ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ

C=RU, S=Ставропольский край, L=г. Ставрополь, T=проректор по учебной работе, OU=ректорат, O="ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ "" СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ""", СНИЛС=15230589951, ИНН=263411176761, E=s.a.maksimovich@stgau.ru, G=Светлана Александровна, SN=Максимович, CN=Максимович Светлана Александровна

УТВЕРЖДАЮ

Директор/Декан
факультета цифровых технологий
Шлаев Дмитрий Валерьевич

_____
_____
«___» _____ 20___ г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫХ МАТЕРИАЛОВ)

Б1.О.14 Технологии разработки защищенного ПО

09.04.03 Прикладная информатика

Искусственный интеллект в кибербезопасности

магистр

очная

Ставрополь, 2026

1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Процесс изучения дисциплины направлен на формирование следующих компетенций ОП ВО и овладение следующими результатами обучения по дисциплине:

Код и наименование компетенции	Код и наименование индикатора достижения	Перечень планируемых результатов обучения по дисциплине
ОПК-2 Способен разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием современных интеллектуальных технологий, для решения профессиональных задач	ОПК-2.1 Понимает методологические основы современных информационно-коммуникационных и интеллектуальных технологий для решения профессиональных задач	**знает** методологические основы современных информационно-коммуникационных и интеллектуальных технологий для решения профессиональных задач
		умеет применять методологические основы современных информационно-коммуникационных и интеллектуальных технологий для решения профессиональных задач
		владеет навыками навыками работы с современными информационно-коммуникационными и интеллектуальными технологиями для решения профессиональных задач
ОПК-2 Способен разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием современных интеллектуальных технологий, для решения профессиональных задач	ОПК-2.3 Разрабатывает оригинальные алгоритмы и программные средства, в том числе с использованием интеллектуальных технологий, для решения профессиональных задач	**знает** алгоритмы и программные средства, в том числе с использованием интеллектуальных технологий, для решения профессиональных задач
		умеет разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием интеллектуальных технологий, для решения профессиональных задач
		владеет навыками навыками разработки оригинальных алгоритмов и программных средств с использованием интеллектуальных технологий для решения профессиональных задач
ОПК-5 Способен разрабатывать и модернизировать программное и аппаратное обеспечение информационных и автоматизированных систем	ОПК-5.2 Модернизирует программное и аппаратное обеспечения информационных и автоматизированных систем для решения профессиональных задач	**знает** программное и аппаратное обеспечения информационных и автоматизированных систем для решения профессиональных задач
		умеет модернизировать программное и аппаратное обеспечения информационных и автоматизированных систем для решения профессиональных задач
		владеет навыками навыками модернизации программного и аппаратного обеспечения информационных и автоматизированных систем для решения профессиональных задач

		знает программное и аппаратное обеспечение информационных и автоматизированных систем для решения профессиональных задач
		умеет выбирать программное и аппаратное обеспечение информационных и автоматизированных систем для решения профессиональных задач
		владеет навыками навыками разработки программного и аппаратного обеспечения информационных и автоматизированных систем для решения профессиональных задач
ПК-2 Способен управлять жизненным циклом программных продуктов и проектами в области информационных технологий	ПК-2.1 Управляет процессом разработки программного обеспечения, осуществляя планирование, контроль исполнения и корректировку планов	**знает** как осуществлять планирование, контроль исполнения и корректировку планов при разработке ПО
		умеет управлять процессом разработки программного обеспечения, осуществляя планирование, контроль исполнения и корректировку планов
		владеет навыками навыками управления процессом разработки программного обеспечения, осуществляя планирование, контроль исполнения и корректировку планов
ПК-4 Способен администрировать средства защиты информации и обеспечивать безопасность информационных систем	ПК-4.1 Администрирует средства защиты информации прикладного и системного программного обеспечения (ОС, СУБД, приложения), настраивая политики безопасности и контролируя их соблюдение	**знает** средства защиты информации прикладного и системного программного обеспечения (ОС, СУБД, приложения), настраивая политики безопасности и контролируя их соблюдение
		умеет администрировать средства защиты информации прикладного и системного программного обеспечения (ОС, СУБД, приложения), настраивая политики безопасности и контролируя их соблюдение
		владеет навыками навыками администрирования средств защиты информации прикладного и системного ПО, настраивая политики безопасности и контролируя их соблюдение

2. Перечень оценочных средств по дисциплине

№	Наименование раздела/темы	Семестр	Код индикаторов достижения компетенций	Оценочное средство проверки результатов достижения индикаторов компетенций

1.	1 раздел. Технологии разработки защищенного ПО			
1.1.	Введение в безопасную разработку. Модели угроз и экосистема Rust	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Устный опрос
1.2.	Управление памятью как основа защищенного ПО	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Тест
1.3.	Аудит кода и статический анализ безопасности (SAST)	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Устный опрос
1.4.	Криптографические примитивы и безопасное хранение данных	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Устный опрос
1.5.	Сетевые взаимодействия и защита от инъекций	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Тест
1.6.	Финальная сборка, аудит и безопасный релиз (DevSecOps)	3	ОПК-2.1, ОПК-2.3, ОПК-5.2, ОПК-5.3, ПК-2.1, ПК-4.1	Устный опрос
	Промежуточная аттестация			За

3. Оценочные средства (оценочные материалы)

Примерный перечень оценочных средств для текущего контроля успеваемости и промежуточной аттестации

№ п/п	Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде (Оценочные материалы)
Текущий контроль			
	Для оценки знаний		
1	Устный опрос	Средство контроля знаний студентов, способствующее установлению непосредственного контакта между преподавателем и студентом, в процессе которого преподаватель получает широкие возможности для изучения индивидуальных особенностей усвоения студентами учебного материала.	Перечень вопросов для устного опроса

2	Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий
	Для оценки умений		
	Для оценки навыков		
	Промежуточная аттестация		
3	Зачет	Средство контроля усвоения учебного материала практических и семинарских занятий, успешного прохождения практик и выполнения в процессе этих практик всех учебных поручений в соответствии с утвержденной программой с выставлением оценки в виде «зачтено», «незачтено».	Перечень вопросов к зачету

4. Примерный фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю) "Технологии разработки защищенного ПО"

Примерные оценочные материалы для текущего контроля успеваемости

Какой механизм языка Rust гарантирует безопасность работы с памятью на этапе компиляции?

A) Сборщик мусора (Garbage Collector)

B) Система владения (Ownership) с правилами заимствования и временами жизни

C) Виртуальная машина с байт-кодом

D) Динамическая проверка типов во время выполнения

Что произойдет при попытке скомпилировать код, который создает изменяемую ссылку (&mut) на данные, уже имеющие неизменяемую ссылку (&) в той же области видимости?

A) Компиляция пройдет успешно, но будет предупреждение

B) Компиляция не удастся с ошибкой проверки заимствований (borrow checker)

C) Программа упадет во время выполнения с паникой (panic)

D) Данные будут автоматически скопированы в новую область памяти

Какая уязвимость НЕВОЗМОЖНА в безопасной части Rust (без использования unsafe)?

A) SQL-инъекция

B) Переполнение буфера (Buffer Overflow)

C) Гонка данных (Data Race) в многопоточном коде

D) Отказ в обслуживании (DoS) через бесконечный цикл

Какой трейт (trait) гарантирует безопасность передачи владения данными между потоками в Rust?

A) Clone

B) Drop

C) Send

D) Default

Какой инструмент статического анализа рекомендуется использовать для проверки "стиля" и обнаружения распространенных ошибок в коде Rust?

A) Valgrind

B) Clippy

C) GDB

D) AddressSanitizer

Что означает ключевое слово unsafe в Rust?

A) Код будет выполняется медленнее

B) Код доверяется программисту и может выполнять операции, не проверяемые компилятором на безопасность

C) Код будет автоматически удален из финальной сборки

D) Код работает только в отладочном режиме (debug)

Как правильно обработать ситуацию, когда результат операции может отсутствовать (например, поиск в коллекции)?

A) Использовать panic!() для немедленного завершения

B) Использовать тип Option<T> и сопоставление с образцом (match)

C) Использовать null и проверять на null

D) Игнорировать результат и продолжить выполнение

Какая библиотека (крейт) рекомендуется для безопасного затирания секретов (паролей, ключей) из памяти?

A) serde

B) tokio

C) zeroize

D) log

Что такое SBOM (Software Bill of Materials) в контексте безопасной разработки?

A) Инструмент для профилирования производительности

B) Перечень всех компонентов и зависимостей, используемых в проекте

C) Методология тестирования на проникновение

D) Система управления версиями кода

Какой механизм в Rust позволяет создавать абстракции с нулевой стоимостью (zero-cost abstractions)?

A) Виртуальные таблицы (vtable) как в C++

B) Монады как в Haskell

C) Дженерики (Generics) и мономорфизация на этапе компиляции

D) Динамическая диспетчеризация через dyn Trait

Примерные оценочные материалы
для проведения промежуточной аттестации (зачет, экзамен)
по итогам освоения дисциплины (модуля)

Какой механизм языка Rust гарантирует безопасность работы с памятью на этапе компиляции?

A) Сборщик мусора (Garbage Collector)

B) Система владения (Ownership) с правилами заимствования и временами жизни

C) Виртуальная машина с байт-кодом

D) Динамическая проверка типов во время выполнения

Что произойдет при попытке скомпилировать код, который создает изменяемую ссылку (&mut) на данные, уже имеющие неизменяемую ссылку (&) в той же области видимости?

A) Компиляция пройдет успешно, но будет предупреждение

B) Компиляция не удастся с ошибкой проверки заимствований (borrow checker)

C) Программа упадет во время выполнения с паникой (panic)

D) Данные будут автоматически скопированы в новую область памяти

Какая уязвимость НЕВОЗМОЖНА в безопасной части Rust (без использования unsafe)?

A) SQL-инъекция

B) Переполнение буфера (Buffer Overflow)

C) Гонка данных (Data Race) в многопоточном коде

D) Отказ в обслуживании (DoS) через бесконечный цикл

Какой трейт (trait) гарантирует безопасность передачи владения данными между потоками в Rust?

A) Clone

B) Drop

C) Send

D) Default

Какой инструмент статического анализа рекомендуется использовать для проверки "стиля" и обнаружения распространенных ошибок в коде Rust?

A) Valgrind

B) Clippy

C) GDB

D) AddressSanitizer

Что означает ключевое слово unsafe в Rust?

A) Код будет выполняется медленнее

B) Код доверяется программисту и может выполнять операции, не проверяемые компилятором на безопасность

C) Код будет автоматически удален из финальной сборки

D) Код работает только в отладочном режиме (debug)

Как правильно обработать ситуацию, когда результат операции может отсутствовать (например, поиск в коллекции)?

A) Использовать panic!() для немедленного завершения

B) Использовать тип Option<T> и сопоставление с образцом (match)

C) Использовать null и проверять на null

D) Игнорировать результат и продолжить выполнение

Какая библиотека (крейт) рекомендуется для безопасного затирания секретов (паролей, ключей) из памяти?

A) serde

B) tokio

C) zeroize

D) log

Что такое SBOM (Software Bill of Materials) в контексте безопасной разработки?

A) Инструмент для профилирования производительности

B) Перечень всех компонентов и зависимостей, используемых в проекте

C) Методология тестирования на проникновение

D) Система управления версиями кода

Какой механизм в Rust позволяет создавать абстракции с нулевой стоимостью (zero-cost abstractions)?

A) Виртуальные таблицы (vtable) как в C++

B) Монады как в Haskell

C) Дженерики (Generics) и мономорфизация на этапе компиляции

D) Динамическая диспетчеризация через dyn Trait

Темы письменных работ (эссе, рефераты, курсовые работы и др.)

Сравнительный анализ моделей управления памятью в C++, Go и Rust с точки зрения безопасности.

Акцент: уязвимости памяти (UAF, Double Free, Buffer Overflow) и способы их предотвращения.

Моделирование угроз по методологии STRIDE для приложений, разработанных на Rust.

Акцент: применение к микросервисной и клиент-серверной архитектуре.

Эволюция SSDLC (Secure Software Development Lifecycle): от Waterfall к DevSecOps.

Акцент: место Rust в современном безопасном конвейере разработки.

Система владения (Ownership) в Rust как архитектурный паттерн безопасности.

Акцент: сравнение с умными указателями в C++ (unique_ptr, shared_ptr).

Правила заимствования и времена жизни (Lifetimes): как компилятор Rust предотвращает висячие ссылки.

Акцент: примеры кода, демонстрирующие ошибки компиляции.

Обработка ошибок без исключений: типы Option<T> и Result<T, E> как способ защиты от непредвиденных состояний.

Акцент: сравнение с механизмами исключений в Java/Python.

Clippy и Rust Analyzer как инструменты Security Code Review.

Акцент: категории линтов, обнаружение потенциальных уязвимостей, настройка в CI/CD.

MIRI — интерпретатор неопределенного поведения: поиск UB в коде на Rust.

Акцент: примеры UB в unsafe-коде и их обнаружение.

Фаззинг-тестирование с cargo-fuzz: методология и примеры обнаружения уязвимостей.

Акцент: интеграция фаззинга в процесс разработки, кейсы из реальных проектов.

Правила безопасного использования unsafe в Rust: инкапсуляция и аудит.

Акцент: когда unsafe необходим и как минимизировать его поверхность.

Разработка безопасной обертки над C-библиотекой на примере OpenSSL или SQLite.

Акцент: работа с FFI, защита от ошибок работы с сырыми указателями.

Сравнение криптографических библиотек в экосистеме Rust: ring, orion, rust-crypto, dalek.

Акцент: безопасность API, производительность, поддержка алгоритмов.

Безопасное хеширование паролей на Rust: реализация Argon2 с использованием крейта argon2.

Акцент: настройка параметров, защита от атак перебора.

Защита секретов в памяти с использованием крейта zeroize.

Акцент: предотвращение утечек через дампы памяти, отладчики и анализ ядра.

Безопасный сетевой стек на Rust: реализация TLS-сервера с использованием tokio и rustls.

Акцент: сравнение с OpenSSL, конфигурация, защита от атак на TLS.

Защита REST API на Rust от инъекционных атак (SQL, Command Injection).

Акцент: использование строгой типизации и валидации через типажи.

Проектирование устойчивого к DDoS микросервиса на Rust (таймауты, лимиты, backpressure).

Акцент: настройка tokio, использование tower middleware.

Контейнеризация Rust-приложений: создание минимальных Docker-образов и distroless-контейнеров.

Акцент: минимизация поверхности атаки, сканирование уязвимостей образов.

Составление SBOM (Software Bill of Materials) для Rust-проектов: инструменты и форматы.

Акцент: cargo-audit, cargo-deny, интеграция с CI/CD.

Применение Rust в критической инфраструктуре: опыт использования в Android, AWS, Microsoft.

Акцент: конкретные проекты (Android Rust, Firecracker, Azure IoT Edge), статистика безопасности, перспективы.