

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-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:



NIST: NVD

Base Score: 8.8 HIGH

CVSS

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, CVE, CVSS

CWE имеет отношение к уязвимости, а не к экземпляру в продукте или системе

CVE имеет отношение к конкретному экземпляру в продукте или системе, а не к основному недостатку

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.

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.

Application: A Goat on the Web

1 of 6

Rows 10

Policy Violations	Component Filename	Version	License	Number of Known Vulnerabilities by Severity					
				V. High	High	Medium	Low	V. Low	Info
✓	spring-web-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		3				
✓	spring-web-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		3				
✓	commons-fileupload-1.2.2.jar	1.2.2	Apache License 2.0(Apache-2.0)		1		1		
✓	spring-beans-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		1				
✓	spring-core-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		1				
✓	spring-beans-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		1				
✓	spring-core-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)		1				
✓	commons-collections-3.1.jar	Unknown	Apache License 2.0(Apache-2.0)		1				
✓	spring-webmvc-3.0.5.RELEASE.jar	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)			3			
✓	spring-webmvc-	3.0.5.RELEASE	Apache License 2.0(Apache-2.0)			3			

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?

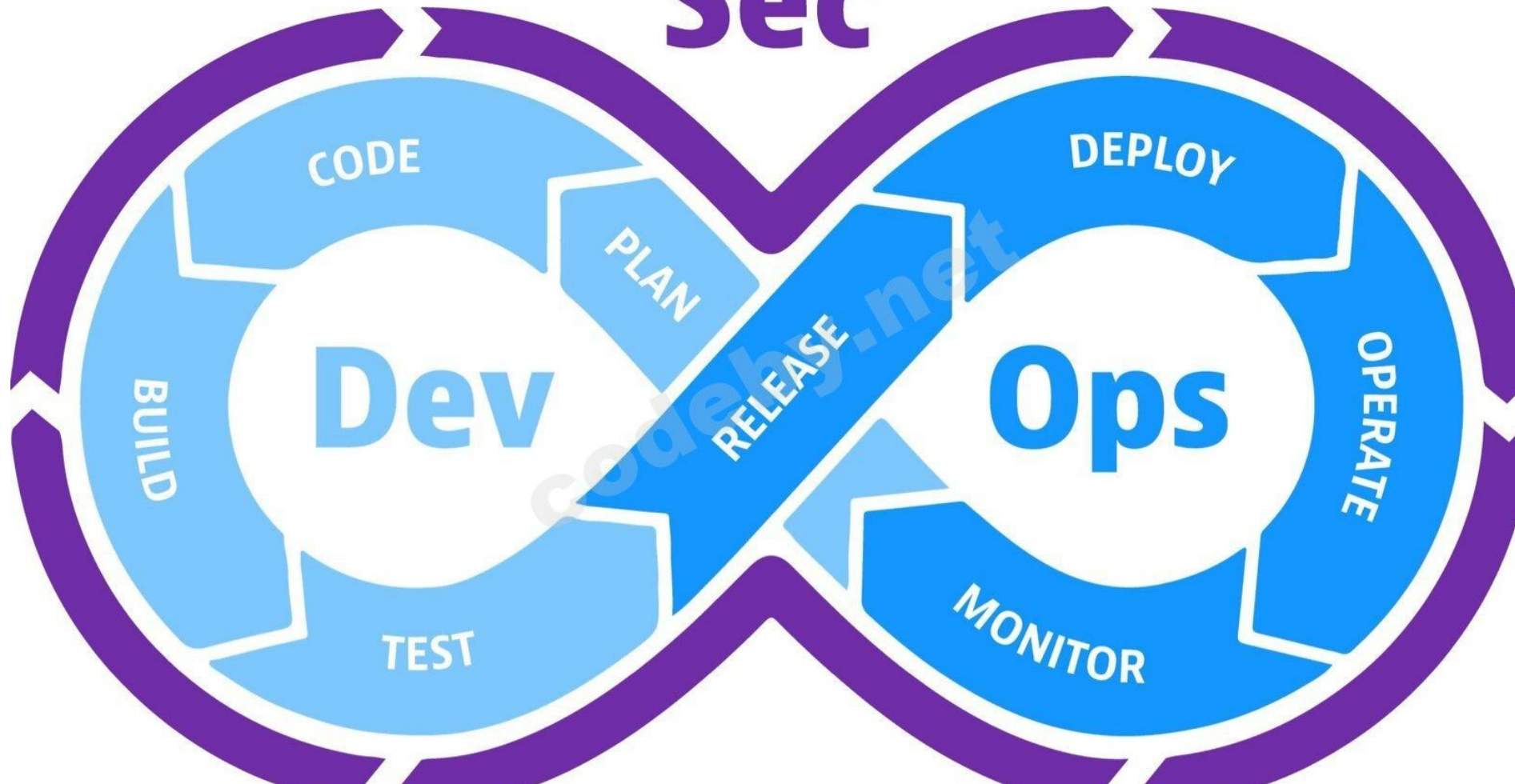
DevSecOps

Практика интеграции безопасности в каждый этап процесса разработки программного обеспечения.

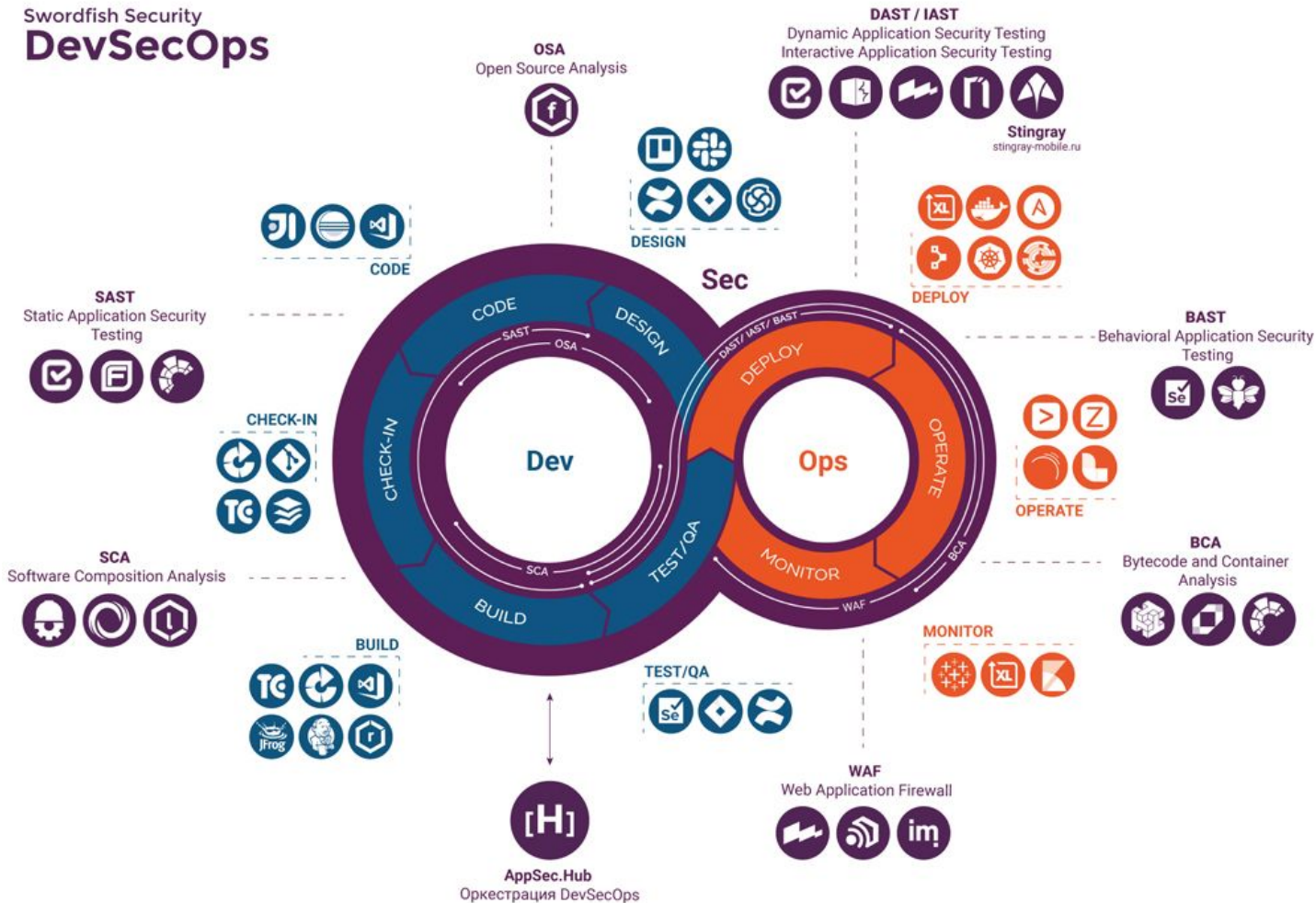
Включает в себя инструменты и процессы, поощряющие сотрудничество между разработчиками, специалистами по безопасности и группами эксплуатации для создания эффективного и безопасного программного обеспечения.

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

Sec



Swordfish Security
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



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

SDL ramp up



- 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



- 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



- 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

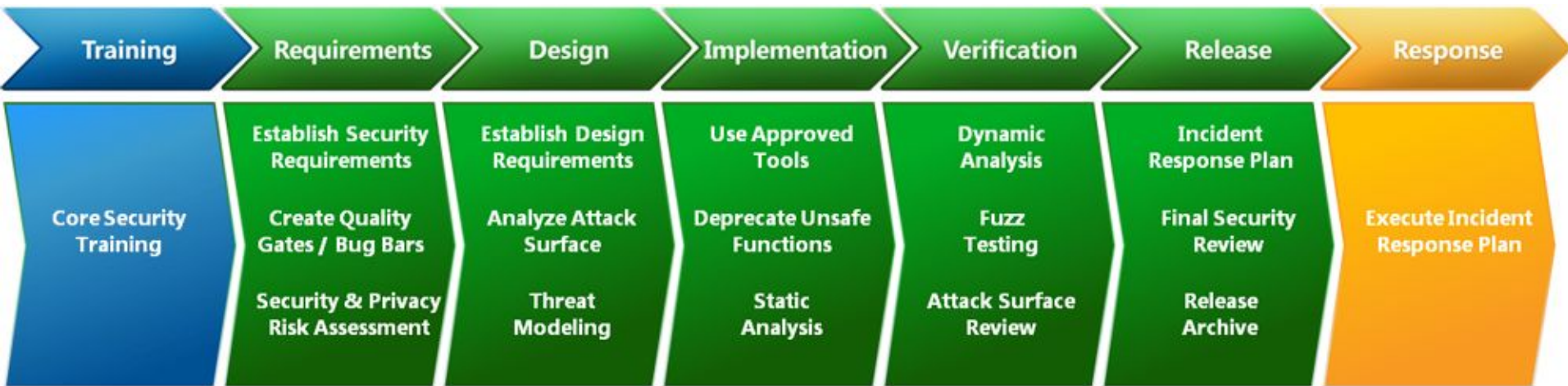


- Additional resources dedicated to address projected growth in Mobile app downloads
- Industry-wide acceptance of practices aligned with SDL
- Adaption of SDL to new technologies and changes in the threat landscape
- Increased industry resources to enable global secure development adoption

2000 — 2001 — 2002 — 2003 — 2004 — 2005 — 2006 — 2007 — 2008 — 2009 — 2010 — 2011 — 2018+ —>

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 (обеспечение обучения)

Define Security Requirements (определение требований безопасности)

Define Metrics and Compliance Reporting (определение метрик и отчетов о соответствии)

Perform Threat Modeling (моделирование угроз)

Establish Design Requirements (определение требований к дизайну)

Практики SDL

Define and Use Cryptography Standards (определение и использование стандартов криптографии)

Manage the Security Risk of Using Third-Party Components (управление рисками безопасности, связанными с использованием сторонних компонентов)

Use Approved Tools (использование одобренных инструментов)

Практики SDL

Perform Static Analysis Security Testing (статическое тестирование безопасности) = (SAST)

Perform Dynamic Analysis Security Testing (динамического тестирования безопасности) = (DAST)

Perform Penetration Testing (тестирование на проникновение)

Establish a Standard Incident Response Process (стандарт процесса реагирования на инциденты)

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;

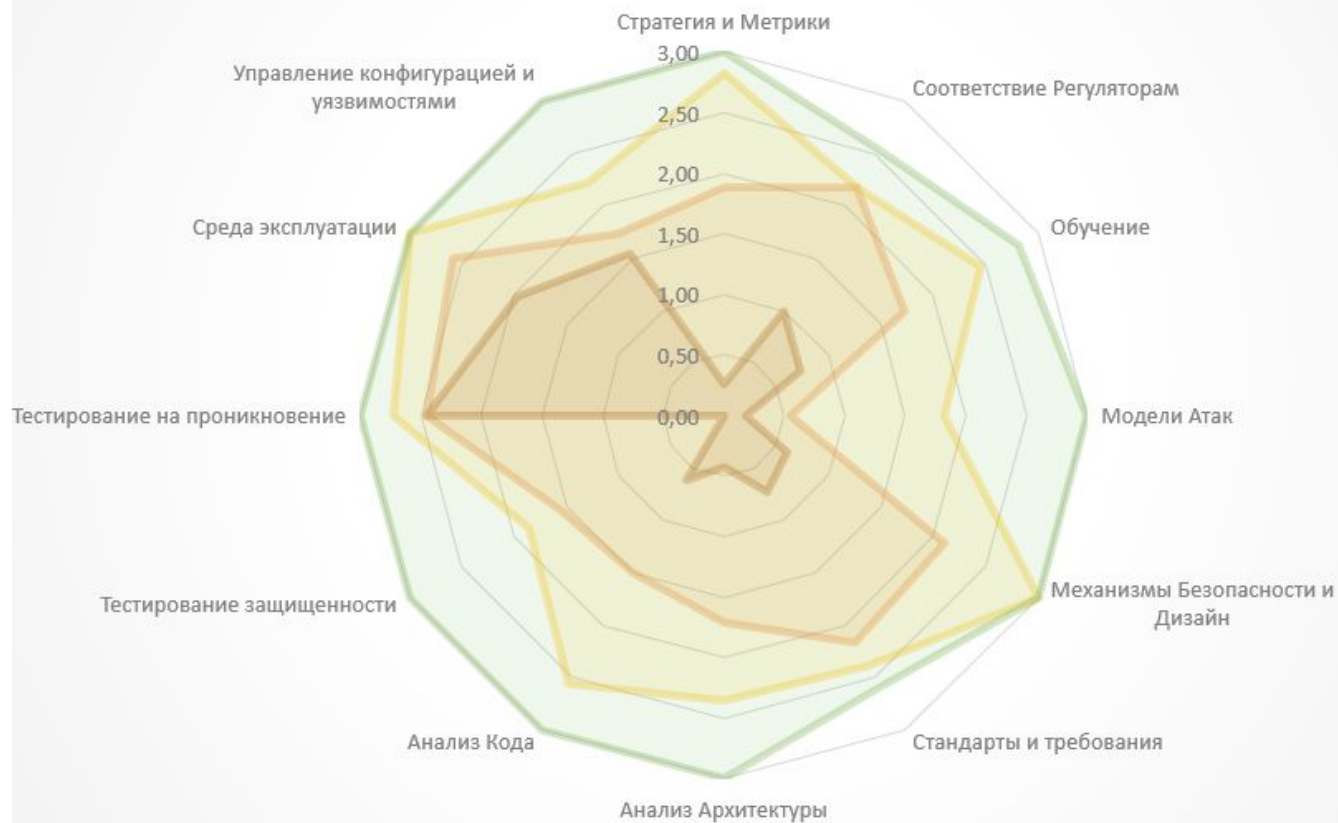
Deployment (Развёртывание и эксплуатация) — практики, которые отвечают за взаимодействие с подразделениями сетевой и инфраструктурной безопасности, а также службами технической поддержки.

Домены BSIMM

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

BSIMM 12

■ Выполняется ■ Запланировано 2022 ■ Запланировано 2023 ■ Запланировано 2024



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.

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

Stream A

Stream B

Threat Assessment

App Risk Profile

Threat Model

Security Requirements

Software Requirements

Supplier Security

Secure Architecture

Architecture Design

Technology Management

Stream A

Stream B

Secure Build

Build Process

Software Dependencies

Secure Deployment

Deployment Process

Secret Management

Defect Management

Defect Tracking

Metrics & Feedback

Stream A

Stream B

Architecture Analysis

Architecture Validation

Architecture Compliance

Requirements-driven Testing

Control Verification

Misuse/Abuse Testing

Security Testing

Scalable Baseline

Deep Understanding

Stream A

Stream B

Incident Management

Incident Detection

Incident Response

Environment Management

Configuration Hardening

Patch & Update

Operational Management

Data Protection

Legacy Management

Stream A

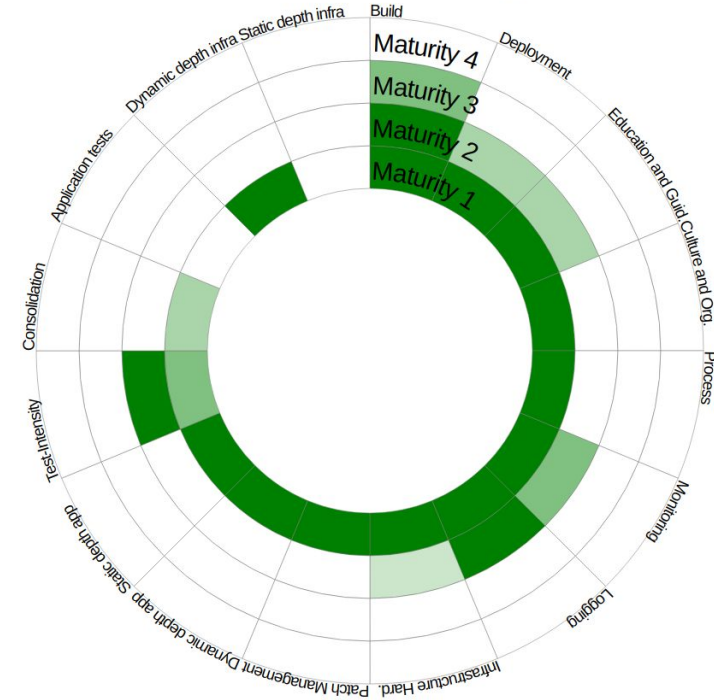
Stream B

OWASP DSOMM

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

<https://dsomm.owasp.org/>

Identification of the degree of the implementation



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

Технологии

Контроль ИБ артефактов,
зависимостей и образовУправление
артефактамиКонтроль использования
сторонних компонентов-
зависимостей
(библиотеки, образы и т.д.)

Защита окружения разработки

Защита
рабочих мест
разработчикаЗащита
секретовЗащит
Build-средыЗащита реестра
артефактовКонтроль
внесения
изменений
в исходный кодЗащита
конвейера
сборки

Контроль разрабатываемого ПО в части ИБ

Анализ ПО (development)

Статический
анализКомпозиционный
анализАнализ образов
контейнеровИдентификация
секретовКонтроль
безопасности
Dockerfile'ов

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

Динамический
анализ приложенийТестирование на проникновение
перед внедрением приложений
в продуктивФункциональное
ИБ-тестированиеКонтроль 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
приложенийКонтроль
сетевого
трафика
(L4-L7)Контроль выполняемых
процессов и их прав доступаАнализ инфраструктуры
и приложений на уязвимостиТестирование на
проникновение продуктивной
средыАнализ 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 угрозОпределение
требований ИБ,
предъявляемых к ПОКонтроль
выполнения
требований ИБРазработка стандартов
конфигураций

Управление ИБ-дефектами

Обработка дефектов ИБ

Консолидация дефектов ИБ

Оценка эффективности
процессов DSOУправление
набором
метрик ИБКонтроль
исполнения метрик
(как собираем и что
с ними делаем)Функциональные
ролиSecurity
ChampionsРазграничение
ролей процесса
DSO

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
разработчикаЗащита реестра
артефактовКонтроль внесения
изменений в исходный кодЗащита конвейера
сборкиРазработка
кода

Контроль разрабатываемого ПО в части ИБ

Анализ ПО (development)

Статический анализ

Композиционный анализ

Анализ образов
контейнеров

Идентификация секретов

Контроль безопасности
Dockerfile'овТестирование
(препрод)

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

Динамический анализ
приложенийКонтроль конфигураций –
Контроль безопасности манифестов (k8s, terraform и т.д.)Функциональное
ИБ-тестированиеТестирование 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 доступаАнализ инфраструктуры и приложений
на уязвимостиУправление контролем доступа к
средам работы приложенийТестирование на проникновение
(pentest) продуктивной средыАнализ событий информационной
безопасности

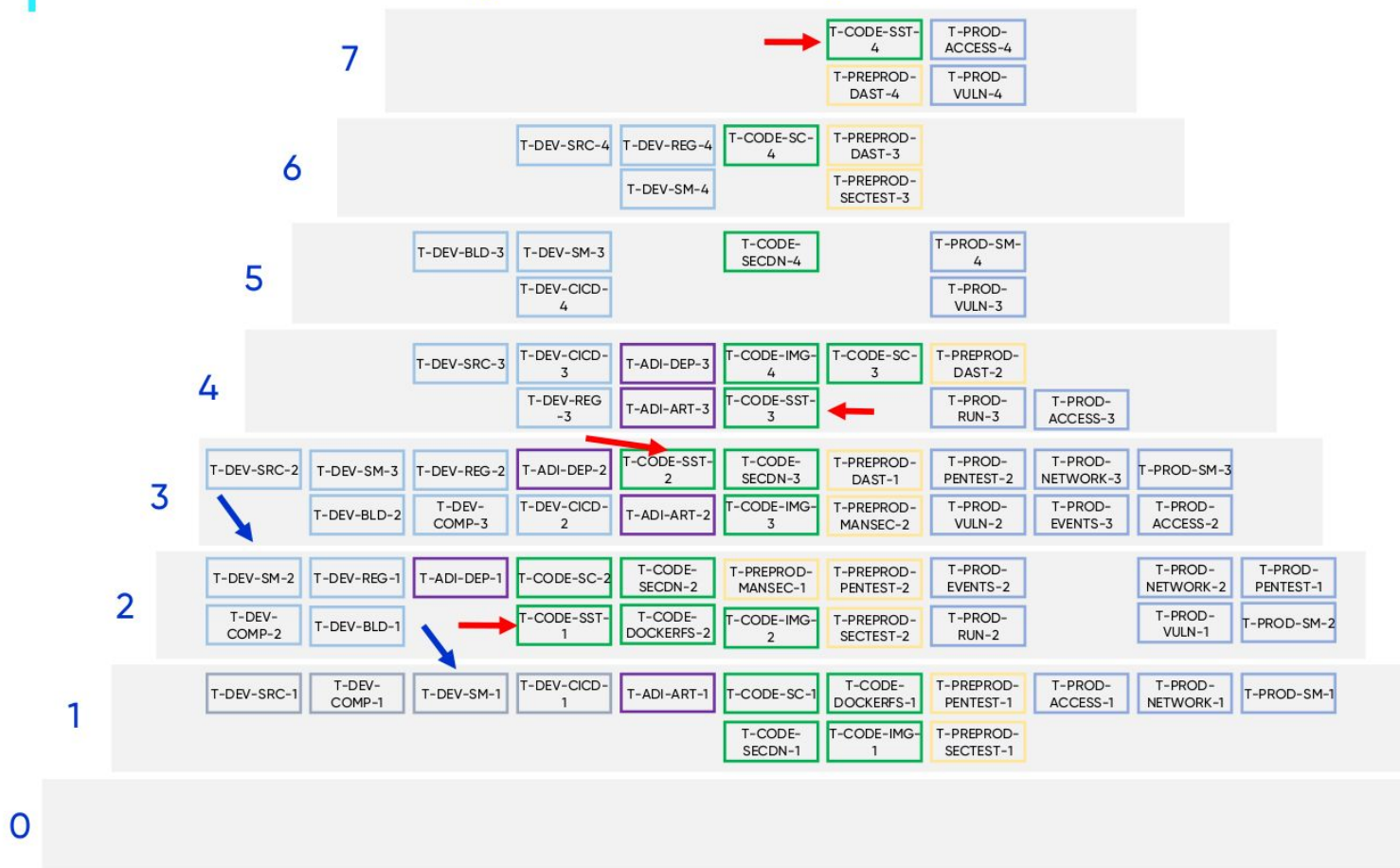
JDSOF: Таблица расчета

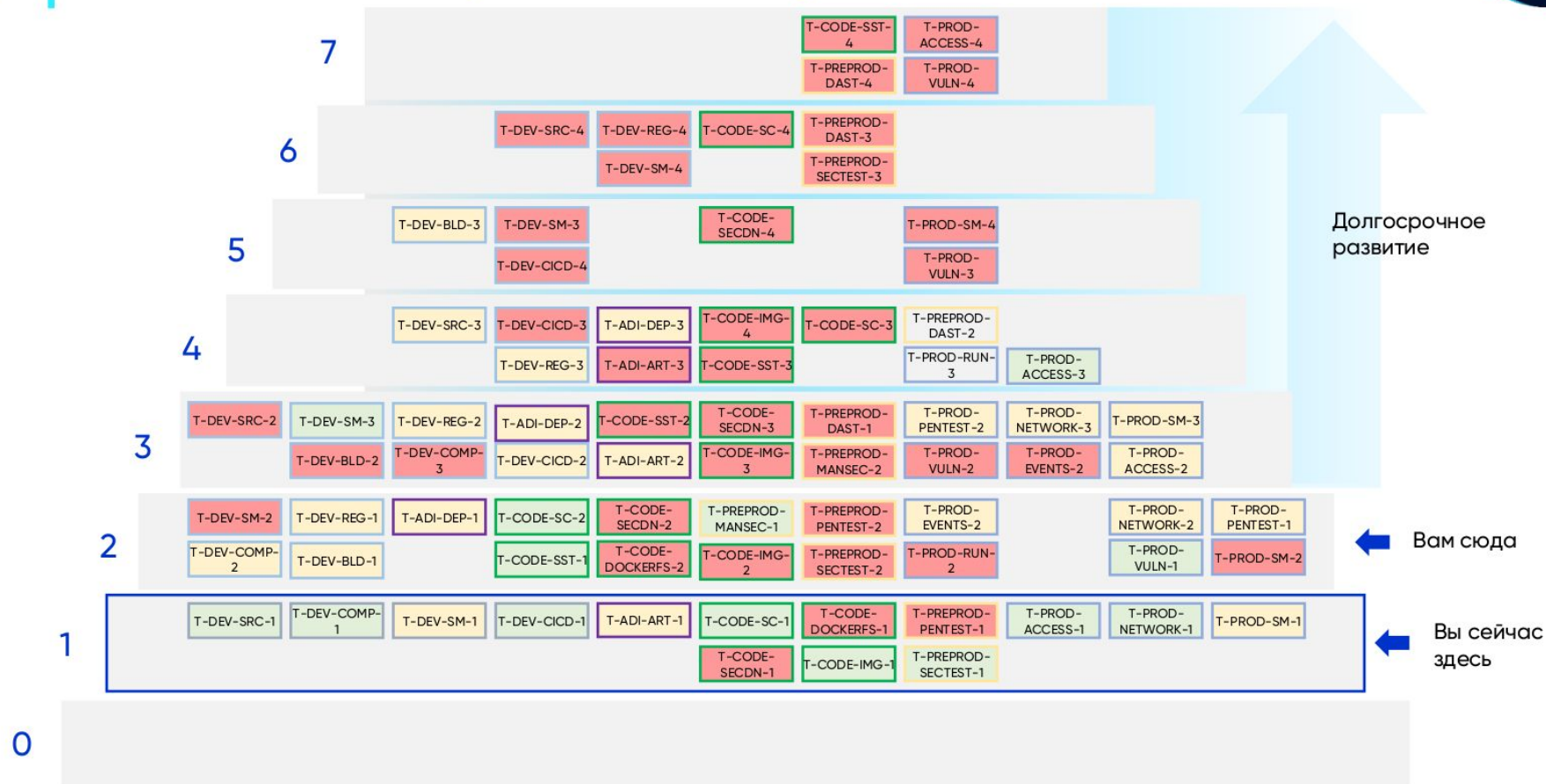
ГРУППА КРИТЕРИЕВ		LEVEL 0 UNINITIATED	LEVEL 1 BEGINNERS	LEVEL 2 INTERMEDIATE	LEVEL 3 ADVANCED	LEVEL 4 EXPERTS
Домен	Практика 1		Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n	Требование 1 Требование 2 ... Требование n
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

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%







Дорожная карта развития функции по результатам аудита JDSOF

Процесс

План-график реализации

Стоимость
развития, ₽Выгоды от реализации активностей
(верхнеуровневое описание)

Процесс	2023			Стоимость развития, ₽	Выгоды от реализации активностей (верхнеуровневое описание)
	2 квартал	3 квартал	4 квартал		
Pr1 Внедрение инструмента Secrets Detection	Pr1.1 Выбор инструмента	Pr1.2 Подготовка регламентов и документации Pr1.3 Внедрение инструмента	Pr1.4 Обучение	Р Р	• Внедрение практики детектирования секретов
Pr2 Внедрение инструмента Secrets Management	Pr2.1 Выбор инструмента	Pr2.2 Подготовка регламентов и документации	Pr2.3 Внедрение инструмента Pr2.4 Обучение	Р Р Р	• Внедрение практики управления секретами
Pr3 Внедрение инструмента контроля Dockerfiles	Pr3.1 Выбор инструмента	Pr3.2 Подготовка регламентов и документации Pr3.3 Внедрение инструмента	Pr3.4 Обучение	Р	Внедрение инструмента контроля Dockerfiles
Pr4 Внедрение инструмента контроля образов контейнеров	Pr4.1 Выбор инструмента	Pr4.2 Подготовка регламентов и документации Pr4.3 Внедрение инструмента	Pr.4 Обучение	Р	Внедрение инструмента контроля образов контейнеров
Pr5 Внедрение функционального тестирования ИБ		Pr5.1 Подготовка регламентов и документации		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>

