Минобрнауки России

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой

Сирота Александр Анатольевич

Кафедра технологий обработки и защиты информации

23.04.2024

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Б1.В.05 Анализ уязвимостей программного обеспечения

1. Код и наименование направления подготовки/специальности:

10.05.01 Компьютерная безопасность

2. Профиль подготовки/специализация:

Анализ безопасности компьютерных систем

3. Квалификация (степень) выпускника:

Специалитет

4. Форма обучения:

Очная

5. Кафедра, отвечающая за реализацию дисциплины:

Кафедра технологий обработки и защиты информации

6. Составители программы:

Дрюченко Михаил Анатольевич, к.т.н., доцент

7. Рекомендована:

протокол №5 от 05.03.2024

8. Учебный год:

2028-2029

9. Цели и задачи учебной дисциплины:

Цель дисциплины – ознакомление студентов с теоретическими и практическими аспектами анализа уязвимостей и общими принципами защиты программного обеспечения (ПО) для повышения безопасности разработки и эксплуатации информационных систем различного назначения.

Основные задачи дисциплины: ознакомление студентов с причинами возникновения и принципами эксплуатации уязвимостей в программном коде, изучение практических примеров уязвимостей в программном коде; изучение принципов анализа кода, внутреннего представления программы для анализа, ознакомление с принципами работы статистических и динамических анализаторов кода; изучение принципов создания безопасного ПО и современных методов защиты исходных и байт кодов программ; овладение практическими навыками формирования комплекса мер для повышения качества разработки ПО.

10. Место учебной дисциплины в структуре ООП:

Блок обязательные дисциплины вариативной части. Для успешного освоения дисциплины необходимы входные знания в области устройства ЭВМ и операционных систем, теориикомпиляторов, информатики и математических основ криптографии.

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями выпускников) и индикаторами их достижения:

Код и название компетенции	Код и название индикатора компетенции	Знания, умения, навыки
ПК-3 Способен проводить анализ безопасности программных средств в компьютерных системах	ПК-3.1 знает основные типы уязвимостей программного обеспечения и возможные пути их устранения	Знает основные виды уязвимостей ПО, принципы работы средств статического и динамического анализа кода, методы устранения уязвимостей. Умеет применять на практике полученные знания и навыки для проверки работоспособности ПО и его анализа на наличие уязвимостей (экспертиза исходного кода, статический и динамический анализ, файззинг-тестирование). Владеет практическими навыками анализа исходного кода на предмет наличия уязвимостей, навыками использования специализированных утилит статического и динамического анализа кода.
ПК-3 Способен проводить анализ безопасности программных средств в компьютерных системах	ПК-3.3 умеет анализировать программные средства на наличия уязвимостей	Знает известные методы анализа ПО на наличие уязвимостей, методы статического и динамического анализа программ, методы проведения экспертизы исходного кода. Умеет применять на практике полученные знания и навыки для анализа ПО на наличие уязвимостей. Владеет специализированными инструментами и практическими навыками анализа ПО на наличие уязвимостей.

12. Объем дисциплины в зачетных единицах/час:

3/108

Форма промежуточной аттестации:

Зачет с оценкой, контрольная работа

13. Трудоемкость по видам учебной работы

Вид учебной работы	Семестр 10	Всего
Аудиторные занятия	40	40
Лекционные занятия	16	16
Практические занятия		0
Лабораторные занятия	24	24
Самостоятельная работа	68	68
Курсовая работа		0
Промежуточная аттестация	0	0
Часы на контроль		0
Всего	108	108

13.1. Содержание дисциплины

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
1. Лекции			
1.1	Теоретические аспекты возникновения уязвимостей	1. Понятие и классификация уязвимостей. 2. Причины возникновения уязвимостей в программном коде и принципы их эксплуатации. 3. Уязвимости переполнения буфера в стеке. 4. Уязвимости переполнения буфера в куче. 5. Методы обнаружения и предотвращения переполнения буфера. 6. Уязвимость форматной строки. 7. Уязвимость переполнения целых чисел. 8. Эксплойты.	Создан онлан электронный курс, размещены материалы к лекции. Размещены индивидуальные задания для выполнения лабораторных работ.

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
1.2	Практические аспекты анализа уязвимостей	9. Практические примеры уязвимостей в программном коде. 10. Типовые сценарии выявления уязвимостей в программном коде. 11. Статические и динамические анализаторы кода. 12. Тестирование по принципу «белого ящика». 13. Файззингтестирование. 14. Повышение качества разработки ПО при использовании специа-лизированных программных средств.	Создан онлан электронный курс, размещены материалы к лекции. Размещены индивидуальные задания для выполнения лабораторных работ.
1.3	Методы защиты программного обеспечения	15. Принципы создания безопасного ПО. 16. Современные методы защиты ПО от взлома. 17. Технические меры защиты ПО. 18. Защита кода от анализа. 19. Принципы работы обфускаторов исходных и байткодов.	Создан онлан электронный курс, размещены материалы к лекции. Размещены индивидуальные задания для выполнения лабораторных работ.
2. Практические занятия			
2.1	нет		
3. Лабораторные работы			

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
3.1	Теоретические аспекты возникновения уязвимостей	1. Изучение принципов действия атаки переполнения буфера, реализация на практике модели атаки переполнения буфера в стеке и куче. 2. Исследование уязвимостей форматной строкой, реализация на практике модели атаки с использованием данной уязвимости. 3. Исследование уязвимости переполнения целых.	Размещены индивидуальные задания для выполнения лабораторных работ.
3.2	Практические аспекты анализа уязвимостей	4. Изучение принципов работы статических анализаторов исходного кода. 5. Изучение принципов динамического анализа кода.	Размещены индивидуальные задания для выполнения лабораторных работ.

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела)	Лекционные занятия	Практические занятия	Лабораторные занятия	Самостоятельная работа	Всего
1	Теоретические аспекты возникновения уязвимостей	4		6	20	30
2	Практические аспекты анализа уязвимостей	3		14	36	53
3	Методы защиты программного обеспечения	3		6	16	25
		10	0	26	72	108

14. Методические указания для обучающихся по освоению дисциплины

1) При изучении дисциплины рекомендуется использовать следующие средства:

- рекомендуемую основную и дополнительную литературу;
- методические указания и пособия;
- контрольные задания для закрепления теоретического материала;
- электронные версии учебников и методических указаний для выполнения лабораторно - практических работ (при необходимости материалы рассылаются по электронной почте).

2) Для максимального усвоения дисциплины рекомендуется проведение письменного опроса (тестирование, решение задач) студентов по материалам лекций и практических работ. Подборка вопросов для тестирования осуществляется на основе изученного теоретического материала. Такой подход позволяет повысить мотивацию студентов при конспектировании лекционного материала.

3) При проведении лабораторных занятий обеспечивается максимальная степень соответствия с материалом лекционных занятий и осуществляется экспериментальная проверка методов, алгоритмов и технологий обработки информации, излагаемых в рамках лекций.

4) При использовании дистанционных образовательных технологий и электронного обучения выполнять все указания преподавателей, вовремя подключаться к online занятиям, ответственно подходить к заданиям для самостоятельной работы.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

№ п/п	Источник
1	Страуструп, Бьерн. Язык программирования C++. Специальное издание = The C++ programming language. Special edition. / Бьерн Страуструп ; пер. с англ. под ред. Н.Н. Мартынова .— Москва : Бином, 2015 .— 1135 с. : ил .— Предм. указ.: с.1117-1135 .— ISBN 978-5-7989-0425-9 .— ISBN 0-201-70073-5.

б) дополнительная литература:

№ п/п	Источник
1	Щербаков, Андрей Юрьевич. Современная компьютерная безопасность. Теоретические основы. Практические аспекты : учебное пособие для студ. вузов / А.Ю. Щербаков .— М. : Кн. мир, 2009 .— 351, [1] с. : ил., табл. — (Высшая школа) .— Библиогр.: с.350-351 .— ISBN 978-5-8041-0378-2.
2	Хогланд Г. Взлом программного обеспечения : Анализ и использование кода / Г. Хогланд, Г. Мак-Гроу. – М. : Вильямс, 2005. - 400 с.
3	Козиол Дж. Искусство взлома и защиты систем / Дж. Козиол, Д. Личфилд, Д. Эйтэл ; редакторы, переводчики, составители: Е. Матвеев. – СПб [и др.] : Питер, 2006. – 416 с.
4	Ховард М. 19 смертных грехов, угрожающих безопасности программ. Как не допустить типичных ошибок : пер. с англ. / М. Ховард, Д. Лебланк, Д. Виега. – М. : ДМК Пресс, 2006. – 287 с.

в) информационные электронно-образовательные ресурсы:

№ п/п	Источник
1	Электронный каталог Научной библиотеки Воронежского государственного университета. – (http // www.lib.vsu.ru/).
2	Образовательный портал «Электронный университет ВГУ».– (https://edu.vsu.ru/)
3	ЭБС Лань, Лицензионный договор №3010, (с 01/03/2024 по 28.02.2025) 06/02 24 от 13.02.2024 (с дополнительным соглашением №1 от 14.03.2024)
4	ЭБС «Университетская библиотека online» (Контракт №3010 06/11 23 от 26.12.2023 (с 26.12.2023 по 25.12.2024)
5	ЭБС «Консультант студента» – Лицензионный договор №980КС/12-2023 / 3010-06/01-24 от 24.01.2024 с 24.01.2024 по 11. 01.2025)
6	Электронная библиотека ВГУ, Договор №ДС-208 от 01.02.2021 с ООО «ЦКБ «БИБКОМ» и ООО «Агентство «Книга-Сервис» о создании Электронной библиотеки ВГУ, (с 01.02.2021 по 31.01.2027)
7	БС BOOK.ru, Договор №3010 15/983 23 от 20.12.2023, (с 01.02.2024 по 31.01.2025)

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1	Щербаков, Андрей Юрьевич. Современная компьютерная безопасность. Теоретические основы. Практические аспекты : учебное пособие для студ. вузов / А.Ю. Щербаков .— М. : Кн. мир, 2009 .— 351, [1] с. : ил., табл. — (Высшая школа) .— Библиогр.: с.350-351 .— ISBN 978-5-8041-0378-2.

17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ), электронное обучение (ЭО), смешанное обучение):

Для реализации учебного процесса используется: ОС Windows v.7, 8, 10, GNU/Linux (Ubuntu), Virtual Box.

При проведении занятий в дистанционном режиме обучения используются технические и информационные ресурсы Образовательного портала "Электронный университет ВГУ" (https:\\edu.vsu.ru), базирующегося на системе дистанционного обучения Moodle, развернутой в университете.

18. Материально-техническое обеспечение дисциплины:

1) 394018, г. Воронеж, площадь Университетская, д. 1, корпус 1а, ауд. 479

Учебная аудитория: специализированная мебель, компьютер преподавателя i5-8400-2,8ГГц, монитор с ЖК 19’’, мультимедийный проектор, экран

ПО: ОС Windows v.7, 8, 10, Набор утилит (архиваторы, файл-менеджеры), LibreOffice v.5-7, Foxit PDF Reader.

2) 394018, г. Воронеж, площадь Университетская, д. 1, корпус 1а, аудитория 292

Учебная аудитория: специализированная мебель, компьютер преподавателя Pentium-G3420-3,2ГГц, монитор с ЖК 17", мультимедийный проектор, экран. Система для видеоконференций Logitech ConferenceCam Group и ноутбук 15.6’’ FHD Lenovo V155-15API

ПО: ОС Windows v.7, 8, 10, GNU/Linux (Ubuntu), Virtual Box, Набор утилит (архиваторы, файл-менеджеры), LibreOffice v.5-7, Foxit PDF Reader.

3) 394018, г. Воронеж, площадь Университетская, д. 1, корпус 1а, ауд. 380

Учебная аудитория: специализированная мебель, компьютер преподавателя i3-3240-3,4ГГц, монитор с ЖК 17'', мультимедийный проектор, экран

ПО: ОС Windows v.7, 8, 10, GNU/Linux (Ubuntu), Virtual Box, Набор утилит (архиваторы, файл-менеджеры), LibreOffice v.5-7, Foxit PDF Reader.

4) 394018, г. Воронеж, площадь Университетская, д. 1, корпус 1а, аудитория 290

Учебная аудитория: специализированная мебель, персональные компьютеры на базе i7-7800x-4ГГц, мониторы ЖК 27'' (12 шт.), мультимедийный проектор, экран.

Лабораторное оборудование искусственного интеллекта: рабочие места – персональные компьютеры на базе i7-7800x-4ГГц, мониторы ЖК 27" (12 шт.); модули АО НПЦ «ЭЛВИС»: процессорный Салют-ЭЛ24ПМ2 (9 шт.), отладочный Салют-ЭЛ24ОМ1 (9 шт.), эмулятор MC-USB-JTAG (9 шт.).

Лабораторное оборудование электроники, электротехники и схемотехники: рабочие места – персональные компьютеры на базе i7-7800x-4ГГц, мониторы ЖК 27" (12 шт.); стенд для практических занятий по электрическим цепям (KL-100); стенд для изучения аналоговых электрических схем (KL-200); стенд для изучения цифровых схем (KL-300).

ПО: ОС Windows v.7, 8, 10, GNU/Linux (Ubuntu), Virtual Box, Набор утилит (архиваторы, файл-менеджеры), LibreOffice v.5-7, Foxit PDF Reader.

19. Оценочные средства для проведения текущей и промежуточной аттестаций

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Разделы дисциплины (модули)	Код компетенции	Код индикатора	Оценочные средства для текущей аттестации
1	Разделы 1-3 Теоретические аспекты возникновения уязвимостей. Практические аспекты анализа уязвимостей. Методы защиты программного обеспечения.	ПК-3	ПК-3.1	Контрольная работа по соответствующим разделам. Лабораторные работы 1-5
2	Разделы 1-3 Теоретические аспекты возникновения уязвимостей. Практические аспекты анализа уязвимостей. Методы защиты программного обеспечения.	ПК-3	ПК-3.3	Контрольная работа по соответствующим разделам. Лабораторные работы 1-5

Промежуточная аттестация

Форма контроля - Зачет с оценкой, контрольная работа

Оценочные средства для промежуточной аттестации: перечень вопросов, практическое задание

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Текущий контроль успеваемости по дисциплине осуществляется с помощью следующих оценочных

средств:

Устный опрос на практических занятиях

Контрольная работа по теоретической части курса

Лабораторные работы

Примерный перечень применяемых оценочных средств

№ п/п	Наименование оценочного средства	Представление оценочного средства в фонде	Критерии оценки
1	Устный опрос на практических занятиях	Вопросы по темам/разделам дисциплины	Правильный ответ – зачтено, неправильный или принципиально неточный ответ - не зачтено
2	Контрольная работа по разделам дисциплины	Теоретические вопросы по темам/разделам дисциплины	Шкала оценивания соответствует приведенной ниже
3	Лабораторная работа	Содержит 5 лабораторных заданий, предусматривающие разработку и исследование программ, содержащих типовые классы уязвимостей	При успешном выполнении работ в течение семестра фиксируется возможность оценивания только теоретической части дисциплины в ходе промежуточной аттестации, в противном случае проверка задания по лабораторным работам выносится на зачет.

Пример задания для выполнения лабораторной работы

Лабораторная работа №1

«Исследование атаки переполнения буфера»

Цель работы: изучение принципов действия атаки переполнения буфера. Реализация на практике модели атаки переполнения буфера в стеке.

Форма контроля: *отчет в электронном виде*

Количество отведенных аудиторных часов: *4*

Задание: На языке Си написать следующие программы:

- уязвимую программу, подверженную переполнению буфера в стеке;
- программу, защищенную от переполнения;
- программу, реализующую атаку переполнения буфера;
- программу эксплойт.

Примеры контрольных вопросов:

1. Что такое уязвимости?
2. Каковы причины возникновения уязвимостей в программном коде?

3. Как эксплуатируются уязвимости?

Пример заданий теста по разделам дисциплины

В приведенных фрагментах кода найти и исправить ошибки (потенциальные уязвимости)

1	void encryptData(char *str)
	{
	  char pwd[64];
	  if(getPassword(pwd, sizeof(pwd)))
	  ...
	  ZeroMemory(pwd, sizeof(pwd));
	}	

2	int main(int argc, char *argv[])
	{
	  char login[100], pwd[100];
	  int pwd_len;
	
	  strcpy(login, argv[1]);
	  strcpy(pwd, argv[2]);
	  pwd_len = atoi(argv[3]);
	
	  // флаг, дающий права администратора
	  int admin = 0;
	  char orig_pwd[100] = "123456";
	
	  if(pwd_len < 1) pwd_len = 0;
	  pwd_len++;
	  if(login = "admin")
	  {
	  admin = 1;
	  for(i=0; i<= pwd_len; i++)
	  {
	  if((pwd[i])!=orig_pwd[i])admin=0;
	  }
	  }
	  setStastus(admin);
	}	

3	bool finished = false;
	char *buf = (char*)malloc(BUF_SIZE);
	...
	if(finished)
	  free(buf);
	...
	free(buf);	

4	char *buf1, *buf2;
	buf1 = (char*)malloc(size);
	...
	buf2 = (char*)malloc(size);
	strncpy(buf2, buf1, size);	

	...	

Приведённые ниже задания рекомендуется использовать при проведении диагностических работ для оценки остаточных знаний по дисциплине.

Компетенция ПК-3.1

Задания закрытого типа

1. Причины возникновения уязвимостей (возможен множественный выбор)

а) ошибки при проектировании и разработке программного и программно-аппаратного обеспечения;

б) несанкционированное внедрение и использование неучтенных программ с последующим необоснованным расходованием ресурсов;

в) несанкционированные неумышленные действия пользователей;

г) сбои в работе аппаратного и программного обеспечения (вызванные сбоями в электропитании, выходом из строя аппаратных элементов в результате старения и снижения надежности, внешними воздействиями электромагнитных полей технических устройств и др.)

2. Ошибки в программе, при которых компиляция завершается успешно, при пробных запусках программа ведет себя нормально, однако при анализе результата выясняется, что он неверный. Для их устранения необходимо «вручную» анализировать алгоритм

а) синтаксические;

б) времени выполнения;

в) алгоритмические;

г) препроцессорные

3. К потенциальным уязвимостям приложений, использующих криптографию, можно отнести (возможен множественный выбор):

а) использование функций диверсификации ключей на основе строки пароля вместо генераторов случайных чисел при создании секретных ключей;

б) отсутствие случайного вектора инициализации IV (задание нулевого вектора) при шифровании в режиме сцепления блоков CBC;

в) генерация хеш паролей без использования «соли»;

г) раздельное хранение ключа шифрования данных K_D и ключа шифрования ключей K_K, предназначенного для шифрования K_D;

д) шифрование паролей вместо хеширования при сохранении в БД;

е) использование предсказуемых начальных параметров для инициализации начального состояния ГПСЧП.

4. Преобразования управляющей логики включает (возможен множественный выбор)

а) замещение инструкций на аналогичные;

б) преобразования агрегации (изменяют порядок выполнения логически связанной последовательности операций и объединяют последовательности операций, которые связанными не являются);

в) трансформации выполнения операций;

г) шифрование строковых констант

5. Обфускация встроенных типов данных включает (возможен множественный выбор)

а) разделение переменных на две или несколько переменных;

б) конвертацию статических данных;
в) модификацию связей наследования;
г) склеивание скалярных переменных

6. Вариант тестирование ПО, предполагающий наличие доступа к любым ресурсам, в первую очередь к исходным кодам, а также техническому заданию и всевозможной документации

а) тестирование по принципу «белого ящика»;
б) тестирование по принципу «черного ящика»;
в) тестирование по принципу «серого ящика»;

7. Проблемы, которые могут быть обнаружены статическими анализаторами кода (возможен множественный выбор):

а) уязвимости форматных строк;
б) модификации кода, приводящие к возникновению уязвимостей, возникающие в результате компиляторной оптимизации;
в) уязвимости переполнения буфера;
г) использование неинициализированных данных;
д) ошибки в многопоточном коде.

8. Вариант тестирования ПО, для которого необходима лишь возможность взаимодействия с ПО, но не требуется подробная информация о внутреннем устройстве программы

а) тестирование по принципу «белого ящика»;
б) тестирование по принципу «черного ящика»;
в) тестирование по принципу «серого ящика»;

9. Вариант тестирования ПО, при котором в распоряжении специалиста находится исполняемый файл приложения и, возможно, некая базовая документация

а) тестирование по принципу «белого ящика»;
б) тестирование по принципу «черного ящика»;
в) тестирование по принципу «серого ящика»;

10. Верно ли утверждение: "Во время трансляции исходного кода в процессорные инструкции, компилятор может вносить в структуру программы значительные изменения, поэтому фрагменты анализируемого исходного кода могут не в полной мере соответствовать тому, как выполняется программа"

а) да;
б) нет

11. При статическом анализе кода обеспечивается (возможен множественный выбор):

а) полное покрытие кода;
б) существует зависимость от используемого компилятора и среды, в которой будет выполняться скомпилированная программа;
в) эффективная диагностика утечек памяти и параллельных ошибок;
г) эффективное обнаружение опечаток и последствий использования Copy-Paste.

12. При динамическом анализе кода обеспечивается (возможен множественный выбор):

а) полное покрытие кода;
б) эффективное обнаружение сложных уязвимостей, таких, которые занимают несколько этапов чтобы поместить программу в определенное состояние и из него вызвать ошибку;
в) возможность диагностики утечек памяти и параллельных ошибок;
г) возможность отслеживания производительности кода.

13. Этапы фаззинга включают (возможен множественный выбор):

а) декомпиляцию тестируемой программы;

б) определение вводимых значений;
в) формирование входных «некорректных» данных;
г) исполнение «некорректных» данных;
д) мониторинг исключений;

14. Метод фаззинга, формирующий тестовые наборы данных на основе коллекции имеющихся (корректно сформированных входных данных), путем искажения последних (искажаются случайные байты или строки)

а) полностью случайное тестирование;
б) мутирующее тестирование;
в) порождающее тестирование

15. Метод фаззинга, не предполагающий наличия информации о синтаксической структуре входных данных, при котором формируются массивы псевдослучайных данных и передаются исследуемой программе.

а) полностью случайное тестирование;
б) мутирующее тестирование;
в) порождающее тестирование

16. Метод фаззинга, предполагающий формирование тестовых наборов на основе предварительного изучения спецификаций протоколов или форматов файлов, создания грамматик с указанием в них переменных и статических данных, а также динамически вычисляемых величин

а) полностью случайное тестирование;
б) мутирующее тестирование;
в) порождающее тестирование

17. Фаззеры, предназначенные для выявления уязвимостей типа SQL-инъекций или межсайтового скриптинга.

а) фаззеры сетевых протоколов;
б) фаззеры веб-приложений;
в) фаззеры файловых форматов;
г) фаззеры командной строки;

18. Выберите неверные рекомендации по использованию криптографических преобразований (возможен множественный выбор):

а) пароли следует всегда хранить с использованием обратимого шифрования;
б) для асимметричного шифрования использовать ECC-криптографию с безопасными кривыми;
в) режим шифрования ECB не следует использовать вне особых обстоятельств;
г) возможна вставка криптографических ключей в исходный код приложения;
д) возможно совместное хранение ключа шифрования ключей и ключа шифрования данных.

19. Возможные последствия ошибки усечения целых чисел (возможен множественный выбор):

а) нежелательная потеря (искажение) данных;
б) индексное переполнение буфера;
в) неопределенное поведение программы

20. Если отрицательные значения передаются стандартным функциям копирования или выделения памяти в качестве размера буферов, то

а) они неявно преобразуются в большие беззнаковые значения, что может привести к переполнению буфера;

б) они неявно преобразуются в беззнаковые значения путем отбрасывания минуса (модуль числа), что может привести к неопределенному поведению программы;

в) фиксируются ошибки на этапе компиляции программы.

21. Сегмент BSS содержит:

а) глобальные и статические переменные, которые не были явным образом инициализированы в исходном коде, и которые будут при запуске программы инициализированы нулями;

б) глобальные переменные, которые инициализированы программистом;

в) локальные переменные и аргументы;

г) динамически размещаемые данные

22. Сегмент Heap содержит:

а) глобальные и статические переменные, которые не были явным образом инициализированы в исходном коде, и которые будут при запуске программы инициализированы нулями;

б) глобальные переменные, которые инициализированы программистом;

в) локальные переменные и аргументы;

г) динамически размещаемые данные

23. Механизм ASLR используется для:

а) быстрого отображения файлов в память;

б) увеличения пространства данных программы на заданное число байт;

в) рандомизации расположения сегментов в адресном пространстве процесса

24. Отметьте верные утверждения. При использовании механизма ASLR (возможен множественный выбор):

а) увеличится размер кода исполняемых файлов;

б) уменьшится время загрузки в память каждого исполняемого файла;

в) возникнет дополнительная несовместимость с ПО и библиотеками, разработанным под версии ОС без ASLR;

г) злоумышленнику будет проще реализовать атаку типа «возврат в библиотеку» (return-to-libc)

25. Отметьте верные утверждения (возможен множественный выбор).

а) адреса системных и библиотечных функций не меняются от одной операционной системы к другой (для обеспечения совместимости);

б) размер переполняющихся буферов может оказаться мал для вмещения в них загрузчика или затирания сколь ни будь значимых структур данных;

в) использование контейнеров и строк из стандартной библиотеки шаблонов предпочтительнее с точки зрения минимизации вероятности атак переполнения буфера.

Задания открытого типа

1. Дефекты ПО, не выявленные в ходе тестирования и не декларированные спецификацией разработчика, предоставляющие злоумышленникам исключительные возможности по разглашению информации законных пользователей, ее модификации, блокированию использованию и уничтожению.

2. Функциональная возможность ПО, не описанная или не соответствующая описанию в документации, при использовании которой возможно нарушение конфиденциальности, доступности или целостности обрабатываемой информации.

3. Способ анализа программы непосредственно при ее выполнении в определенной среде.

4. Известное значение, которое может помещаться между буфером и управляющими данными (адресом возврата) в стеке для мониторинга переполнения буфера.

Задания с развёрнутым ответом

1. Сформулируйте стратегии уменьшения рисков возникновения уязвимостей при выборе платформы и компилятора, разработке дизайна и архитектуры ПО, организации процесса разработки и тестирования.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутый ответ, корректно формулирует стратегии уменьшения рисков возникновения уязвимостей.	3 балла
Обучающийся приводит достаточно развернутый ответ. В формулировках стратегий уменьшения рисков возникновения уязвимостей могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В формулировках стратегий уменьшения рисков возникновения уязвимостей могут содержаться отдельные неточности.	1 балл
Представлено неполное или содержащее грубые ошибки описание стратегий уменьшения рисков возникновения уязвимостей.	0 баллов

2. Опишите типовые уязвимости для приложений, использующих криптографические методы. Приведите рекомендации по предотвращению данных уязвимости.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое описание типовых уязвимостей для приложений, использующих криптографические методы. Дает корректные рекомендации по предотвращению данных уязвимостей.	3 балла
Обучающийся приводит достаточно развернутое описание типовых уязвимостей для приложений, использующих криптографические методы. Дает корректные рекомендации по предотвращению данных уязвимостей. В описании могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описании типовых уязвимостей для приложений, использующих криптографические методы, могут содержаться отдельные неточности. Отсутствуют или приведены с существенными ошибками рекомендации по предотвращению данных уязвимостей.	1 балл
Представлено неполное или содержащее грубые ошибки описание типовых уязвимостей для приложений, использующих криптографические методы. Отсутствуют рекомендации по предотвращению данных уязвимостей.	0 баллов

3. Перечислите и охарактеризуйте основные технические меры защиты ПО.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое и безошибочное описание основных технических мер по защите ПО, включая выполнение на стороне сервера, водяные знаки и отпечатки пальцев, установку подлинности, шифрование, обфускацию кода и т.д.	3 балла
Обучающийся приводит достаточно развернутое описание основных технических мер по защите ПО. Описание может содержать незначительные неточности.	2 балла
Представлено краткое описание основных технических мер по защите ПО, отражающее лишь некоторые из известных подходов. Описание может содержать незначительные неточности.	1 балл
Представлено неполное или содержащее грубые ошибки описание основных технических мер по защите ПО.	0 баллов

Компетенция ПК-3.3

Задания закрытого типа

1. Отметьте фрагменты кода, содержащие потенциальные уязвимости (возможен множественный выбор)

а)
```
void encryptData(char *str)
{
    char pwd[64];
    if(getPassword(pwd, sizeof(pwd)))
    ...
    memset(pwd, 0, sizeof(pwd));
}
```

б)
```
int buf[5]={1,2,3,4,5};
int *ptr = buf;
delete[]ptr;
```

в)
```
char src[17] = "some test string", dst[18];
strncpy(dst, src, sizeof(src)-1);
```

г)
```
#define BUF_SIZE 256
void f(char* str)
{
    short len;
    char buf[BUF_SIZE];
    len = strlen(str);
    if(len < BUF_SIZE) strcpy(buf, str);
}
```

2. Отметьте фрагменты кода, содержащие потенциальные ошибки и уязвимости (возможен множественный выбор)

а)
```
int f(pthread_mutex_t *mutex)
{
    int result = pthread_mutex_lock(mutex);
    if(0 != result) return result;
    // access shared resource …
    return pthread_mutex_unlock(mutex);
}
```

б)
```
unsigned int f()
{
    int val=0;
    ...
    if(g() == ERROR) val = -1;
    ...
    return val;
}
```

в)
```
int flag;
char buf1[20], buf2[20];
...
printf("Enter string\n");
gets(buf1);
printf(buf1);
```

г)
```
int f(bool flag)
{
     int *ptr;
     if(flag) *ptr = 0;
...
}
```

3. Отметьте фрагменты кода, содержащие потенциальные ошибки и уязвимости (возможен множественный выбор)

а)
```
int *ptr = (int*)malloc(sizeof(int));
...
delete ptr;
```

б)
```
int *init_buf(int size)
{
    int *buf = new int[size];
    for(int i=0; i<size; i++) buf[i] = i++;
    return buf;
}
void delete_buf(int *&buf)
{
    delete []buf;
    buf = nullptr;
}
int main()
{
    int size = 8;
    int *buf = init_buf(size);
    ...
    delete_buf(buf);
    return 0;
}
```

в)
```
bool isValidAdd(unsigned short x, unsigned short y)
{
    if(static_cast<unsigned short>(x+y)< x) return false;
    return true;
}
```

г)
```
long size = init_size();
char *buf1, *buf2;
buf1 = (char*)malloc(size);
buf2 = (char*)malloc(size);
strncpy(buf2, buf1, size);
```

4. Отметьте фрагменты кода, содержащие потенциальные ошибки и уязвимости (возможен множественный выбор)

а)
```
#define SQUARE(val) val * val
```

б)
```
void f(pthread_mutex_t *mutex)
{
    pthread_mutex_lock(mutex);
    // access shared resource ...
    pthread_mutex_unlock(mutex);
}
```

в)
```
void f()
{
    MyObj *obj = new MyObj()
    ...
    delete obj;
}
```

г)
```
double divide(double x, double y) { return x/y; }
```

5. Отметьте фрагменты кода, содержащие потенциальные ошибки и уязвимости (возможен множественный выбор)

а)
```
int *ptr = new int[100];
...
delete[]ptr;
```

б)
```
int f()
{
     int *p = malloc(SIZE);
     int *q = malloc(SIZE);
     …
     if(flag) p = q;
     …
}
```

в)
```
int f(int x, int y)
{
int d;
     try { d = x/y;   }
     catch(...) { /*...*/ }
     ...
}
```

г)
```
void code_ptr()
{
    char buff[8];
    void (*some_func) ();
    ...
    printf("password:");
    gets(buff);
    ...
    some_func();
}
```

Задания открытого типа

1. Приведите примеры инструментов (минимум 3) для статического анализа кода на языке C/C++.

2. Совокупность методик и средств по трансформации кода, меняющих его представление, но сохраняющих логику, направленных на затруднение анализа исходного и исполняемого кода программ.

3. Утилиты, транслирующие исполняемый модуль в относительно эквивалентный исходный код на языке программирования высокого уровня.

4. Программа, фрагмент кода или последовательность команд, использующие уязвимости в ПО и применяемые для проведения атаки на вычислительную систему.

5. Проверка программного обеспечения на соответствие требованиям безопасности для выявления уязвимостей, которые могут появиться на этапе его эксплуатации.

Задания с развёрнутым ответом

1. Приведите правила преобразования целых типов на языках C/C++. Опишите уязвимости при работе с целыми числами, приведите примеры фрагментов кода, содержащих уязвимости переполнения целых чисел.

Критерии оценивания	Шкала оценок
Обучающийся приводит полное и безошибочное описание правил преобразования целых типов на языках C/C++. Приводит корректное развернутое описание и примеры уязвимостей целочисленного переполнения.	3 балла
Обучающийся приводит корректное описание правил преобразования целых типов на языках C/C++. В приведенном описании уязвимостей целочисленного переполнения могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описание правил преобразования целых типов на языках C/C++, а также описании уязвимостей целочисленного переполнения могут содержаться отдельные неточности. Отсутствуют примеры фрагментов кода, содержащих уязвимости переполнения целых чисел.	1 балл
Представлено неполное или содержащее грубые ошибки описание правил преобразования целых типов на языках C/C++ и описание уязвимостей целочисленного переполнения. Отсутствуют примеры фрагментов кода.	0 баллов

2. Опишите уязвимость типа переполнения буфера в стеке. Приведите рекомендации по обнаружению или предотвращению данной уязвимости.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое описание уязвимости переполнения буфера в стеке. Дает корректные рекомендации по обнаружению или предотвращению данной уязвимости.	3 балла
Обучающийся приводит достаточно развернутое описание уязвимости переполнения буфера в стеке. Дает корректные рекомендации по обнаружению или предотвращению данной уязвимости. В описании могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описании уязвимости переполнения буфера в стеке могут содержаться отдельные неточности. Отсутствуют или приведены с существенными ошибками рекомендации по обнаружению или предотвращению данной уязвимости.	1 балл
Представлено неполное или содержащее грубые ошибки описание уязвимости переполнения буфера в стеке. Отсутствуют рекомендации по обнаружению или предотвращению данной уязвимости.	0 баллов

3. Опишите уязвимость типа переполнения буфера в куче. Приведите рекомендации по обнаружению или предотвращению данной уязвимости.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое описание уязвимости переполнения буфера в куче. Дает корректные рекомендации по обнаружению или предотвращению данной уязвимости.	3 балла
Обучающийся приводит достаточно развернутое описание уязвимости переполнения буфера в куче. Дает корректные рекомендации по обнаружению или предотвращению данной уязвимости. В описании могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описании уязвимости переполнения буфера в куче могут содержаться отдельные неточности. Отсутствуют или приведены с существенными ошибками рекомендации по обнаружению или предотвращению данной уязвимости.	1 балл
Представлено неполное или содержащее грубые ошибки описание уязвимости переполнения буфера в куче. Отсутствуют рекомендации по обнаружению или предотвращению данной уязвимости.	0 баллов

4. Опишите уязвимости форматных строк. Приведите рекомендации по обнаружению или предотвращению данного типа уязвимостей.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое описание уязвимостей форматных строк. Дает корректные рекомендации по обнаружению или предотвращению таких уязвимостей.	3 балла
Обучающийся приводит достаточно развернутое описание уязвимостей форматных строк. Дает корректные рекомендации по обнаружению или предотвращению данных уязвимостей. В описании могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описании уязвимостей форматных строк могут содержаться отдельные неточности. Отсутствуют или приведены с существенными ошибками рекомендации по обнаружению или предотвращению таких уязвимостей.	1 балл
Представлено неполное или содержащее грубые ошибки описание уязвимостей форматных строк. Отсутствуют рекомендации по обнаружению или предотвращению таких уязвимостей.	0 баллов

5. Опишите уязвимости некорректного использования завершающего нуля. Приведите рекомендации по обнаружению или предотвращению данного типа уязвимостей.

Критерии оценивания	Шкала оценок
Обучающийся приводит развернутое описание уязвимостей некорректного использования завершающего нуля. Дает корректные рекомендации по обнаружению или предотвращению таких уязвимостей.	3 балла
Обучающийся приводит достаточно развернутое описание уязвимостей некорректного использования завершающего нуля. Дает корректные рекомендации по обнаружению или предотвращению данных уязвимостей. В описании могут содержаться незначительные неточности.	2 балла
Обучающийся приводит недостаточно развернутый ответ. В описании уязвимостей некорректного использования завершающего нуля могут содержаться отдельные неточности. Отсутствуют или приведены с существенными ошибками рекомендации по обнаружению или предотвращению таких уязвимостей.	1 балл
Представлено неполное или содержащее грубые ошибки описание уязвимостей некорректного использования завершающего нуля. Отсутствуют рекомендации по обнаружению или предотвращению таких уязвимостей.	0 баллов

20.2 Промежуточная аттестация

Промежуточная аттестация может включать в себя проверку теоретических вопросов, а также, при необходимости (в случае невыполнения в течение семестра), проверку выполнения установленного перечня лабораторных заданий, позволяющих оценить уровень полученных знаний и/или практическое (ие) задание(я), позволяющее (ие) оценить степень сформированности умений и навыков.

Для оценки теоретических знаний используется перечень контрольно-измерительных материалов. Каждый контрольно-измерительный материал для проведения промежуточной аттестации включает два задания - вопросов для контроля знаний, умений и владений в рамках оценки уровня сформированности компетенции. При оценивании используется количественная шкала. Критерии оценивания приведены в таблице ниже.

Для оценивания результатов обучения на экзамене используются следующие содержательные показатели (формулируется с учетом конкретных требований дисциплины):

1. знание теоретических основ учебного материала, основных определений, понятий и используемой терминологии;
2. умение обосновывать свои суждения и профессиональную позицию по излагаемому вопросу;
3. владение навыками программирования, использования современных программных средств разработки и отладки программ.
4. владение навыками анализа исходного кода на предмет наличия уязвимостей, навыками использования специализированных утилит статического и динамического анализа кода.

Различные комбинации перечисленных показателей определяют критерии оценивания результатов обучения (сформированности компетенций) на зачете:

- высокий (углубленный) уровень сформированности компетенций;
- повышенный (продвинутый) уровень сформированности компетенций;
- пороговый (базовый) уровень сформированности компетенций.

Для оценивания результатов обучения на зачете используется – зачтено, не зачтено по результатам тестирования.

Соотношение показателей, критериев и шкалы оценивания результатов обучения на государственном экзамене представлено в следующей таблице.

Критерии оценивания компетенций и шкала оценок

Критерии оценивания компетенций	Уровень сформированности компетенций	Шкала оценок
Обучающийся демонстрирует полное соответствие знаний, умений, навыков по приведенным критериям свободно оперирует понятийным аппаратом и приобретенными знаниями, умениями, применяет их при решении практических задач. Успешно выполнены лабораторные работы в соответствии с установленным перечнем.	Повышенный уровень	Отлично
Ответ на контрольно-измерительный материал не полностью соответствует одному из перечисленных выше показателей, но обучающийся дает правильные ответы на дополнительные вопросы. При этом обучающийся демонстрирует соответствие знаний, умений, навыков приведенным в таблицах показателям, но допускает незначительные ошибки, неточности, испытывает затруднения при решении практических задач. Успешно выполнены лабораторные работы в соответствии с установленным перечнем.	Базовый уровень	Хорошо
Обучающийся демонстрирует неполное соответствие знаний, умений, навыков приведенным в таблицах показателям, допускает значительные ошибки при решении практических задач. При этом ответ на контрольно-измерительный материал не соответствует любым двум из перечисленных показателей, обучающийся дает неполные ответы на дополнительные вопросы. Успешно выполнены лабораторные работы в соответствии с установленным перечнем.	Пороговый уровень	Удовлетворительно
Ответ на контрольно-измерительный материал не соответствует любым трем из перечисленных показателей. Обучающийся демонстрирует отрывочные, фрагментарные знания, допускает грубые ошибки. Не выполнены лабораторные работы в соответствии с установленным перечнем.	–	Неудовлетворительно

Примерный перечень вопросов к зачету

№	Содержание
1	Классификация уязвимостей ПО
2	Уязвимость переполнения буфера в стеке
3	Уязвимость переполнения буфера в куче
4	*Уязвимость переполнения целых*
5	*Уязвимость форматной строкой*
6	*Методы обнаружения уязвимостей. Тестирование по принципу «белого ящика»*
7	*Методы обнаружения уязвимостей. Тестирование по принципу «черного ящика»*
8	*Динамический анализ кода, фаззингтестирование*
9	*Проблемы безопасности ПО, связанные с компиляторной оптимизацией*
10	*Принципы создания безопасного ПО, ГОСТ Р569392016*
11	*Методы защиты ПО от взлома*
12	*Технические меры защиты ПО*
13	*Приемы обфускации*
14	*Динамическое ветвление и контекстная зависимость*
15	*Динамические анализаторы кода*
16	*Средства отладки и взлома программ*
17	*Обфускация абстрактных данных*
18	Обфускация кода на этапе дизассемблирования

Пример контрольно-измерительного материала

УТВЕРЖДАЮ

Заведующий кафедрой технологий обработки и защиты информации

_____ А.А. Сирота

__.__.2024

Направление подготовки / специальность 10.05.01 Компьютерная безопасность

Дисциплина Б1.В.05 Анализ уязвимостей программного обеспечения

Форма обучения Очное

Вид контроля Зачет с оценкой

Вид аттестации Промежуточная

Контрольно-измерительный материал № 1

1. Классификация уязвимостей ПО
2. Статические анализаторы кода

Преподаватель_____ М.А. Дрюченко