МИНОБРНАУКИ РОССИИ
Ярославский государственный университет им. П.Г. Демидова

Кафедра компьютерной безопасности и математических методов обработки информации

УТВЕРЖДАЮ

Декан математического факультета

_____ Нестеров П.Н.

20 июня 2023 г.

Рабочая программа дисциплины

Разработка безопасного программного обеспечения

Направление подготовки (специальности)
10.04.01 Информационная безопасность

Направленность (профиль)
«Управление информационной безопасностью»

Форма обучения очная

Программа рассмотрена
на заседании кафедры
от 14 апреля 2023 г., протокол № 8

Программа одобрена НМК
математического факультета
протокол № 9 от 3 мая 2023 г.

Ярославль

1. Цель освоения дисциплины.

Целью дисциплины является изучение принципов разработки безопасного программного обеспечения.

2. Место дисциплины в структуре образовательной программы.

Дисциплина «Защита программ и данных» относится к числу дисциплин базовой части профессионального цикла. Для лучшего усвоения данной дисциплины необходимо, чтобы студент владел знаниями, умениями и навыками, сформированными в процессе обучения в бакалавриате по направлениям 10.03.01, 11.03.02, 03.03.03, 11.03.01.

Знания и практические навыки, полученные в результате изучения дисциплины, используются в дисциплинах «Защищенные информационные системы», «Аудит информационной безопасности», «Разработка безопасного программного обеспечения», «Защита от вредоносного программного обеспечения», а также при разработке курсовых и дипломных работ.

3. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы.

Процесс изучения дисциплины направлен на формирование следующих элементов компетенций в соответствии с ФГОС ВО года для специальности ИБ, направленных на приобретение следующих знаний, умений, навыков и (или) опыта деятельности:

Формируемая компетенция (код и формулировка)	**Индикатор достижения компетенции (код и формулировка)**	**Перечень планируемых результатов обучения**
Профессиональные компетенции		
ПК-1 Способен разрабатывать математические модели систем обеспечения информационной безопасности, математически доказывать их соответствие выбранным политикам безопасности.	**И-ПК-1_5** Способен реализовывать механизмы безопасности в программном обеспечении.	**Знать:** − Механизмы безопасности, применяемые в ПО. − Математическое обоснование безопасности применяемых в ПО алгоритмов и моделей. **Уметь:** − Разрабатывать механизмы безопасности ПО. − Доказывать соответствие разработанных механизмов безопасности выбранным политикам безопасности. − Применять системное криптографическое API при

		разработке ПО. **Владеть навыками:** − Применения механизмов подписывания и проверки исполняемых файлов, библиотек и сборок. − Использования подготовленных выражений при исполнении SQL-запросов. − Использования безопасных HTML-фреймворков и санитайзеров HTML-верстки.
	И-ПК-1_6 Способен применять математический аппарат при доказательстве соответствия механизмов безопасности ПО выбранным политикам безопасности.	**Знать:** − Математическое обоснование безопасности применяемых в ПО алгоритмов и моделей. **Уметь:** − Доказывать соответствие механизмов безопасности ПО выбранным политикам безопасности.
ПК-2 Способен анализировать математические модели систем обеспечения информационной безопасности, а также проводить тестирование средств защиты информации на соответствие этим моделям.	**И-ПК-2_8** Способен проводить тестирование механизмов безопасности ПО на соответствие моделям обеспечения информационной безопасности.	**Уметь:** − Разрабатывать автотесты для проверок механизмов безопасности ПО. − Применять инструменты статического анализа безопасности ПО. − Применять инструменты динамического анализа безопасности ПО.

4. Объем, структура и содержание дисциплины.

Общая трудоемкость дисциплины составляет 4 зачетных единицы, 144 акад. часов.

№ п/п	Темы (разделы) дисциплины, их содержание	Семестр	Виды учебных занятий, включая самостоятельную работу студентов, и их трудоемкость (в академических часах)						Формы текущего контроля успеваемости Форма промежуточной аттестации (по семестрам)
			Контактная работа						
			лекции	практические	лабораторные	консультации	аттестационные испытания	самостоятельная работа	
1	Правила безопасной разработки.	3	4	2				6	Практические занятия
2	Безопасное программирование в C/C++.	3	6	2				6	Практические занятия
3	Безопасная разработка web-приложений.	3	6	2				7	Практические занятия
4	Применение криптографии в ПО.	3	6	4				7	Практические занятия
5	Анализ безопасности кода.	3	6	16				6	Практические занятия
6	Безопасное управление секретами.	3	2	4				6	Практические занятия
7	Применение механизмов подписи для проверки программных компонентов.	3	2	2				6	Практические занятия
						2	0,5	33,5	Экзамен
	Всего за 3 семестр		32	32		2	0,5	77,5	
	Всего		**32**	**32**		**2**	**0,5**	**77,5**	

Содержание разделов дисциплины:

Тема 1. Правила безопасной разработки.

1. Common Weakness Enumeration (CWE). CWE Top 25.
2. Стандарты безопасной разработки Software Engineering Institute Computer Emergency Response Team (SEI CERT).
3. Common Vulnerability and Exposures (CVE).
4. National Vulnerability Database (NVD).
5. Common Vulnerability Scoring System (CVSS).

6. Руководства по разработке и тестированию Open Web Application Security Project (OWASP). OWASP Top 10.

7. Общая проблема небезопасной архитектуры приложений и legacy-кода.

Тема 2. Безопасное программирование в C/C++.

1. Небезопасная работа с памятью. Обращение к невыделенной и освобожденной памяти. Разыменование нулевого указателя. Освобождение памяти, которая не была ранее выделена функцией управления памяти. Выделение памяти недостаточной для объекта.

2. Небезопасная работа с числами. Ошибки при переполнении и округлении.

3. Небезопасная работа с массивами.

4. Небезопасная работа со строками. Ошибки с нуль-терминированными строками.

5. Небезопасная работа с вводом-выводом. Ошибки форматной строки. Состязательные ситуации time-of-check, time-of-use (TOCTOU).

6. Небезопасное использование функции system.

7. Небезопасное использование сигналов.

8. Небезопасность необработки ошибок библиотек.

9. Небезопасная работа с интерфейсами операционных систем (ОС).

Тема 3. Безопасная разработка web-приложений.

1. Разработка безопасных механизмов идентификации и аутентификации пользователей. Сессионные куки. JWT.

2. Разработка безопасных механизмов авторизации. Ролевая модель доступа. Доступ к файлам. Доступ к URL и обработчикам HTTP-запросов.

3. CORS. Настройка доступа к API с авторизованных доменов.

4. Защита от инъекций. Подходы к экранированию пользовательских данных. Серверные и клиентские XSS санитайзеры.

5. Защита от XSS.

6. Использование уязвимых компонентов.

7. Небезопасная конфигурация. Оставление настроек по умолчанию. Оставление ненужных возможностей.

8. HTTP заголовки безопасности: Strict-Transport-Security, Content-Security-Policy, X-Frame-Options, X-Content-Type-Options, Referrer-Policy, Cache-Control, Clear-Site-Data, Feature-Policy.

9. Отсутствие проверки целостности ПО и данных. CI/CD pipeline. Атака на цепочку поставок. Использование неизвестного кода.

10. Отсутствие достаточного логирования и мониторинга.

11. Защита от подделки запросов на клиентской стороне. CSRF-токены. Флаг SameSite на куки.

12. Защита от подделки запросов на серверной стороне.

13. Тестирование web-приложений.

Тема 4. Применение криптографии в ПО.

1. Использование Linux Crypto API.

2. Использование криптографическое API Windows.

3. Использование сторонних библиотек.

4. Тестирование криптографических реализаций. Стандарт FIPS 140.

5. Управление ключами.

Тема 5. Анализ безопасности кода.

1. Ошибки параллельного программирования. Поиск состояния гонки с помощью ThreadSanitizer.

2. Статический анализ кода на безопасность.

3. Динамический анализ кода на безопасность.

Тема 6. Безопасное управление секретами.

1. Проблема создания, доставки и хранения секрета.

2. Системы управления секретами.

3. Проблема логирования и копирования секретов. Системы поиска секретов. DLP системы.

Тема 7. Применение механизмов подписи для проверки программных компонентов.

1. Подпись бинарных файлов на Windows с помощью SignTool.

2. Подпись сборок .NET.

3. Подпись сборок Java.

4. Проверка подписи исполняемых файлов, библиотек и сборок.

5. Механизм .NET Code-Access Security.

5. Образовательные технологии, используемые при осуществлении образовательного процесса по дисциплине.

В процессе обучения используются следующие образовательные технологии:

Вводная лекция – дает первое целостное представление о дисциплине и ориентирует студента в системе изучения данной дисциплины. Студенты знакомятся с назначением и задачами курса, его ролью и местом в системе учебных дисциплин и в системе подготовки в целом. Дается краткий обзор курса, история развития науки и практики, достижения в этой сфере, имена известных ученых, излагаются перспективные направления исследований. На этой лекции высказываются методические и организационные особенности работы в рамках

данной дисциплины, а также дается анализ рекомендуемой учебно-методической литературы.

Академическая лекция – последовательное изложение материала, осуществляемое преимущественно в виде монолога преподавателя. Требования к академической лекции: современный научный уровень и насыщенная информативность, убедительная аргументация, доступная и понятная речь, четкая структура и логика, наличие ярких примеров, научных доказательств, обоснований, фактов.

Практическое занятие – занятие, посвященное освоению конкретных умений и навыков и закреплению полученных на лекции знаний.

6. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине.

1) Операционные системы.

1. ОС Windows по программе Microsoft Azure Dev Tools for Teaching.
2. ОС Linux, свободно распространяемые дистрибутивы.

2) Гипервизоры и эмуляторы.

1. VirtualBox. Свободно распространяемое ПО. https://www.virtualbox.org/ (дата обращения 19.01.2022)

3) Среды разработки и SDK.

1. Visual Studio по программе Microsoft Azure Dev Tools for Teaching.
2. Visual Studio Code. Свободно распространяемое ПО. https://code.visualstudio.com/ (дата обращения 19.01.2022)
3. Windows SDK, как часть Windows по программе Microsoft Azure Dev Tools for Teaching. https://developer.microsoft.com/en-us/windows/downloads/windows-sdk/ (дата обращения 19.01.2022)
4. OpenJDK. Свободно распространяемое ПО. https://openjdk.java.net/ (дата обращения 19.01.2022)
5. ASP.NET. Свободно распространяемое ПО. https://dotnet.microsoft.com/en-us/apps/aspnet (дата обращения 19.01.2022)

4) Инструменты тестирования.

1. CodeQL. Бесплатное ПО для академических целей. https://github.com/github/codeql (дата обращения 19.01.2022)
2. ThreadSanitizer. Свободно распространяемое ПО. https://clang.llvm.org/docs/ThreadSanitizer.html (дата обращения 19.01.2022)
3. gitleaks. Свободно распространяемое ПО. https://github.com/zricethezav/gitleaks
4. CATS. Свободно распространяемое ПО. https://github.com/Endava/cats (дата обращения 19.01.2022)

5. american fuzzy lop. Свободно распространяемое ПО. https://github.com/google/AFL (дата обращения 19.01.2022)

6. sqlmap. Свободно распространяемое ПО. https://sqlmap.org/ (дата обращения 19.01.2022)

7. XSStrike. Свободно распространяемое ПО. https://github.com/s0md3v/XSStrike (дата обращения 19.01.2022)

8. Robot Framework. Свободно распространяемое ПО. https://robotframework.org/ (дата обращения 19.01.2022)

5) Системы управления секретами.

1. HashiCorp Vault Free. Свободно распространяемое ПО. https://github.com/hashicorp/vault (дата обращения 19.01.2022)

7. Перечень основной и дополнительной учебной литературы, ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины.

1) Основная литература.

1. Семь безопасных информационных технологий [Электронный ресурс] : учеб. / А.В. Барабанов [и др.]. — Электрон. дан. — Москва : ДМК Пресс, 2017. — 224 с. — 10 Режим доступа: https://e.lanbook.com/book/97352.

2. Казарин, О. В. Надежность и безопасность программного обеспечения : учебное пособие для вузов / О. В. Казарин, И. Б. Шубинский. — Москва : Издательство Юрайт, 2022. — 342 с. — (Высшее образование). — ISBN 978-5-534-05142-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/493262 (дата обращения: 28.01.2022)

2) Дополнительная литература.

1. Тобиас, К. Дневник охотника за ошибками. Путешествие через джунгли проблем безопасности программного обеспечения [Электронный ресурс] — Электрон. дан. — Москва : ДМК Пресс, 2013. — 240 с. — Режим доступа: https://e.lanbook.com/book/4812.

2. Ховард, М. 19 смертных грехов, угрожающих безопасности программ. Как не допустить типичных ошибок [Электронный ресурс] : учеб. пособие / М. Ховард, Д. Лебланк, Д. Виега. — Электрон. дан. — Москва : ДМК Пресс, 2009. — 288 с. — Режим доступа: https://e.lanbook.com/book/1118.

3) Ресурсы сети Интернет.

1. Журнал «Хакер» [Электронный ресурс] // URL: https://xakep.ru (дата обращения 19.01.2022)

2. NIST SP 800-57 Part 1 Rev. 5 [Электронный ресурс] // URL: https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-5/final (дата обращения 19.01.2022)

3. SEI CERT Coding Standards [Электронный ресурс] // URL: https://wiki.sei.cmu.edu/confluence/display/seccode (дата обращения 19.01.2022)

4. Common Weakness Enumeration [Электронный ресурс] // URL: https://cwe.mitre.org/ (дата обращения 19.01.2022)

5. Common Vulnerability and Exposures [Электронный ресурс] // URL: https://cve.mitre.org (дата обращения 19.01.2022)

6. National Vulnerability Database [Электронный ресурс] // URL: https://nvd.nist.gov/ (дата обращения 19.01.2022)

7. CVSS [Электронный ресурс] // URL: https://nvd.nist.gov/vuln-metrics/cvss (дата обращения 19.01.2022)

8. OWASP [Электронный ресурс] // URL: https://owasp.org/ (дата обращения 19.01.2022)

9. JSON Web Tokens, RFC 7519 [Электронный ресурс] // URL: https://jwt.io/ (дата обращения 19.01.2022)

10. Mozilla Developer Network [Электронный ресурс] // URL: https://developer.mozilla.org/en-US/ (дата обращения 19.01.2022)

11. Linux Kernel Crypto API [Электронный ресурс] // URL: https://www.kernel.org/doc/html/v4.18/crypto/index.html (дата обращения 19.01.2022)

12. Microsoft CryptoAPI [Электронный ресурс] // URL: https://docs.microsoft.com/en-us/windows/win32/seccrypto/cryptoapi-system-architecture (дата обращения 19.01.2022)

8. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине.

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине включает в свой состав специальные помещения:

− учебные аудитории для проведения занятий лекционного типа, оборудованные персональной компьютерной техникой с установленными средствами визуализации текстов в формате DOC/DOCX, PDF, F2B, файлов изображений, презентаций и мультимедийных файлов, а также – видеопроектором и жалюзи на окнах;

− учебные аудитории для проведения практических занятий: лаборатория программно-аппаратных средств обеспечения информационной безопасности;

− учебные аудитории для проведения групповых и индивидуальных консультаций,

− учебные аудитории для проведения текущего контроля и промежуточной аттестации;

− помещения для самостоятельной работы;

− помещения для хранения и профилактического обслуживания технических средств обучения.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети Интернет и обеспечением доступа в электронную информационно-образовательную среду организации.

Число посадочных мест в лекционной аудитории больше либо равно списочному составу потока, а в аудитории для практических и лабораторных занятий – списочному составу группы обучающихся.

Автор(ы):

Доцент кафедры компьютерной безопасности и математических методов обработки информации В.Н. Князев

Приложение №1 к рабочей программе дисциплины «Разработка безопасного программного обеспечения»

Фонд оценочных средств для проведения текущей и промежуточной аттестации студентов по дисциплине

1. Типовые контрольные задания и иные материалы, используемые в процессе текущего контроля успеваемости.

Оценка текущей успеваемости выставляется в зависимости от результатов самостоятельной работы студента и его работы на практических занятиях. Задания для самостоятельной работы выдаются учащимся на последнем часе лекционных занятий по теме. Оценка и обсуждение выполненных студентами заданий по самостоятельной работе производится на практическом занятии по данной теме и учитывается на ряду с результатами практических занятий при выставлении оценки текущей успеваемости.

Каждое практическое занятие рассчитано на 2 академических часа. На практическом занятии преподавателем с помощью доступных информационных технологий и собственных подготовленных материалов демонстрируется решение прикладных задач дисциплины. Студенты повторяют ход решения самостоятельно на отдельных устройствах.

Тема 1. Правила безопасной разработки.

1. Практическое занятие № 1. Классификация уязвимостей по CWE. Вычисление CVSS. Поиск CVE по продуктам. Применение контрмер из CWE и CVE.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания механизмов безопасности ПО.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения статического анализа безопасности ПО в плане поиска известных уязвимостей.

Тема 2. Безопасное программирование в C/C++.

1. Практическое занятие № 2. Разбор примеров SEI CERT C++.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части и способности применять механизмы безопасности ПО.

Тема 3. Безопасная разработка web-приложений.

1. Практическое занятие № 3. Разработка безопасного web-приложения на C# с применением фреймфорка ASP.NET.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания и способности применять механизмы безопасности ПО, в части владения навыками

использования подготовленных выражений при исполнении SQL-запросов и безопасных HTML-фреймворков и санитайзеров HTML-верстки.

Тема 4. Применение криптографии в ПО.

1. Практическое занятие № 4. Использование Linux Crypto API.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания и способности применять механизмы безопасности ПО, в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части умения доказывать соответствие разработанных механизмов безопасности выбранным политикам безопасности, в части способности применять системное криптографическое API при разработке ПО.

Проверка сформированности ПК-1, индикатор И-ПК-1_6 в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части способности доказывать соответствие механизмов безопасности ПО выбранным политикам безопасности.

2. Практическое занятие № 5. Использование криптографическое API Windows.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания и способности применять механизмы безопасности ПО, в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части умения доказывать соответствие разработанных механизмов безопасности выбранным политикам безопасности, в части способности применять системное криптографическое API при разработке ПО.

Проверка сформированности ПК-1, индикатор И-ПК-1_6 в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части способности доказывать соответствие механизмов безопасности ПО выбранным политикам безопасности.

Тема 5. Анализ безопасности кода.

1. Практическое занятие № 6. Статический анализ безопасности C++ с помощью CodeQL.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения статического анализа безопасности ПО.

2. Практическое занятие № 7. Статический анализ безопасности C# и Java с помощью CodeQL.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения статического анализа безопасности ПО.

3. Практическое занятие № 8. Фаззинг C++ с помощью AFL.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения динамического анализа безопасности ПО в плане фаззинга приложений на C++.

4. Практическое занятие № 9. Поиск состояний гонки с помощью ThreadSanitizer.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения динамического анализа безопасности ПО.

5. Практическое занятие № 10. Фаззинг OpenAPI с помощью CATS.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения динамического анализа безопасности ПО.

6. Практическое занятие № 11. Динамический анализ web-приложений на SQL инъекции с помощью sqlmap.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения динамического анализа безопасности ПО.

7. Практическое занятие № 12. Динамический анализ web-приложений на SQL инъекции с помощью XSStrike.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения динамического анализа безопасности ПО.

8. Практическое занятие № 13. Написание автотестов безопасности web-приложений с помощью Robot Framework.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности разрабатывать автотесты для проверок механизмов безопасности ПО.

Тема 6. Безопасное управление секретами.

1. Практическое занятие № 14. Настройка инфраструктуру распределения секретов на основе HashiCorp Vault.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания и способности применять механизмы безопасности ПО, в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части умения доказывать соответствие разработанных механизмов безопасности выбранным политикам безопасности, в части способности применять системное криптографическое API при разработке ПО.

Проверка сформированности ПК-1, индикатор И-ПК-1_6 в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части способности доказывать соответствие механизмов безопасности ПО выбранным политикам безопасности.

2. Практическое занятие № 15. Поиск секретов с помощью CodeQL и gitleaks.

Проверка сформированности ПК-2, индикатор И-ПК-2_8 в части способности применения статического анализа безопасности ПО.

Тема 7. Применение механизмов подписи для проверки программных компонентов.

1. Практическое занятие № 16. Подписание исполняемых файлов, библиотек и сборок. Проверка подписи. Настройка .NET CAS.

Проверка сформированности ПК-1, индикатор И-ПК-1_5 в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части умения доказывать соответствие разработанных механизмов безопасности выбранным политикам безопасности, в части владения навыкам применения механизмов подписывания и проверки исполняемых файлов, библиотек и сборок.

Проверка сформированности ПК-1, индикатор И-ПК-1_6 в части знания математического обоснования безопасности применяемых в ПО алгоритмов и моделей, в части способности доказывать соответствие механизмов безопасности ПО выбранным политикам безопасности.

9. Список вопросов и (или) заданий для проведения промежуточной аттестации.

Список вопросов к экзамену.

1. CWE: назначение и применение. Иерархии внутри CWE.
2. CWE Top 25.
3. SEI CERT C++ Coding Standard: правила и примеры.
4. SEI CERT Oracle Coding Standard for Java: правила и примеры.
5. CVE: назначение и применение.
6. NVD: назначение и применение.
7. CVSSv3: назначение и применение.
8. OWASP Web Security Testing Guide.
9. OWASP Top 10.
10. Общая проблема небезопасной архитектуры приложений и legacy-кода.
11. Безопасное программирование в C/C++. Небезопасная работа с памятью. Обращение к невыделенной и осовобожденной памяти. Разыменование нулевого указателя. Освобождение памяти, которая не была ранее выделена функцией управления памяти. Выделение памяти недостаточной для объекта. Методы противодействия.
12. Безопасное программирование в C/C++. Небезопасная работа с числами. Ошибки при переполнении и округлении. Методы противодействия.
13. Безопасное программирование в C/C++. Небезопасная работа с массивами. Методы противодействия.
14. Безопасное программирование в C/C++. Небезопасная работа со строками. Ошибки с нуль-терминированными строками. Методы противодействия.
15. Безопасное программирование в C/C++. Небезопасная работа с вводом-выводом. Ошибки форматной строки. Состязательные ситуации TOCTOU. Методы противодействия.

16. Безопасное программирование в C/C++. Небезопасное использование функции system. Методы противодействия.

17. Безопасное программирование в C/C++. Небезопасное использование сигналов. Методы противодействия.

18. Безопасное программирование в C/C++. Небезопасность необработки ошибок библиотек. Методы противодействия.

19. Безопасное программирование в C/C++. Небезопасная работа с интерфейсами ОС. Методы противодействия.

20. Безопасная разработка web-приложений. Разработка безопасных механизмов идентификации и аутентификации пользователей. Сессионные куки. JWT.

21. Безопасная разработка web-приложений. Разработка безопасных механизмов авторизации. Ролевая модель доступа. Доступ к файлам. Доступ к URL и обработчикам HTTP-запросов.

22. Безопасная разработка web-приложений. CORS. Настройка доступа к API с авторизованных доменов.

23. Безопасная разработка web-приложений. Защита от инъекций. Подходы к экранированию пользовательских данных. Серверные и клиентские XSS санитайзеры.

24. Безопасная разработка web-приложений. Защита от XSS.

25. Безопасная разработка web-приложений. Использование уязвимых компонентов. Методы противодействия.

26. Безопасная разработка web-приложений. Небезопасная конфигурация. Оставление настроек по умолчанию. Оставление ненужных возможностей. Методы противодействия.

27. Безопасная разработка web-приложений. HTTP заголовки безопасности: Strict-Transport-Security, Content-Security-Policy, X-Frame-Options, X-Content-Type-Options, Referrer-Policy, Cache-Control, Clear-Site-Data, Feature-Policy.

28. Безопасная разработка web-приложений. Отсутствие проверки целостности ПО и данных. CI/CD pipeline. Атака на цепочку поставок. Использование неизвестного кода. Методы противодействия.

29. Безопасная разработка web-приложений. Отсутствие достаточного логирования и мониторинга. Методы противодействия.

30. Безопасная разработка web-приложений. Защита от подделки запросов на клиентской стороне. CSRF-токены. Флаг SameSite на куки.

31. Безопасная разработка web-приложений. Защита от подделки запросов на серверной стороне.

32. Тестирование web-приложений.

33. Использование Linux Crypto API.

34. Использование криптографическое API Windows.

35. Тестирование криптографических реализаций. Стандарт FIPS 140.

36. Управление ключами. Рекомендации NIST о ключевом расписании: NIST SP 800-57.

37. Ошибки параллельного программирования. Состояние гонки и его обнаружение.

38. Статический анализ кода на безопасность.

39. Динамический анализ кода на безопасность.

40. Проблема создания, доставки и хранения секрета. Системы управления секретами.

41. Проблема логирования и копирования секретов. Системы поиска секретов. DLP системы.

42. Подпись бинарных файлов на Windows с помощью SignTool. Проверка подлинности бинарных файлов в разных сценариях.

43. Подпись сборок .NET. Проверка подлинности сборок .NET в разных сценариях.

44. Подпись сборок Java. Проверка подлинности сборок Java в разных сценариях.

45. Проверка подписи исполняемых файлов, библиотек и сборок.

46. Механизм .NET Code-Access Security.

Правила выставления оценки на экзамене.

В экзаменационные билет включается два теоретических вопроса. На подготовку к ответу дается не менее 1часа.

По итогам экзамена выставляется одна из оценок: «отлично», «хорошо», «удовлетворительно» или «неудовлетворительно».

Оценка «Отлично» выставляется студенту, который демонстрирует глубокое понимание теоретической части дисциплины, полные знания об инструментах и техниках их применения для разработки безопасного ПО. Осуществляет межпредметные связи. Дает развернутые, полные и четкие ответы на вопросы экзаменационного билета и дополнительные вопросы, соблюдает логическую последовательность при изложении материала. Грамотно использует терминологию дисциплины.

Оценка «Хорошо» выставляется студенту, ответ которого на экзамене в целом соответствуют указанным выше критериям, но отличается меньшей обстоятельностью, глубиной, обоснованностью и полнотой. В ответе имеют место отдельные неточности (несущественные ошибки), которые исправляются самим студентом после дополнительных и (или) уточняющих вопросов экзаменатора.

Оценка «Удовлетворительно» выставляется студенту, который демонстрирует поверхностное понимание теории, дает недостаточно полные и последовательные ответы на вопросы экзаменационного билета и дополнительные вопросы, но при этом демонстрирует умение выделить существенные и несущественные признаки и установить причинно-следственные связи. Студент знает и правильно применяет терминологию дисциплины, но допускает ошибки в определении и раскрытии некоторых основных понятий. При аргументации ответа студент не обосновывает свои суждения. На часть дополнительных вопросов студент затрудняется дать ответ или дает неверные ответы. Имеет существенные пробелы в знаниях об инструментах и техниках их применения для разработки безопасного ПО.

Оценка «Неудовлетворительно» выставляется студенту, который демонстрирует разрозненные, бессистемные знания; беспорядочно и неуверенно излагает материал. Не умеет выделять главное и второстепенное, не умеет соединять теоретические положения с практикой, не устанавливает межпредметные связи. Дает неполные ответы, логика и последовательность изложения которых имеют существенные и принципиальные нарушения, в ответах отсутствуют выводы. Дополнительные и уточняющие вопросы экзаменатора не приводят к коррекции ответов студента. На основную часть дополнительных вопросов студент затрудняется дать ответ или дает неверные ответы. Выставляется студенту, не показывающему знаний об инструментарии дисциплины.

Оценка «Неудовлетворительно» выставляется также студенту, который взял экзаменационный билет, но отвечать отказался.

Приложение № 2 к рабочей программе дисциплины «Разработка безопасного программного обеспечения»

Методические указания для студентов по освоению дисциплины

В изложении учебного материала по дисциплине «Разработка безопасного программного обеспечения» одинаковое время занимают теоретические и практические занятия. Уклон дисциплины в целом исключительно практический. Теоретические занятия необходимы для освоения основ предмета и построения межпредметных связей. Также на теоретических занятиях обсуждаются конкретные задачи разработки безопасного ПО и то, в каком объеме, с каким качеством и за какое время разные подходы к построению процесса разработки могут привести к цели выпуска ПО без ошибок безопасности. В таких обсуждениях отдельное внимание уделяется сравнительному анализу инструментов безопасной разработки и тестирования ПО на безопасность.

В процессе изучения дисциплины, рекомендуется регулярное повторение пройденного лекционного материала. Материал, изложенный на лекциях, представленный в предлагаемой учебной литературе, необходимо дома еще раз прорабатывать и, при необходимости, дополнять информацией, полученной на консультациях, практических занятиях и из рекомендованных ресурсов сети Интернет.

Для наилучшего результата изучения дисциплины рекомендуется повторять условия практических занятий на личных ПК с тем, чтобы попробовать применить пройденные на практических занятиях техники на новых примерах. При такой работе всегда неизбежно студент сталкивается с трудностями и у него появляются вопросы, которые на практическом занятии не пришли в голову. Эти вопрос нужно адресовать преподавателю на следующих занятиях, для ответов на вопросы преподаватель выделяет время на каждом занятии. Решение проблем, с которыми студент столкнулся сам, формирует у студента наиболее полные компетенции.

В конце семестра изучения дисциплины студенты сдают экзамен. Экзамен принимается по экзаменационным билетам, каждый из которых включает в себя два теоретических вопроса. Предусмотрена групповая консультация.