, CVE

Secrets Detection, Secret manager

SAST, DAST, SCA, DevSecOps

Фреймворки безопасной разработки

- Microsoft SDL

- Jet DSO Framework

CWE

Common Weakness Enumeration

база данных дефектов безопасности, которые могут проявиться в архитектуре, проектировании, коде или реализации ПО и могут быть использованы злоумышленниками для получения несанкционированного доступа к системе.

https://cwe.mitre.org/

CVE

Common Vulnerabilities and Exposures

база данных общеизвестных уязвимостей

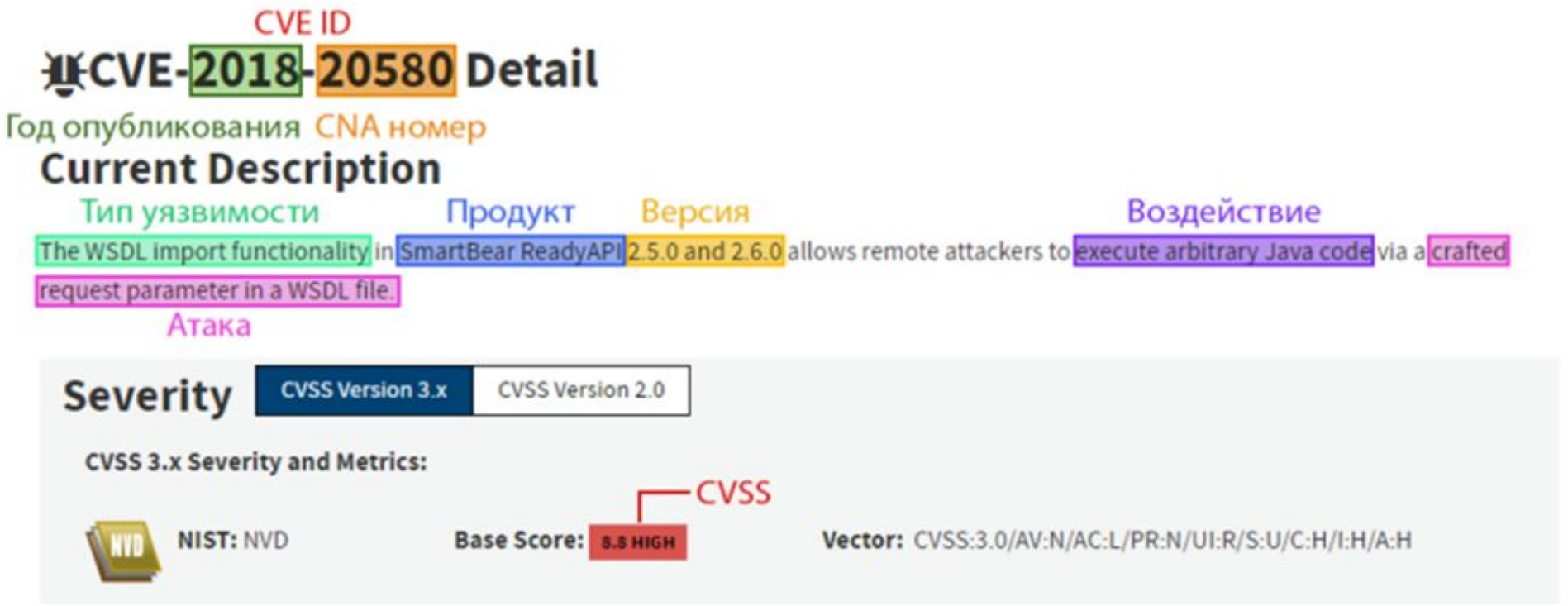
CVE-год-номер

https://cve.mitre.org/

https://www.cve.org/

https://github.com/CVEProject

CVE


CVE ID
CVE-2018-20580 Detail
Год опубликования
CNA номер
Current Description
Тип уязвимости
Продукт
Версия
Воздействие
The WSDL import functionality in SmartBear ReadyAPI 2.5.0 and 2.6.0 allows remote attackers to execute arbitrary Java code via a crafted request parameter in a WSDL file.
Атака
Severity
CVSS Version 3.x
CVSS Version 2.0
CVSS 3.x Severity and Metrics:
CVSS
NIST: NVD
Base Score: 8.8 HIGH
Vector: CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

CVSS

Common Vulnerability Scoring System

открытый стандарт для оценки степени опасности уязвимостей

По CVSS на основании набора метрик и формул вычисляется оценка опасности уязвимости, она может принимать значения от 0 до 10, где 10 – наивысший балл.

Калькулятор CVSS:

https://bdu.fstec.ru/calc

CWE, CVS, CVSS

CWE — перечень дефектов, способных спровоцировать уязвимости

CVE — список уязвимостей

CVSS — шкала опасности

Банк данных угроз безопасности информации

https://bdu.fstec.ru/vul

ФСТЭК России

Федеральная служба по техническому и экспортному контролю

ФАУ «ГНИИИ ПТЗИ ФСТЭК России»

Государственный научно-исследовательский испытательный институт проблем технической защиты информации

SAST

SAST (Static Application Security Testing) — процесс тестирования кода приложения на наличие ошибок и уязвимостей в исходном коде.

Метод “белого ящика”.

Обеспечивает соответствие руководствам и стандартам без фактического выполнения базового кода.

Работает на этапе написания кода = позволяет выявлять ошибки на ранних стадиях, что снижает затраты.

SAST

Хорошо интегрируется в процесс разработки, в IDE, в CI.

Что такое CI?

Хорошо выявляет риски переполнения буфера, SQL-инъекции, межсайтового скриптинга (XSS) и другие.

Точно указывает на подозрительный фрагмент кода.

SAST

Список инструментов статического анализа:

https://github.com/analysis-tools-dev/static-analysis

https://analysis-tools.dev/

Цикл лекций по статическому анализу кода и фаззинг-тестированию ПО:

https://bdu.fstec.ru/education

DAST

DAST (Dynamic Application Security Testing) — процесс проверки программы в рабочем состоянии для поиска уязвимостей.

Метод «чёрного ящика».

Анализирует приложения во время их выполнения, обнаруживая: повреждение памяти, незащищённые конфигурации серверов, риски межсайтового скриптинга, проблемы с правами пользователей, внедрение вредоносного SQL-кода и другие критически важные уязвимости.

DAST

Запускает автоматические проверки, имитирующие вредоносные внешние атаки на программу;

имитирует случайное поведение и действия пользователя

Цель состоит в том, чтобы определить неожиданные результаты.

Обычно тестирует все точки доступа, чтобы найти уязвимости.

DAST

Требует внимания специалистов по безопасности.

Для этого экспертам необходимо глубокое понимание тестируемого приложения, а также знание серверов приложений, баз данных, веб-серверов, потоков трафика приложений и списков контроля доступа.

https://owasp.org/www-community/Vulnerability_Scanning_Tools

SCA

SCA (Software Composition Analysis) — процесс определения компонентов (в особенности — с открытым исходным кодом), из которых состоит программное обеспечение, и проверки их безопасности.

По статистике используется более 90% кода open source;

бОльшая часть уязвимостей обнаруживается в транзитивных зависимостях.

SCA

Находит все связанные компоненты, поддерживающие их библиотеки, а также их прямые и транзитивные зависимости;

лицензии на программное обеспечение, уязвимости и потенциальные эксплойты.

В процессе сканирования создаётся спецификация, обеспечивающая полную инвентаризацию программных активов проекта.

SCA

- Повышение прозрачности
- Понимание логики зависимостей
- Оценка приоритета уязвимостей
- Повышение скорости безопасной разработки

https://owasp.org/www-community/Component_Analysis

Ещё немного инструментов для любознательных

OSA (Open Source Analysis)

IAST (Interactive Application Security Testing)

BAST (Business (или Behavioral) Application Security Testing)

BCA (Bytecode and Container Analysis)

WAF (Web Application Firewall)

Подходы к аудиту программных систем

- Проведение аудита приложения, кода, документации. В связи со значительными затратами времени на аудит значительно отстаёт от актуального приложения или значительно замедляет выпуск.

- Проведение аудита процесса разработки кода.

DevSecOps

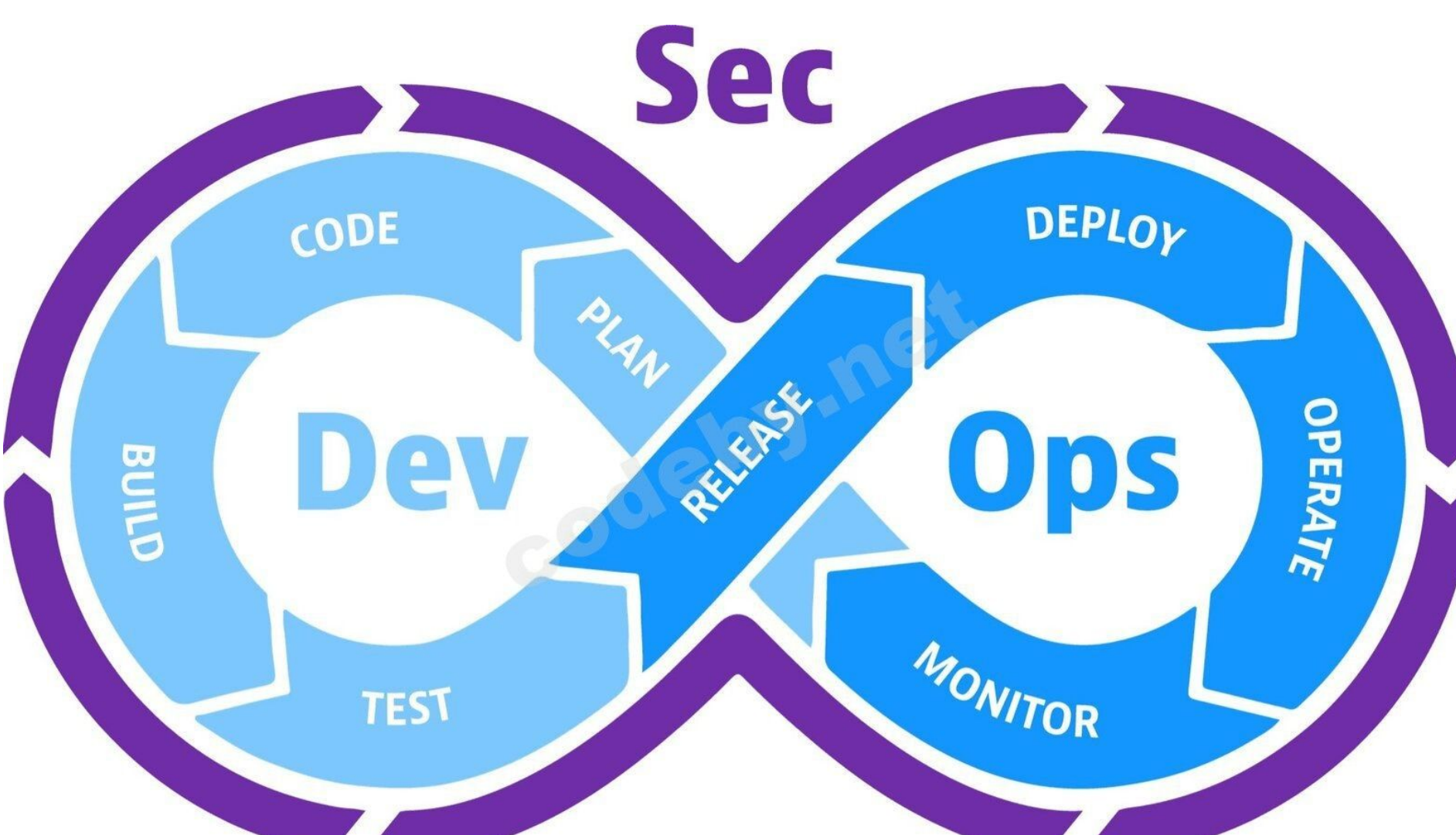
Что такое DevOps?

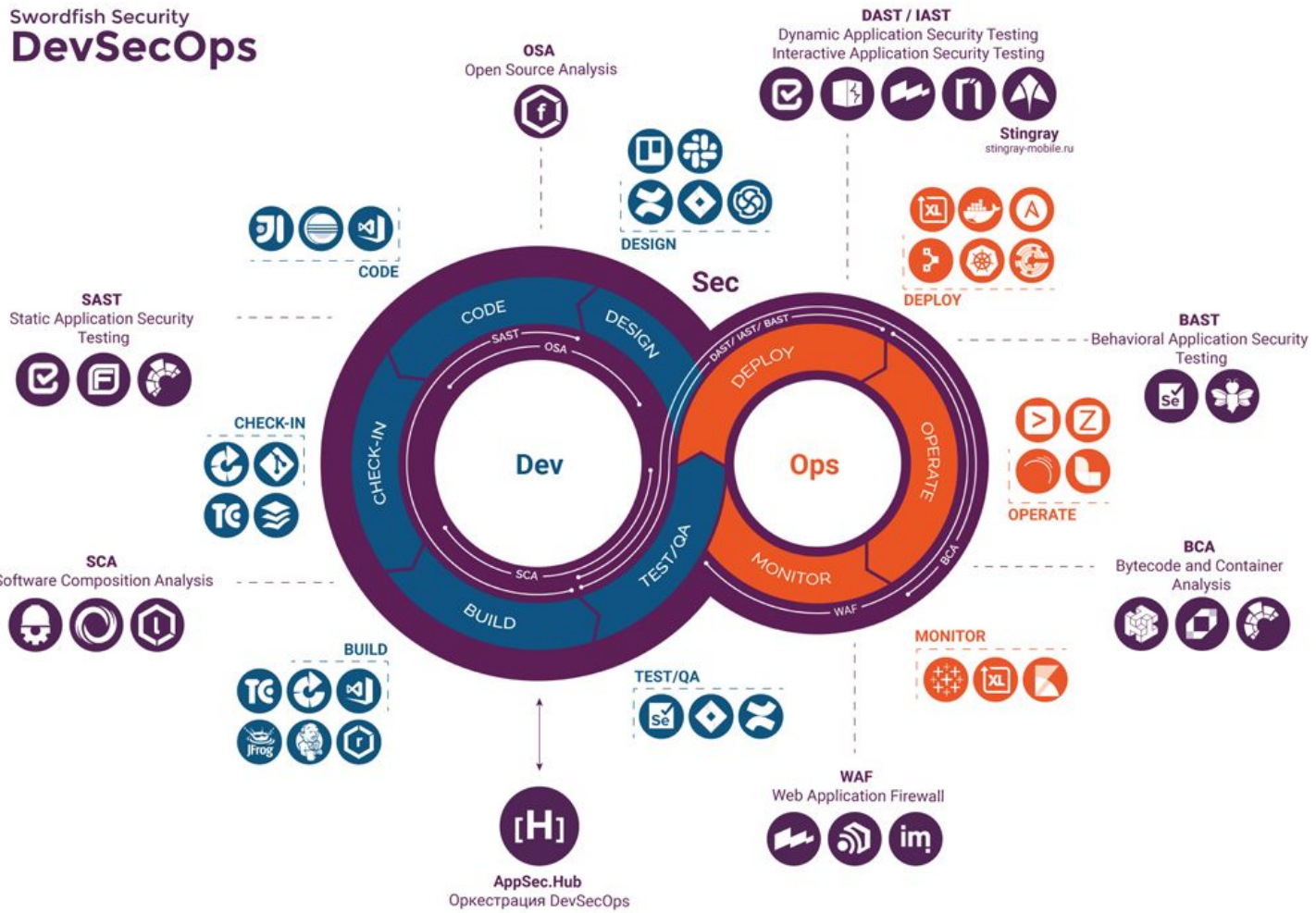
DevSecOps

Практика интеграции безопасности в каждый этап процесса разработки программного обеспечения.

Включает в себя инструменты и процессы, поощряющие сотрудничество между разработчиками, специалистами по безопасности и группами эксплуатации для создания эффективного и безопасного программного обеспечения.

DevSecOps привносит в компании культурную трансформацию, при которой за безопасность ответственны все, кто создает программное обеспечение.


Sec
CODE
PLAN
BUILD
Dev
TEST
RELEASE
DEPLOY
OPERATE
Ops
MONITOR


Swordfish Security
DevSecOps
OSA
Open Source Analysis
DAST / IAST
Dynamic Application Security Testing
Interactive Application Security Testing
Stingray
stingray-mobile.ru
DESIGN
CODE
DEPLOY
SAST
Static Application Security Testing
Sec
BAST
Behavioral Application Security Testing
CHECK-IN
Dev
Ops
OPERATE
SCA
Software Composition Analysis
BCA
Bytecode and Container Analysis
BUILD
MONITOR
TEST/QA
WAF
Web Application Firewall
AppSec.Hub
Оркестрация DevSecOps

Фреймворки безопасной разработки

Microsoft Security Development Lifecycle

BSIMM

OWASP SAMM

OWASP DSOMM

JDSOF

Microsoft SDL

https://www.microsoft.com/en-us/securityengineering/sdl/practices

Security Development Lifecycle

(жизненный цикл безопасной разработки) концепция разработки, заключающаяся в обеспечении безопасности разрабатываемого приложения, идентификации рисков и управления ими.

Развивается с 2000 года

SDL Timeline

The perfect storm

2000 — 2001

- Growth of home PC's
- Rise of malicious software
- Increasing privacy concerns
- Internet use expansion

SDL ramp up

2002 — 2003

- Bill Gates' TwC memo
- Microsoft security push
- Microsoft SDL released
- SDL becomes mandatory policy at Microsoft
- Windows XP SP2 and Windows Server 2003 launched with security emphasis

Setting a new bar

2004 — 2005 — 2006 — 2007

- Windows Vista and Office 2007 fully integrate the SDL
- SDL released to public
- Data Execution Prevention (DEP) & Address Space Layout Randomization (ASLR) introduced as features
- Threat Modeling Tool

Collaboration

2008 — 2009 — 2010

- Microsoft joins SAFECode
- Microsoft Establish SDL Pro Network
- Defense Information Systems Agency (DISA) & National Institution Standards and Technology (NIST) specify featured in the SDL
- Microsoft collaborates with Adobe and Cisco on SDL practices
- SDL revised under the Creative Commons License

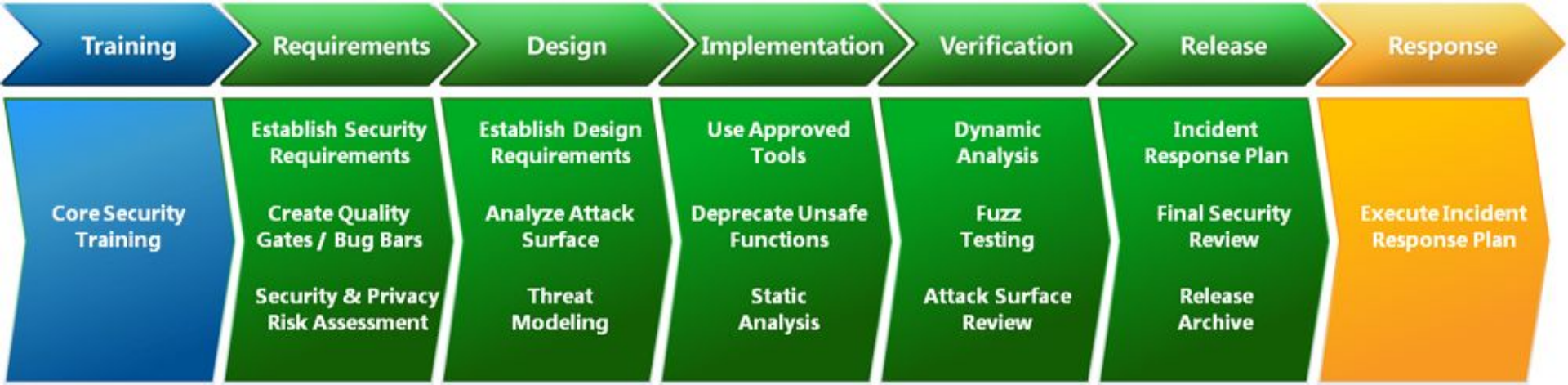
Selective tooling and Automation

2011 — 2018+

- Additional resources dedicated to address projected growth in Mobile app downloads
- Industry-wide acceptance of practices aligned with SDL
- Adaption of SDL to new technologies and changes in the threat landscape
- Increased industry resources to enable global secure development adoption

Microsoft SDL

Версия 2010



Этапы SDL

Pre-SDL: Security Training — обучение информационной безопасности.

Requirements — анализ требований всех заинтересованных сторон. Здесь должны учитываться в том числе модели угроз как для корпоративной инфраструктуры, так и для разрабатываемых систем.

Этапы SDL

Design (планирование, проектирование) — преобразования требований в план для реализации их в приложении. Здесь должны учитываться требования безопасного проектирования.

Implementation (разработка ПО) — фокусирование на качестве и факте реализации ранее спроектированных требований в коде приложения. В этот этап также входит проверка зависимостей.

Этапы SDL

Verification — верификация, тестирование. При этом важно проводить тестирования в части аспектов информационной безопасности, в частности использовать проверки на наличие уязвимостей в коде.

Release — развёртывание и сопровождение, в которое входит мониторинг событий безопасности и реагирование на инциденты.

Практики SDL

Provide Training

Define Security Requirements

Define Metrics and Compliance Reporting

Perform Threat Modeling

Establish Design Requirements

Define and Use Cryptography Standards

Практики SDL

Manage the Security Risk of Using Third-Party Components

Use Approved Tools

Perform Static Analysis Security Testing (SAST)

Perform Dynamic Analysis Security Testing (DAST)

Perform Penetration Testing

Establish a Standard Incident Response Process

BSIMM

Building Security In Maturity Model

https://www.bsimm.com/

модель оценки зрелости процесса безопасной разработки

содержит более структурированное описание SSDL

Развивается с 2008 года компанией Synopsys Software Integrity Group на основе интервью со специалистами по безопасности, в 2009 году вышла BSIMM1

BSIMM помогает

1. Оценивать текущий уровень процессов безопасной разработки и находить слабые места
2. Знакомиться с best practices и понимать, к чему нужно стремиться
3. Отслеживать повышение уровня зрелости
4. Составлять дорожные карты и грамотно внедрять процессы безопасной разработки
5. Определять уровень своей эффективности относительно других компаний

Домены BSIMM

Governance (Управление) — практики, которые отвечают за организацию, управление и оценку эффективности SSDL;

SSDL — жизненный цикл программного обеспечения с интегрированными контрольными точками безопасности ПО и его деятельности

Intelligence (База знаний) — практики для сбора и консолидации знаний в области ИБ внутри организации. Они нужны для того, чтобы внедрять и тиражировать практики разработки защищенного ПО в полном объеме;

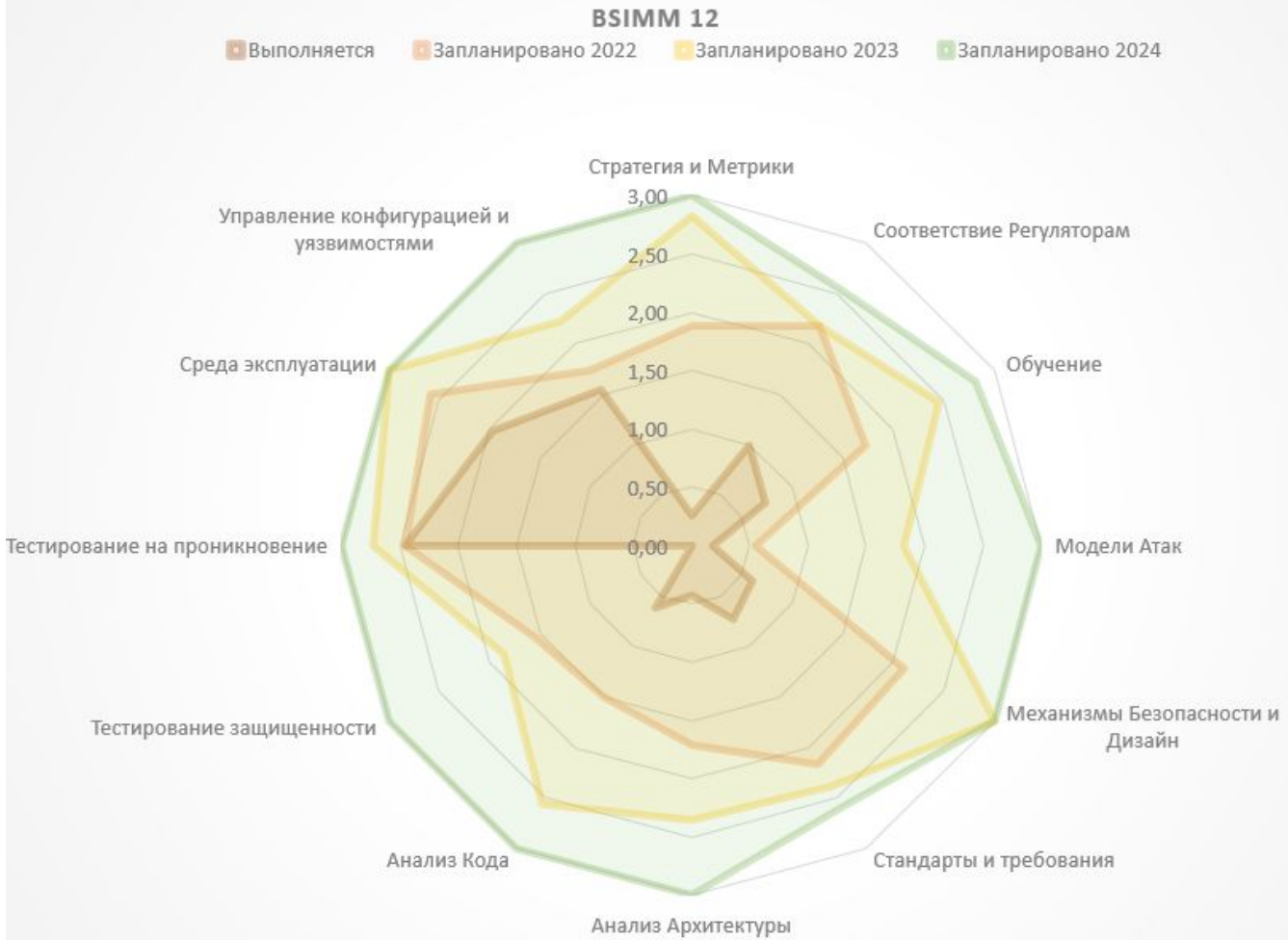
Домены BSIMM

SSDL Touchpoints (Точки соприкосновения с жизненным циклом разработки ПО) — практики для анализа и оценки конкретных артефактов и процессов в рамках производства ПО;

Deployment (Развёртывание и эксплуатация) — практики, которые отвечают за взаимодействие с подразделениями сетевой и инфраструктурной безопасности, а также службами технической поддержки.

Домены BSIMM

ПРАКТИКИ			
Управление	**База знаний**	**Точки соприкосновения с жизненным циклом разработки ПО**	**Развёртывание и эксплуатация**
1. Стратегия и метрики	4. Механизмы безопасности и дизайн	7. Анализ архитектуры	10. Тестирование на проникновение
2. Соответствие регуляторам	5. Модели атак	8. Анализ кода	11. Среда эксплуатации
3. Обучение	6. Стандарты и требования	9. Тестирование защищённости	12. Управление конфигурацией и уязвимостями


BSIMM 12
Выполняется
Запланировано 2022
Запланировано 2023
Запланировано 2024
Стратегия и Метрики
Соответствие Регуляторам
Обучение
Модели Атак
Механизмы Безопасности и Дизайн
Стандарты и требования
Анализ Архитектуры
Анализ Кода
Тестирование защищенности
Тестирование на проникновение
Среда эксплуатации
Управление конфигурацией и уязвимостями
3,00
2,50
2,00
1,50
1,00
0,50
0,00

OWASP (Open Web Application Security Project)

https://owasp.org/

Открытый проект по обеспечению безопасности веб-приложений (OWASP) представляет собой некоммерческий, образовательный, благотворительный фонд, помогающий организациям начать проектировать, разрабатывать, приобретать, использовать и поддерживать безопасное ПО. Все инструменты, документы, форумы и отделения OWASP являются бесплатными и открытыми для тех, кто заинтересован в улучшении безопасности приложений.

OWASP SAMM

Open SAMM (Open Software Assurance Maturity Model)

https://www.opensamm.org/

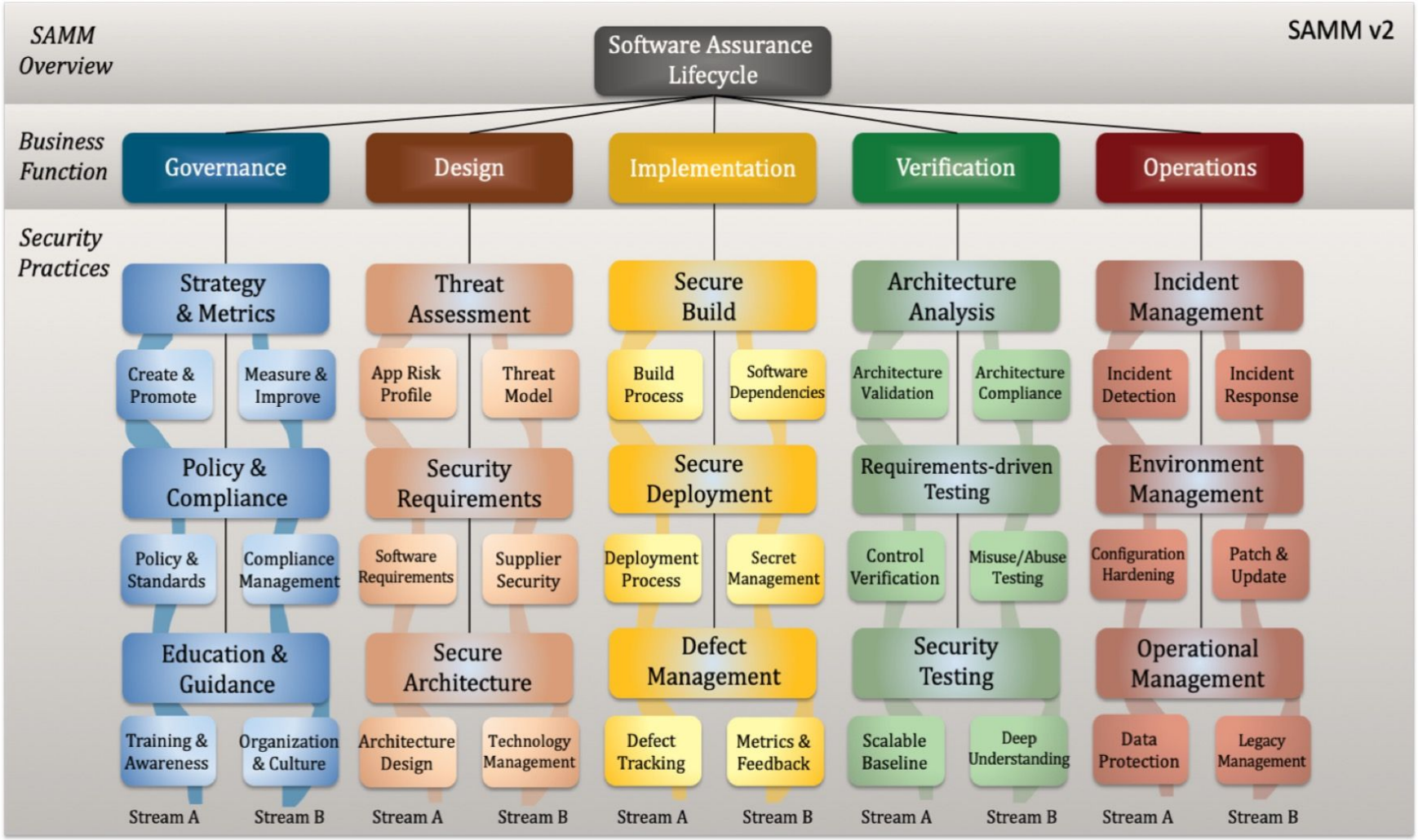
https://owaspsamm.org/

модель обеспечения безопасности ПО, фреймворк базы знаний и документации, помогающий построить цикл разработки безопасных приложений

Модули OWASP SAMM

- Описание самой модели, подхода к построению SDL;
- Опросник — большая анкета, отвечая на вопросы которой, вы поймете, на каком уровне сейчас находитесь. Это позволит сделать план, чтобы добраться до заветной цели.
- Шаблоны внедрения OWASP SAMM. В новой версии, помимо шаблонов, появились бенчмарки.

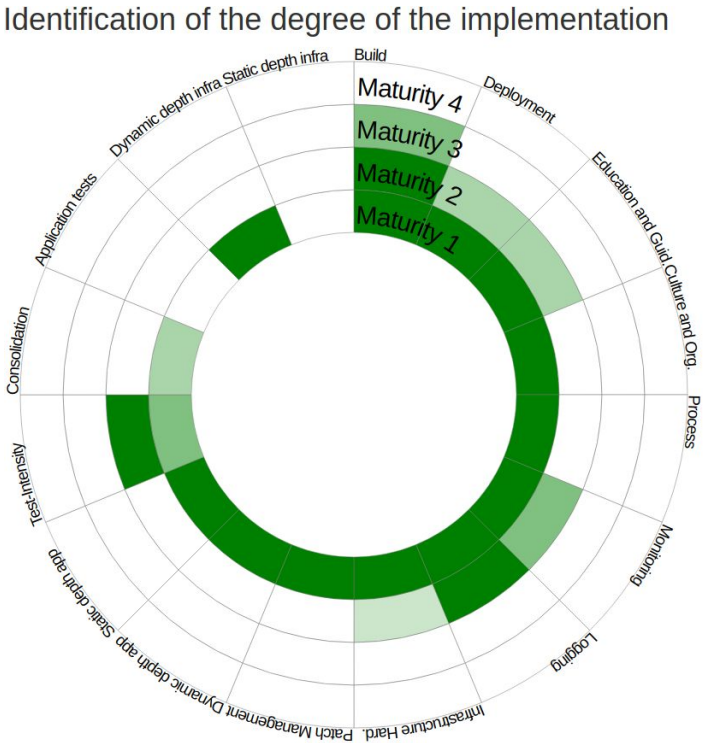
OWASP SAMM подразумевает три уровня зрелости.


SAMM Overview
SAMM v2
Software Assurance Lifecycle
Business Function
Governance
Design
Implementation
Verification
Operations
Security Practices
Strategy & Metrics
Create & Promote
Measure & Improve
Policy & Compliance
Policy & Standards
Compliance Management
Education & Guidance
Training & Awareness
Organization & Culture
Threat Assessment
App Risk Profile
Threat Model
Security Requirements
Software Requirements
Supplier Security
Secure Architecture
Architecture Design
Technology Management
Secure Build
Build Process
Software Dependencies
Secure Deployment
Deployment Process
Secret Management
Defect Management
Defect Tracking
Metrics & Feedback
Architecture Analysis
Architecture Validation
Architecture Compliance
Requirements-driven Testing
Control Verification
Misuse/Abuse Testing
Security Testing
Scalable Baseline
Deep Understanding
Incident Management
Incident Detection
Incident Response
Environment Management
Configuration Hardening
Patch & Update
Operational Management
Data Protection
Legacy Management
Stream A
Stream B
Stream A
Stream B
Stream A
Stream B
Stream A
Stream B
Stream A
Stream B

OWASP DSOMM

https://owasp.org/www-project-devsecops-maturity-model/

https://dsomm.owasp.org/



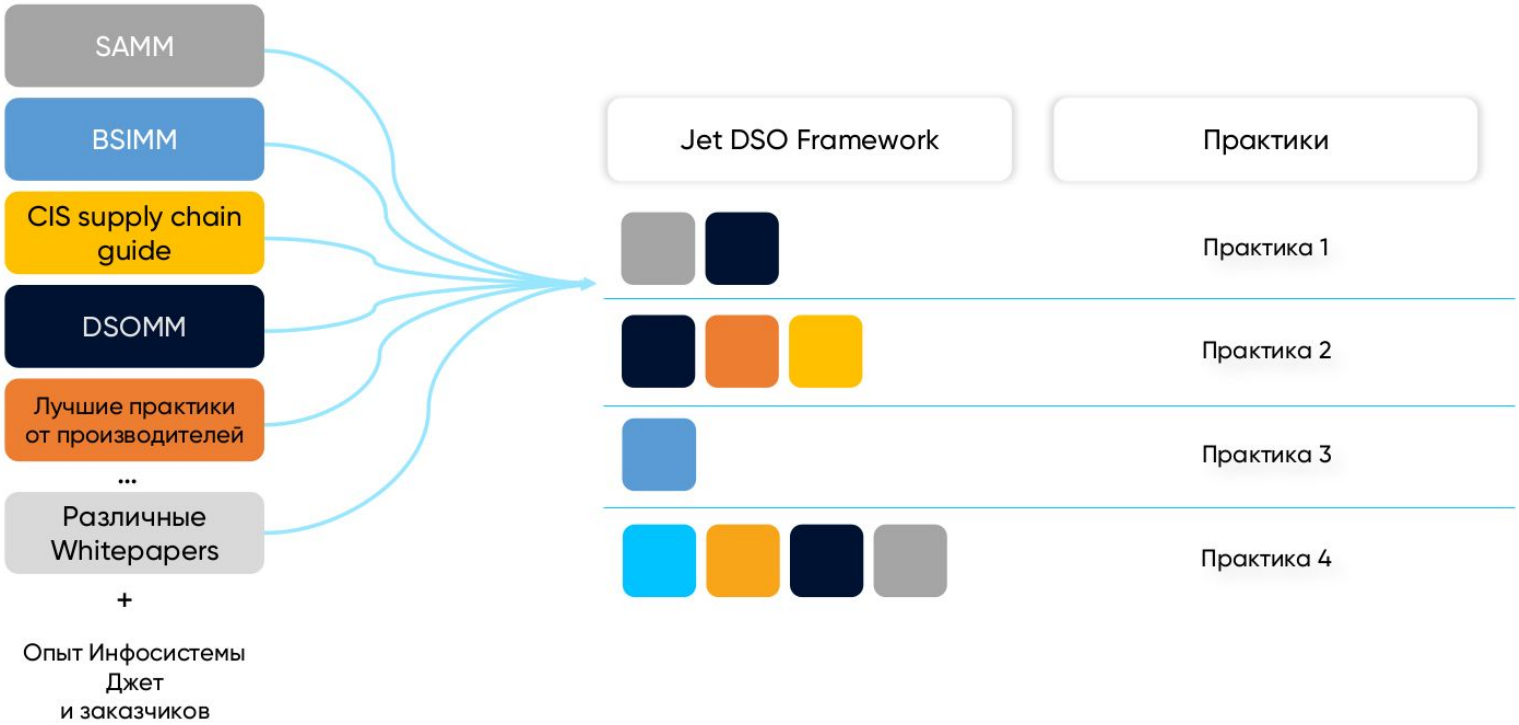
Jet DSO Framework

https://www.youtube.com/live/GnZc6m9YbYY?feature=share&t=2189

https://jet.su/devsecops/

Jet DSO Framework

https://www.youtube.com/live/GnZc6m9YbYY?feature=share&t=2189



Jet DSO Framework

Решаемые задачи:

- Определение текущего состояния AS IS
- Определение желаемого состояния TO BE
- Определение инициатив, необходимых для достижения целевого состояния
- Информация для бюджетирования реализации инициатив
- Определение вектора развития

Jet DSO Framework

Составляющие:

- Пиратская карта
- Таблица расчёта
- Пирамида зрелости

Cyber Camp

JDSOF: пиратская карта

Технологии

Контроль ИБ артефактов, зависимостей и образов

- Управление артефактами
- Контроль использования сторонних компонентов-зависимостей (библиотеки, образы и т.д.)

Защита окружения разработки

- Защита рабочих мест разработчика
- Защита секретов
- Защит Build-среды
- Защита реестра артефактов
- Контроль внесения изменений в исходный код
- Защита конвейера сборки

Контроль разрабатываемого ПО в части ИБ

Анализ ПО (development)

- Статический анализ
- Композиционный анализ
- Анализ образов контейнеров
- Идентификация секретов
- Контроль безопасности Dockerfile'ов

Анализ ПО в режиме Runtime – Preprod (после сборки, но до deploy в прод)

- Динамический анализ приложений
- Тестирование на проникновение перед внедрением приложений в продуктив
- Функциональное ИБ-тестирование
- Контроль конфигураций - Контроль безопасности манифестов (k8s, terraform и т.д.)

Защита ПО и инфраструктуры в режиме Runtime (monitor&operate)

- Управление секретами
- Управление контролем доступа к средам работы приложений
- Контроль сетевого трафика (L4-L7)
- Контроль выполняемых процессов и их прав доступа
- Анализ инфраструктуры и приложений на уязвимости
- Тестирование на проникновение продуктивной среды
- Анализ событий информационной безопасности

Процессы и методология

Обучение и Onboarding

- Обучение специалистов
- Управление базой знаний DSO
- Процесс «подключения» команд

Контроль и формирование требований ИБ к ПО

- Оценка критичности приложений и моделирование угроз
- Определение требований ИБ, предъявляемых к ПО
- Контроль выполнения требований ИБ
- Разработка стандартов конфигураций

Управление ИБ-дефектами

- Обработка дефектов ИБ
- Консолидация дефектов ИБ

Оценка эффективности процессов DSO

- Управление набором метрик ИБ
- Контроль исполнения метрик (как собираем и что с ними делаем)

Функциональные роли

- Security Champions
- Разграничение ролей процесса DSO

JDSOF: домен «технологии»

Окружение разработки

Контроль ИБ-артефактов, зависимостей и образов

- Управление артефактами
- Контроль использования сторонних компонентов-зависимостей (библиотеки, образы и т.д.)

Защита окружения разработки

- Защита секретов
- Защита Build-среды
- Защита рабочих мест разработчика
- Защита реестра артефактов
- Контроль внесения изменений в исходный код
- Защита конвейера сборки

Разработка кода

Контроль разрабатываемого ПО в части ИБ

Анализ ПО (development)

- Статический анализ
- Композиционный анализ
- Анализ образов контейнеров
- Идентификация секретов
- Контроль безопасности Dockerfile'ов

Тестирование (препрод)

Анализ ПО в режиме Runtime – Preprod (после сборки, но до deploy в прод)

- Динамический анализ приложений
- Контроль конфигураций - Контроль безопасности манифестов (k8s, terraform и т.д.)
- Функциональное ИБ-тестирование
- Тестирование на проникновение перед внедрением приложений в продуктив

Эксплуатация

Защита ПО и инфраструктуры в режиме Runtime (monitor&operate)

- Управление секретами
- Контроль сетевого трафика (L4-L7)
- Контроль выполняемых процессов и их прав доступа
- Анализ инфраструктуры и приложений на уязвимости
- Управление контролем доступа к средам работы приложений
- Тестирование на проникновение (pentest) продуктивной среды
- Анализ событий информационной безопасности

JDSOF: Таблица расчета

ГРУППА КРИТЕРИЕВ		LEVEL 0 UNINITIATED	LEVEL 1 BEGINNERS	LEVEL 2 INTERMEDIATE	LEVEL 3 ADVANCED	LEVEL 4 EXPERTS
Домен	Практика 1		Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n
	...					
	Практика n		Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n

Cyber Camp

JDSOF: Таблица расчета + тепловая матрица

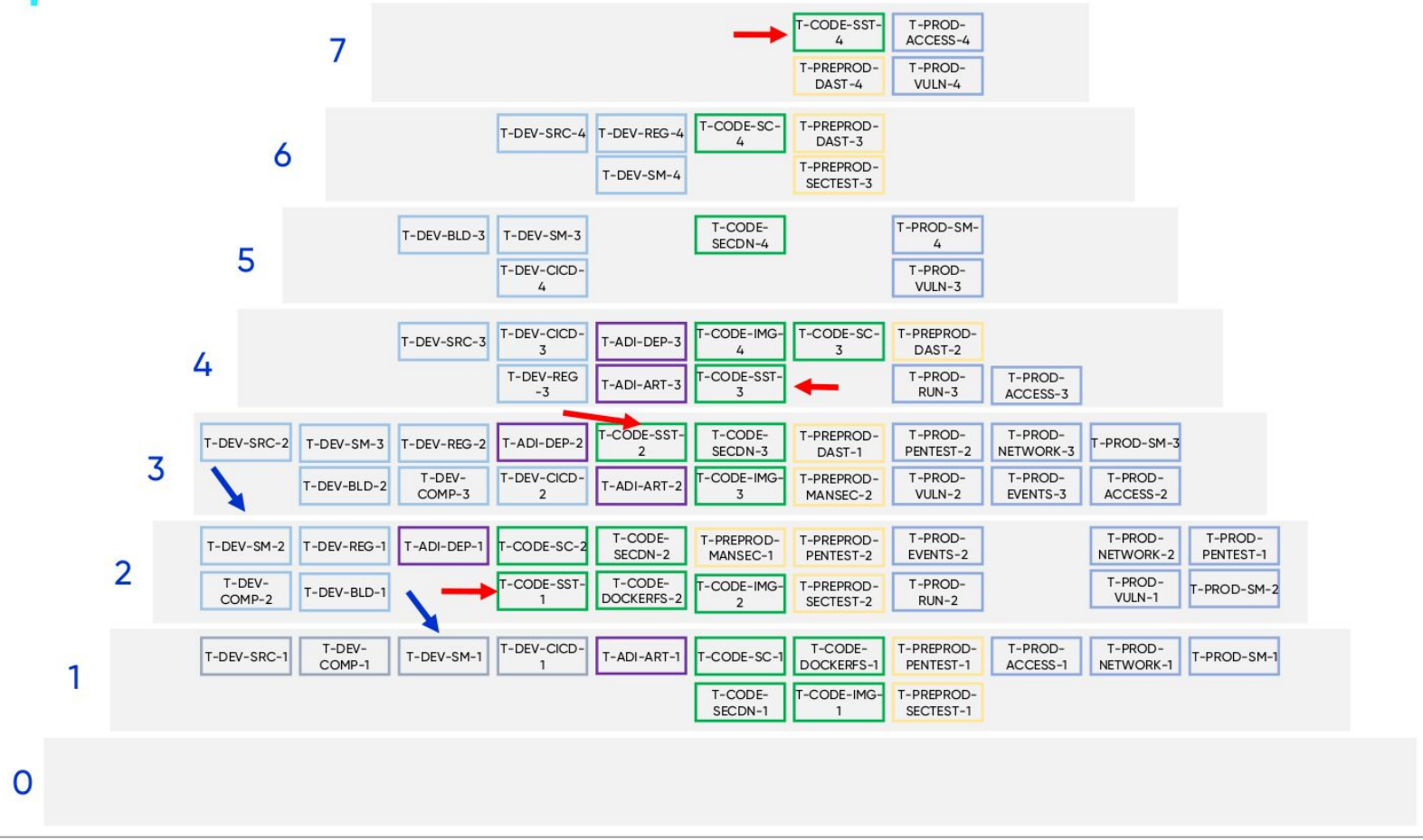
ГРУППА КРИТЕРИЕВ		LEVEL 0 UNINITIATED	LEVEL 1 BEGINNERS	LEVEL 2 INTERMEDIATE	LEVEL 3 ADVANCED	LEVEL 4 EXPERTS
Поддомен. Анализ ПО (T-CODE)	Статический анализ SAST (T-CODE-SST)		100%	60%	10%	0%
	Композиционный анализ (T-CODE-SC)		80%	65%	15%	0%
	Анализ образов контейнеров (T-CODE-IMG)		100%	100%	50%	0%
	Идентификация секретов (T-CODE-SECDN)		0%	0%	0%	0%
	Контроль безопасности Dockerfile'ов (T-CODE-DOCKERFS)		70%	30%	10%	0%

Cyber Camp

JDSOF: Пирамида зрелости

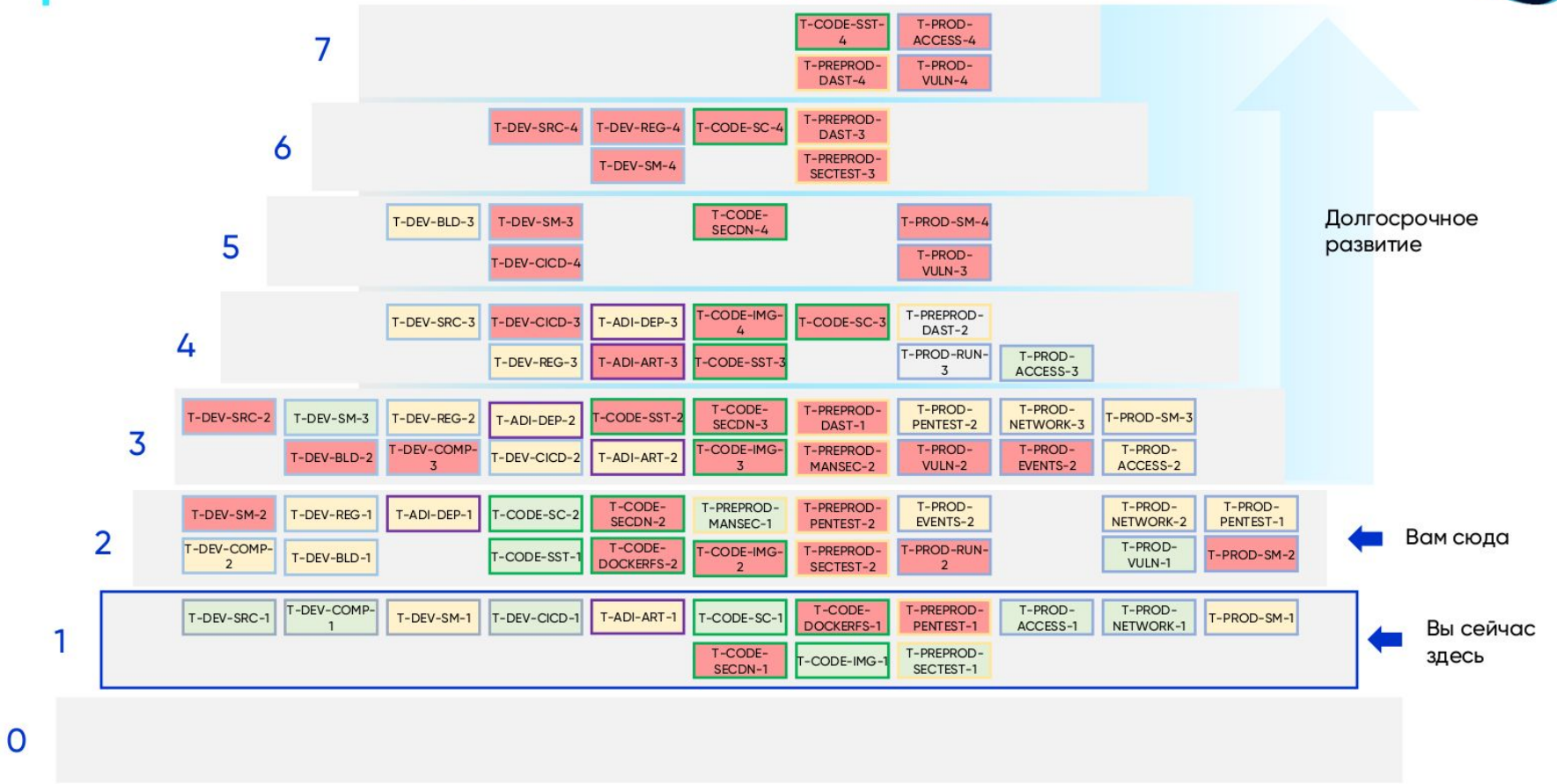
Cyber Camp

JDSOF: Пирамида зрелости



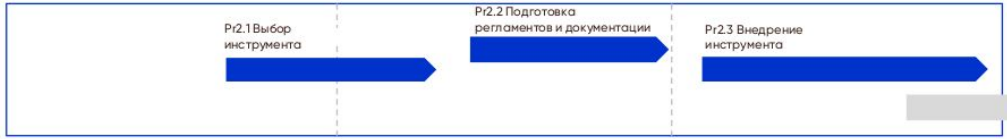
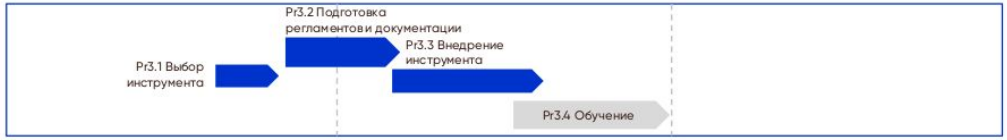
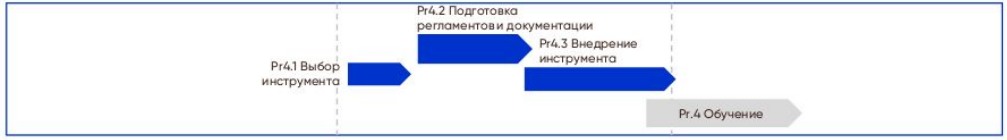
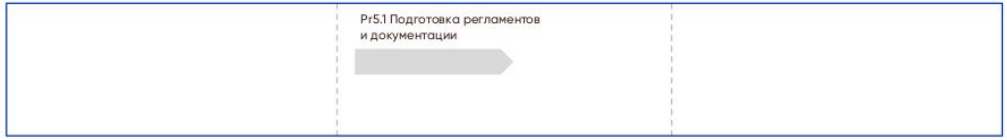
Cyber Camp

JDSOF: Пирамида зрелости



Дорожная карта развития функции по результатам аудита JDSOF

Cyber Camp

Процесс	План-график реализации (2023)	Стоимость развития, ₽	Выгоды от реализации активностей (верхнеуровневое описание)
Pr1 Внедрение инструмента Secrets Detection		₽ ₽	• Внедрение практики детектирования секретов
Pr2 Внедрение инструмента Secrets Management		₽ ₽ ₽	• Внедрение практики управления секретами
Pr3 Внедрение инструмента контроля Dockerfiles		₽	Внедрение инструмента контроля Dockerfiles
Pr4 Внедрение инструмента контроля образов контейнеров		₽	Внедрение инструмента контроля образов контейнеров
Pr5 Внедрение функционального тестирования ИБ		Реализуется внутренними силами компании	Внедрение функционального тестирования ИБ

ГОСТ Р 56939-2016

Защита информации. Разработка безопасного программного обеспечения. Общие требования.

Настоящий стандарт направлен на достижение целей, связанных с предотвращением появления и/или устранением уязвимостей программ, и содержит перечень мер, которые рекомендуется реализовать на соответствующих этапах жизненного цикла программного обеспечения.

Вопросы к экзамену/зачёту/собеседованию

CWE, CVE

Secrets Detection, Secret manager

SAST, DAST, SCA, DevSecOps

Фреймворки безопасной разработки

- Microsoft SDL

- Jet DSO Framework

Secret Manager

Secret Manager — сервис для безопасного хранения секретов: личных данных, ключей API, паролей, сертификатов и другой конфиденциальной информации.

Secret Manager работает в связке с системой управления криптографическими ключами Key Manager, тем самым обеспечивая надежное шифрование секретов.

Secret Detection

https://docs.github.com/en/code-security/secret-scanning/configuring-secret-scanning-for-your-repositories