МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра Информационных технологий и систем безопасности

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

(Дипломная работа)

На тему « Разработка модели системы ИИ для расширения возможностей защиты веб-приложений в системе WAF-систем **»**

Исполнитель _____ Лупандин Владимир Юрьевич

(подпись) (фамилия, имя, отчество)

Руководитель _____ Козлов Юрий Викторович

(подпись) (фамилия, имя, отчество)

«К защите допускаю»

Заведующий кафедрой _____ Лепешкин Олег Михайлович

(подпись) (фамилия, имя, отчество)

«____»____ 20___ г.

Санкт-Петербург

МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра Информационных технологий и систем безопасности

«УТВЕРЖДАЮ»

Заведующий кафедрой

_____ _____

(подпись) (фамилия, имя, отчество)

«___»___20___ года

Задание

на выпускную квалификационную работу

студенту _____

(фамилия, имя, отчество)

1. Тема Разработка модели системы ИИ для расширения возможностей защиты веб-приложений в системе WAF-систем

закреплена приказом ректора Университета от «_____» _____20___ года, № ____

2. Срок сдачи законченной работы «___» _____20___ года

3. Исходные данные к выпускной квалификационной работе:

4. Перечень вопросов, подлежащих разработке (краткое содержание работы:

Введение. Актуальность темы, цели и задачи ВКР

Глава 1 Основы обеспечения безопасности в образовательной организации

(наименование главы)

Глава 2 Разработка аналитико-математической модели принятия решений в системе обеспечения информационной безопасности образовательной организации

(наименование главы)

Глава 3 Методика применения аналитико-математической модели в системе обеспечения информационной безопасности образовательной организации

(наименование главы)

Глава 4 Научно-экономическое обоснование методики построения системы управления информационной безопасностью образовательной организации

(наименование главы)

Заключение. Выводы по работе в целом. Оценка степени решения поставленных задач. Практические рекомендации.

5. Перечень материалов, представляемых к защите:

– Пояснительная записка;

– Схема _____

(наименование схемы)

– Диаграмма _____

(наименование диаграммы)

6. Консультанты по работе

6.1. _____

6.2. _____

…

7. Дата выдачи задания: «__» _____ 20_____ года Руководитель выпускной квалификационной работы

_____ _____

(должность, ученая степень, ученое звание, фамилия, имя, отчество) (подпись)

Задание принял к исполнению «___»_____20___ года

Студент _____ _____

(фамилия, имя, отчество, учебная группа) (подпись)

…

8. Дата выдачи задания: «__» _____ 20_____ года Руководитель выпускной квалификационной работы

_____ _____

(должность, ученая степень, ученое звание, фамилия, имя, отчество) (подпись)

Задание принял к исполнению «___»_____20___ года

Студент _____ _____

(фамилия, имя, отчество, учебная группа) (подпись)

РЕФЕРАТ

Дипломная работа: __с., ____рис., ____табл., _____ приложения, ____ источников литературы.

СИСТЕМА УПРАВЛЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТЬЮ, СТАНДАРТИЗАЦИЯ В ОБЛАСТИ СИСТЕМ УПРАВЛЕНИЯ ИНФОРМАЦИОННОЙБЕЗОПАСНОСТЬЮ, ПОЛИТИКА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ, АНАЛИЗ РИСКОВ.

Объект исследования:

Предмет исследования:

Цель работы:

В дипломной работе проводится анализ…

Разработан …

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 5
ГЛАВА 1. АНАЛИТИЧЕСКИЙ ОБЗОР И ТЕОРЕТИЧЕСКИЕ ОСНОВЫ 9
1.1. Угрозы безопасности веб-приложений и особенности прикладного уровня 9
1.2. Класс WAF-систем и принципы работы ModSecurity/CRS 11
1.3. Проблема тюнинга правил и критерии качества фильтрации 13
1.4. Использование ИИ и LLM для расширения возможностей WAF: потенциал и риски 15
1.5. Нормативные и организационные ограничения для внедрения автоматизации в ИБ 17
ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ МОДЕЛИ СИСТЕМЫ ИИ В СОСТАВЕ WAF-СИСТЕМ 19
2.1. Требования к системе и исходные предпосылки проектирования 19
2.2. Архитектура стенда и логическая декомпозиция компонентов 20
2.3. Автоматизация установки ОС и развертывания Kubernetes 21
2.4. Реализация WAF-контура: конфигурации, правила и механизмы обновления 23
2.5. Контур мониторинга: сбор, хранение и визуализация событий 24
2.6. Контур LLM-агентов: назначение, взаимодействие и ограничения 25
2.7. Модель данных и хранение телеметрии 28
2.8. Жизненный цикл правила и протокол контролируемого внедрения 29
2.9. Меры безопасности и ограничения применения LLM 30
ГЛАВА 3. АЛГОРИТМ ВНЕДРЕНИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ МЕТОДИКИ 32
3.1. Алгоритм внедрения и использования разработанного решения 32
3.2. Результаты моделирования и экспериментальной оценки 33
3.3. Практический пример применения методики 37
3.4. Критерии эффективности применения методики 38
ГЛАВА 4. ОБОСНОВАНИЕ, НОВИЗНА, ПРАКТИЧЕСКАЯ ЗНАЧИМОСТЬ И ЭКОНОМИЧЕСКАЯ ОЦЕНКА 41
4.1. Обоснование проектных решений и предметной основы 41
4.2. Научная новизна и практическая значимость результатов 42
4.3. Экономическая оценка эффективности и ограничения применимости 43
ЗАКЛЮЧЕНИЕ 46
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 47

ВВЕДЕНИЕ

Рост доли бизнес-процессов, реализуемых через веб-интерфейсы и программные интерфейсы (API), приводит к росту критичности уязвимостей веб-приложений и к расширению спектра прикладных атак. Практическим ориентиром для систематизации рисков служит перечень OWASP Top 10 (2021), в котором фиксируются наиболее распространенные классы проблем (ошибки контроля доступа, инъекции, небезопасные конфигурации и др.). [17] В российской практике требования обеспечения защищенности и обработки данных обосновываются нормами законодательства об информации, персональных данных, а также регуляторными требованиями к защите объектов и информационных систем. [3] [4] [5] [6] [7] [8] [9]

В прикладных системах защиты веб-ресурсов ключевую роль играет класс средств WAF (Web Application Firewall), который обеспечивает контроль и блокирование нежелательных HTTP-запросов по сигнатурным и/или эвристическим правилам. На практике широкое распространение получили решения на базе Nginx и ModSecurity с подключением набора правил OWASP Core Rule Set (CRS) и механизмов оценивания аномалий (anomaly scoring). [18] [19] [20] При этом эксплуатация WAF сопряжена с высокой трудоемкостью ручного тюнинга правил и обработкой ложноположительных срабатываний, а также с необходимостью сокращения времени реакции на новые угрозы на основе актуализации правил.[18]

Развитие больших языковых моделей (LLM) сформировало предпосылки для применения генеративного ИИ в задачах анализа телеметрии, корреляции событий и формирования предложений по политикам безопасности. Однако использование

LLM в контуре защиты требует формализации ограничений, механизмов валидации и контролируемого внедрения изменений, поскольку модели подвержены генерации некорректных решений и могут провоцировать деградацию качества фильтрации.[25] [26]

Настоящая выпускная квалификационная работа посвящена теме «Разработка модели системы ИИ для расширения возможностей защиты веб-приложений в системе WAF-систем». Решение реализовано в виде воспроизводимого лабораторного стенда, развертываемого автоматически: автоустановка Debian на трех узлах (одна управляющая нода и две рабочие), развертывание кластера Kubernetes с помощью Kubespray, а также установка сервисов в отдельных пространствах имен (namespace) - WAF, LLM и мониторинг.

В WAF-контуре используется Nginx + ModSecurity + OWASP CRS; в мониторинге - Loki/Promtail/Grafana для централизованного сбора журналов и визуализации; в LLM-контуре - ollama и модель Qwen2.5-3B-Instruct, а также набор агентов, выполняющих сбор событий, анализ обходов WAF и безопасное внесение обновлений правил через ConfigMap и контролируемое применение к подам. Технические артефакты стенда оформлены как набор скриптов и манифестов Kubernetes (приложения к работе).

Структура текста и оформление ссылок ориентированы на требования ГОСТ к научно-техническим отчетам и библиографическому описанию, применяемые при подготовке выпускных квалификационных работ.[1] [2]

Целью работы является снижение трудоемкости эксплуатации WAF и повышение доли отраженных прикладных атак за счет внедрения модели системы ИИ, автоматически формирующей предложения по корректировке правил

ModSecurity/CRS на основании телеметрии и результатов тестирования, при сохранении допустимого уровня ложноположительных срабатываний.

Для достижения цели решаются следующие задачи:

-выполнить анализ предметной области WAF-защиты и ограничений сигнатурного подхода в условиях динамичной угрозной среды;

разработать архитектуру воспроизводимого стенда (автоустановка ОС, развертывание Kubernetes, манифесты сервисов WAF/LLM/мониторинга) с учетом требований безопасности и управляемости;

-сформировать модель данных для накопления событий ModSecurity и результатов тестов; определить жизненный цикл правила и протокол безопасного внесения изменений;

-разработать алгоритм применения LLM-агентов (сбор-анализ-предложение-валидация-внедрение) и набор ограничений, обеспечивающих приемлемый риск автоматизации;

-провести моделирование и экспериментальную оценку по методике GoTestWAF; определить критерии эффективности: доля отраженных атак, уровень false positive, время реакции на новые угрозы и трудоемкость сопровождения.

Объект исследования - процессы защиты веб-приложений на уровне прикладного протокола HTTP/HTTPS в контуре WAF-систем, функционирующих в контейнеризованной инфраструктуре. Предмет исследования - методы автоматизированной адаптации правил ModSecurity/CRS на основании телеметрии, логов и результатов автоматизированного тестирования с применением LLM.

Методы исследования включают: анализ нормативных и технических источников, проектирование архитектуры и модели данных, экспериментальное моделирование на стенде, сравнительную оценку по метрикам качества (TPR/FPR) и эксплуатационным метрикам (время реакции, трудоемкость). Для проверки

эффективности используется инструмент GoTestWAF, обеспечивающий воспроизводимую методику оценки WAF и API-защиты.[27]

ГЛАВА 1. АНАЛИТИЧЕСКИЙ ОБЗОР И ТЕОРЕТИЧЕСКИЕ ОСНОВЫ

1.1. Угрозы безопасности веб-приложений и особенности прикладного уровня

Веб-приложение в современной архитектуре выступает фронт-эндом к бизнес-логике и данным, включая персональные и коммерчески значимые сведения. В отличие от сетевых атак, прикладные атаки используют корректно установленные TCP-соединения и легитимные протоколы (HTTP/HTTPS) и поэтому плохо детектируются средствами, ориентированными только на сетевые признаки. Наиболее распространенные классы атак на прикладном уровне описываются в OWASP Top 10 и включают инъекции (SQL/LDAP/OS-команды), XSS, обходы аутентификации/авторизации, нарушения целостности данных и др. [17] В контексте российских требований по защите информации значимость угроз определяется требованиями к защите персональных данных и критической инфраструктуры, а также подходами к моделированию угроз и рисков.[3] [4] [5] [6] [9] [33]

Ключевая особенность прикладного уровня состоит в том, что «вредоносность» запроса определяется не только наличием сигнатур, но и контекстом: структурой URL-путей, параметрами, телом запроса, заголовками, последовательностью запросов, наличием токенов и бизнес-правил. Поэтому для практической защиты применяются уровни контроля, дополняющие сетевую фильтрацию: контроль входных данных, WAF-проверки, защита API, а также процесс secure development (безопасная разработка) и управление конфигурациями.

Поверхность атаки веб-приложения на прикладном уровне описывается структурой HTTP-запроса и контекстами интерпретации его элементов сервером и прикладной логикой: строка запроса (URI, query string), заголовки, cookies, тело

(JSON/XML/form-data), а также параметры маршрутизации и аутентификации. В отличие от атак на сетевом уровне, прикладные воздействия часто не имеют “аномальности” на уровне пакетов и требуют анализа смысловых признаков, включая кодировки, вложенные структуры и сериализации данных. Типовые классы рисков систематизированы, в частности, в OWASP Top 10, где доминируют проблемы контроля доступа, инъекционные классы уязвимостей и ошибки обработки данных.[17]

В современных архитектурах значимую долю трафика составляют API-запросы (REST/JSON, SOAP, GraphQL), для которых традиционные сигнатурные подходы усложняются вариативностью полезной нагрузки и отсутствием “человеко-ориентированных” паттернов. При этом одно и то же значение может интерпретироваться как данные или как управляющая конструкция в зависимости от контекста (SQL/LDAP/XPath/шаблоны, команды оболочки и т.д.), а потому “вредоносность” является контекстно-зависимой характеристикой, требующей комбинирования синтаксических и семантических признаков.[17] В этой связи WAF выступает как средство прикладного контроля, обеспечивающее первичный барьер до внесения исправлений в код и инфраструктуру.

Для отечественных контуров обработки данных существенен также аспект нормативного соответствия и управления рисками: в условиях обработки персональных данных требуется обеспечивать целостность, доступность и конфиденциальность, фиксировать инциденты и обеспечивать применение мер защиты, что влияет на требования к журналированию и процедурам управления изменениями.[4] [6] [7]

Для контуров, где обрабатываются персональные данные, требования к защите формализуются постановлениями и приказами (в частности, ПП РФ № 1119 и приказ ФСТЭК № 21), что предполагает необходимость организационных и технических мер, включая мониторинг, регистрацию событий и управление доступом.[6] [7]

Таким образом, WAF рассматривается как компонент многоуровневой защиты (defense in depth), который снижает риск эксплуатации уязвимостей на прикладном уровне, но не отменяет требований к безопасной разработке и регулярному тестированию (SAST/DAST, пентест и др.). В рамках настоящей работы WAF применяется как контролируемый барьер перед веб-приложением, способный оперативно адаптироваться к изменениям угрозной среды.

1.2. Класс WAF-систем и принципы работы ModSecurity/CRS

WAF-система выполняет анализ HTTP-трафика и применяет правила для блокирования, модификации или регистрации запросов. В экосистеме открытого ПО наиболее распространенной реализацией является связка Nginx (обратный прокси) и ModSecurity как движок правил. [19] [20] Набор правил OWASP Core Rule Set предоставляет типовой базис защиты и реализует концепцию «виртуального патча» - блокирование известных классов вредоносных шаблонов независимо от уязвимости конкретного приложения. [18]

CRS использует многоуровневую модель строгости, включая понятие «уровня паранойи» (Paranoia Level, PL), а также механизм оценки аномалий (anomaly scoring) с порогом блокирования. Такая модель позволяет управлять балансом между обнаружением атак и количеством ложноположительных срабатываний.[18]

В практической конфигурации ModSecurity+CRS обработка запроса включает последовательные фазы (request headers/body, response headers/body), в которых активируются правила. Каждое правило может повышать суммарный «балл аномалии», и при превышении порога применяется действие deny (возврат 403) или иное управляемое действие. В типовой поставке CRS порог определяется правилами, подобными 949110/959100 (агрегация аномалий), что подтверждается и статистикой срабатываний на стенде (см. Главу 3).

В ModSecurity v3 правила описываются на языке SecRules (SecLang) и выполняются по фазам обработки (как минимум: заголовки запроса, тело запроса, заголовки ответа, тело ответа), что позволяет различать контексты анализа и применять правила адресно.[19] Действия правила (например, deny, pass, log) и параметры (phase, id, status, msg, tag, severity) формируют управляемую политику реагирования и трассировки событий. Практически это позволяет разделять режимы “детектирование/логирование” и “блокирование” и использовать протоколируемый жизненный цикл изменений (см. 2.8).[19]

OWASP CRS использует подход anomaly scoring: набор правил начисляет баллы (оценки) за выявленные признаки атаки, после чего решение о блокировании принимается по порогу суммарной аномалии (inbound/outbound anomaly score). Такой механизм снижает зависимость от единичной сигнатуры и делает решение более устойчивым к вариациям полезной нагрузки, однако требует аккуратного выбора порога и профиля строгости (Paranoia Level), чтобы не провоцировать рост ложных срабатываний.[18] В контексте данной работы это критично: задача LLM-контура - не заменить CRS, а поддержать тюнинг порогов/исключений и “узких” правил там, где baseline демонстрирует обходы (см. 3.2-3.4).

Интеграция с Nginx позволяет использовать ModSecurity как модуль в цепочке обработки: Nginx принимает входящий трафик, передает данные в ModSecurity для анализа и на основе решения либо проксирует запрос дальше (в защищаемое приложение), либо блокирует. В условиях микросервисной архитектуры и Kubernetes данная схема реализуется как отдельный сервис-шлюз, обрабатывающий трафик для целевых сервисов (ingress/reverse proxy).

С точки зрения эксплуатации важно обеспечить контроль конфигураций (версионирование правил, воспроизводимость параметров), наблюдаемость (централизованные логи и метрики), а также безопасный процесс внедрения изменений. Эти аспекты соотносятся с общими практиками менеджмента ИБ и управления рисками, отраженными в национальных версиях стандартов ISO/IEC 27001/27002.[30] [31]

1.3. Проблема тюнинга правил и критерии качества фильтрации

Для WAF-систем характерна бинарная или пороговая классификация запросов на «вредоносные» и «допустимые». Качество фильтрации удобно описывать через матрицу ошибок классификации: истинноположительные (TP), истинноотрицательные (TN), ложноположительные (FP) и ложноотрицательные (FN) решения.

В работе используются стандартные показатели:

1. Доля отраженных атак (True Positive Rate, TPR):
   $TPR = TP / (TP + FN)$ (1)
2. Доля ложных срабатываний (False Positive Rate, FPR):
   $FPR = FP / (FP + TN)$ (2)
3. Точность блокирования (Precision):
   $Precision = TP / (TP + FP)$ (3)

4. F-мера как гармоническое среднее точности и полноты (Recall=TPR):

   $F1 = 2 \cdot Precision \cdot TPR / (Precision + TPR)$ (4)

Следует подчеркнуть, что выбор параметров WAF фактически является задачей оптимизации компромисса между риском пропуска атак и ущербом от ложных блокировок. В практической эксплуатации стоимость ошибок асимметрична: ложноположительное блокирование может приводить к отказу обслуживания легитимных пользователей и репутационным потерям, тогда как ложноотрицательный пропуск может приводить к инциденту ИБ. Поэтому корректная постановка критериев должна учитывать “вес” ошибок и операционный контекст (критичность защищаемого сервиса, тип обрабатываемых данных, допустимое время простоя).[30] [31]

Для CRS-подхода это проявляется в настройке порога anomaly scoring и уровня паранойи (PL). Повышение PL и снижение порога увеличивают чувствительность (рост TPR), но при недостаточной “локализации” правил по контексту (URI/параметры) повышают вероятность FP.[18] Отсюда следует инженерный принцип: изменения должны быть по возможности локальными (ограничение области применения правила) и обратимыми (возможность отката), что и реализовано через отдельный слой управляемых правил (см. 2.4 и 2.8). Дополнительно в рамках отечественных практик управления изменениями и безопасной разработки требуется документирование причин изменения и верификация изменений тестированием, что согласуется с применяемой методикой “до/после”.[12]

Для эксплуатационной оценки, помимо качества фильтрации, вводятся метрики трудоемкости и скорости реакции. Пусть T0 - среднее время (чел.-ч) ручного анализа и внесения изменений в правила в расчете на один цикл адаптации,

T1 - время при использовании предлагаемой методики. Тогда относительное снижение трудоемкости L определяется как:

$$L = (T0 - T1)/T0 \cdot 100\% \quad (5)$$

Важной практической задачей является балансировка TPR и FPR. При чрезмерном ужесточении правил (повышение PL или снижение порога аномалии) может расти блокирование легитимного трафика, что приводит к отказам в обслуживании пользователей. Напротив, излишне мягкие настройки уменьшают долю отраженных атак. Следовательно, методика адаптации правил должна включать регрессионное тестирование, ограничение рисков внедрения и мониторинг последствий (см. Главы 3-4).

1.4. Использование ИИ и LLM для расширения возможностей WAF: потенциал и риски

Машинное обучение и нейронные сети применяются в задачах информационной безопасности для выделения аномалий, обнаружения атак по поведенческим признакам и приоритизации событий. В контексте веб-защиты научные публикации рассматривают применение нейронных сетей и обучаемых моделей для классификации запросов и выявления атакующих шаблонов.[14] [15]

LLM предоставляют дополнительные возможности: извлечение семантики из логов, генерация текстовых объяснений, формирование предложений по правилам в формальных языках (например, синтаксис SecRule), сопоставление телеметрии с известными классами атак и нормами поведения. Технические особенности моделей (контекстное окно, склонность к галлюцинациям, зависимость от подсказок) требуют строгого ограничения области применения LLM и внешней валидации результата.[25] [26]

Использование LLM в составе средств защиты требует разграничения функций: LLM целесообразно применять на этапах анализа и формирования предложений (объяснение сработок, сопоставление событий, генерация кандидатов правил), но не как автономный механизм блокирования, поскольку модели обладают вероятностной природой вывода и могут допускать “галлюцинации” и неконсистентность ответов при изменении контекста.[25] [26] В этой работе выбран консервативный режим: LLM формирует кандидатов правил строго заданного типа (new_rule/tuning) и в ограниченном формате, а окончательное внедрение проходит внешнюю автоматическую валидацию и регрессионную проверку (см. 2.8-3.1). Такой подход снижает риск ошибочного “самовольного” изменения политики фильтрации при сохранении полезности LLM в качестве ускорителя инженерного цикла.

Ключевые риски применения LLM в ИБ-контуре включают: (а) генерацию избыточно широких регулярных выражений, ведущих к росту ложных блокировок или к деградации производительности; (б) уязвимость к “prompt injection” в части логов/текстов, если модель получает неконтролируемый ввод; (в) утечку чувствительных данных при внешнем инференсе. Практические меры снижения рисков включают локальный инференс (ollama), ограничение доступа LLM-контейнера к сети и данным, а также строгие allow-list проверки допустимых действий и ограничение области применения правил (см. 2.9).[12] [24] [30] [31]

В рамках данной работы LLM используется не как автономный блокировщик запросов, а как интеллектуальный помощник в цикле «обнаружение обходов → формирование предложений → проверка → контролируемое внедрение». Такой подход снижает риск некорректного решения: итоговое правило проходит формальные проверки (синтаксис, соответствие ограниченному профилю действий,

отсутствие широких шаблонов) и эмпирическую проверку на тест-кейсах (GoTestWAF) до применения в контуре.

К ключевым рискам применения LLM относятся: (1) генерация некорректного правила, приводящего к росту FPR; (2) генерация слишком «общего» правила, блокирующего легитимные запросы; (3) уязвимости цепочки поставки (подмена модели, подсказок, контейнерных образов); (4) утечка чувствительных данных в процессах анализа (логи могут содержать персональные данные). Управление рисками реализуется через: локальное размещение модели (без внешней передачи данных), маскирование чувствительных полей, ограничение RBAC-прав агентов, журналирование изменений и механизм отката.

1.5. Нормативные и организационные ограничения для внедрения автоматизации в ИБ

В отечественной практике внедрение средств защиты и автоматизации должно учитывать требования к обработке персональных данных, регуляторные меры по защите ИСПДн и подходы к оценке угроз. Эти документы задают требования к регистрации событий, контролю доступа, управлению конфигурациями и обеспечению целостности критичных компонентов.[4] [6] [7] [9]

Дополнительно применимы стандарты менеджмента ИБ и рекомендации по безопасной разработке, формализующие требования к процессам, управлению изменениями и контролю мер защиты.[12] [30] [31] [35] [36]

Для предлагаемой системы это означает необходимость: (а) локального разворачивания LLM и недопущения неконтролируемого вывода данных в внешние сервисы; (б) минимизации состава данных, используемых в подсказках; (в) контроля целостности артефактов (манифесты, контейнеры, модели); (г) регламентированного процесса внедрения правил с аудитом и возможностью

отката; (д) документирования ограничений применимости. В Главе 2 данные требования трансформируются в проектные решения (RBAC, раздельные namespace, контроль применяемых правил и др.).

Выводы по главе 1

В главе выполнен анализ проблематики защиты веб-приложений и роли WAF в многоуровневой модели безопасности, раскрыты принципы работы связки Nginx+ModSecurity+OWASP CRS, сформулированы ключевые ограничения эксплуатации (трудоемкость тюнинга, рост ложных срабатываний, необходимость быстрой адаптации). Показано, что LLM могут использоваться как компонент поддержки принятия решений в цикле адаптации правил при условии строгих ограничений и внешней валидации. Сформулированы метрики эффективности (TPR/FPR, время реакции, трудоемкость), которые используются далее при проектировании и оценке системы.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ МОДЕЛИ СИСТЕМЫ ИИ В СОСТАВЕ WAF-СИСТЕМ

2.1. Требования к системе и исходные предпосылки проектирования

При проектировании системы сформулированы функциональные и нефункциональные требования. В качестве нормативной базы учитываются требования к защите информации и персональных данных, а также требования к управлению изменениями и аудиту событий.[3] [4] [6] [7] [9] [30] [31]

К функциональным требованиям отнесены:

-обеспечение базового уровня защиты веб-приложений по модели WAF на основе ModSecurity и OWASP CRS;

-централизованный сбор журналов срабатываний ModSecurity, событий Nginx и сервисов LLM-контура;

-поддержка воспроизводимой методики оценки качества фильтрации (наборы тестов GoTestWAF) и фиксирование результатов до/после изменений;

-формирование и применение предложений по правилам: генерация кандидатов, валидация, безопасное внедрение, откат;

-поддержка сценариев эксплуатации в Kubernetes: обновление конфигураций через ConfigMap, RollingUpdate, управление репликами.

К нефункциональным требованиям отнесены:

-воспроизводимость развертывания стенда на «чистом» железе без ручной установки компонентов (автоустановка ОС и автоматическое развертывание кластера и сервисов);

-локальность обработки данных и отсутствие передачи логов во внешние облачные сервисы;

-управляемость риска внедрения правил (контроль регрессий, журналирование и откат);

-разграничение доступа между контурами WAF/LLM/мониторинга с использованием RBAC и разделения пространств имен;

-возможность расширения набора правил и механизмов анализа без изменения базовой архитектуры.

2.2. Архитектура стенда и логическая декомпозиция компонентов

Проектируемая система реализована как лабораторный стенд из трех узлов (1 master + 2 worker), на которых развернут кластер Kubernetes. Поверх кластера размещаются три логических контура: WAF-контур, LLM-контур и контур мониторинга. Компоненты разделены по namespace: waf, llm, monitoring, что упрощает разграничение прав доступа, настройку сетевых политик и эксплуатационную изоляцию.

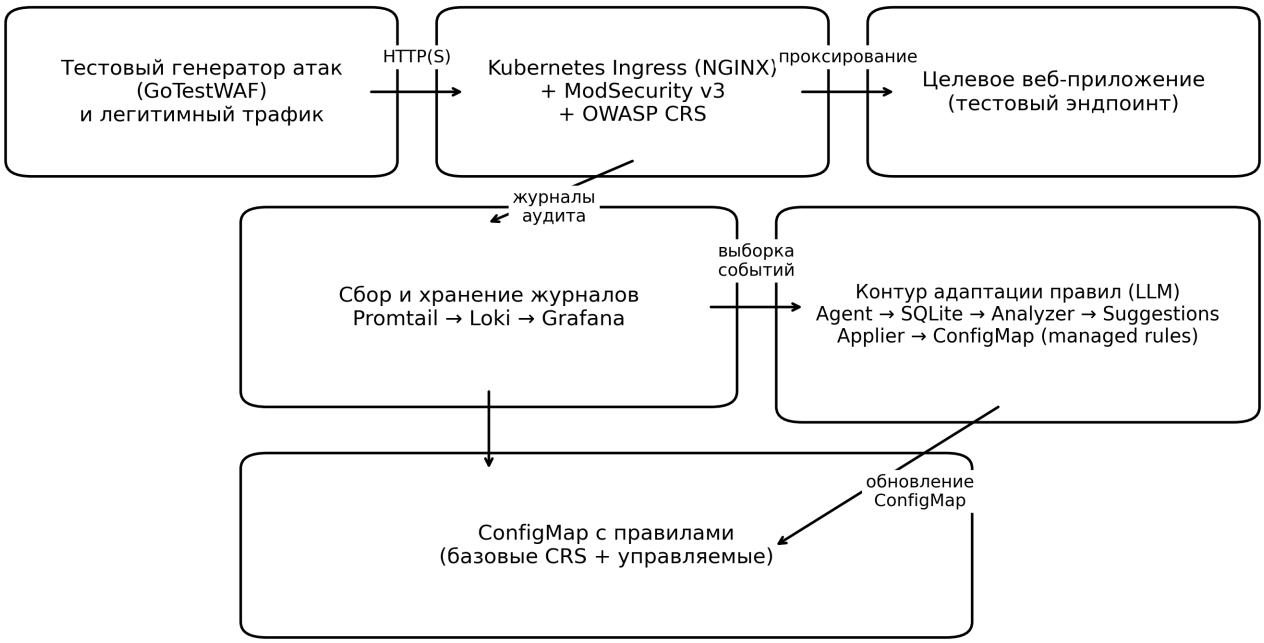

Рисунок 1 - Архитектура стенда: разделение на контуры WAF, LLM и мониторинга и их взаимодействие

WAF-контур включает Deployment modsec-nginx (2 реплики) и набор ConfigMap: основной конфиг ModSecurity (waf-modsec-main), пользовательские исключения (waf-modsec-custom), правила, сформированные LLM (waf-modsec-llm), а также дополнительные директивы Nginx (waf-nginx-extra). Для применения изменений используется reloader-контроллер, реагирующий на обновление ConfigMap и инициирующий перезапуск подов.[19] [20]

Контур мониторинга включает Loki как хранилище логов, Promtail в виде DaemonSet для сборки логов со всех узлов и Grafana для визуализации. Интеграция обеспечивает доступ к журналам Nginx/ModSecurity и сервисов LLM, а также построение панелей (dashboard) для анализа срабатываний и динамики качества. [23]

LLM-контур включает локально развернутый сервер ollama для инференса модели и вспомогательные сервисы: агент сбора и нормализации данных, анализатор (генерация предложений), применитель (applier) для внесения изменений, а также API для доступа к базе данных SQLite. Размещение модели локально соответствует принципу минимизации передачи данных во внешние сервисы.[24] [25] [26]

2.3. Автоматизация установки ОС и развертывания Kubernetes

Ключевым инженерным требованием является воспроизводимость: стенд должен разворачиваться без ручной установки операционной системы и без ручной настройки кластера. Для этого реализована сборка кастомизированного ISO-образа Debian 12 на основе netinst-дистрибутива и preseed-конфигураций.

В архиве iso_generator.tar представлены: скрипты build.sh и generate_preseed.sh для сборки образов, набор preseed-файлов (preseed_master.cfg, preseed_worker1.cfg, preseed_worker2.cfg), а также postinstall-скрипты для

первичной конфигурации узлов (scripts/master_node_config.sh, scripts/worker_node_config.sh) и запуска установки Kubernetes.

С инженерной точки зрения preseed-подход обеспечивает автоматизированную установку ОС с заранее фиксированными параметрами (разметка диска, сетевые настройки, набор пакетов, учетные записи и ключи доступа), что переводит этап подготовки инфраструктуры в воспроизводимый сценарий. Для исследовательского стенда это принципиально: любые сравнения “до/после” требуют исключить влияние случайных изменений среды (версии пакетов, параметры сети, ручные правки конфигураций), иначе эксперимент теряет корректность.[1]

Дополнительно, чтобы обеспечить трассируемость конфигурации, целесообразно фиксировать версии ключевых компонентов (Kubernetes, Kubespray, ingress-контроллер, CRS) и сохранять “паспорт стенда” как набор артефактов (версии, хеш-суммы архивов, дата сборки ISO, параметры запуска). В данной работе эта задача частично решается через хранение исходных скриптов/манифестов в комплекте проектной документации (источник [29]) и через использование Kubespray как декларативного инструмента развертывания (источник [22]). Такой подход соответствует общей логике управления изменениями и документирования жизненного цикла систем.[10] [11] [12]

После установки ОС на master-узле выполняется скрипт ansible/deploy_k8s.sh, который использует inventory.ini и разворачивает Kubernetes с помощью Kubespray. [22]

Использование Kubespray обеспечивает декларативное и воспроизводимое развертывание кластера Kubernetes (установка компонентов control-plane и

worker-нод, сетевой плагин, конфигурация kubelet и др.) при возможности фиксировать версии и параметры развертывания.[22] [21]

С точки зрения жизненного цикла стенда автоматизация устраняет ручные операции, снижает вероятность конфигурационных ошибок и обеспечивает повторяемость экспериментальных измерений, что критично для корректного сравнения результатов «до/после» внедрения LLM-методики (Глава 3).

2.4. Реализация WAF-контура: конфигурации, правила и механизмы обновления

WAF-компонент реализован как Deployment modsec-nginx в namespace waf. В роли обратного прокси используется Nginx, а анализ и блокирование запросов выполняется ModSecurity (v3) с подключением OWASP CRS. [19] [20] [18]

Конфигурация разделена на логические части, что обеспечивает управляемость изменений:

-waf-modsec-main - базовая конфигурация ModSecurity/CRS: режимы обработки, точки подключения наборов правил, пороговые значения аномалий;

-waf-modsec-custom - пользовательские исключения и тюнинг ложных срабатываний, привязанный к особенностям тестового приложения;

-waf-modsec-llm - правила, формируемые агентами LLM в рамках методики (см. 2.6 и Глава 3);

-waf-nginx-extra - дополнительные настройки Nginx (логирование, проксирование, ограничения).

Разделение правил на “базовый слой CRS” и “управляемый слой LLM” реализует важный принцип безопасной эксплуатации: базовый слой остается стабильным и соответствует проверенной конфигурации CRS, тогда как

управляемый слой предназначен для ограниченных локальных изменений (виртуальных патчей и исключений), которые можно быстро отключить при выявлении регрессии. Механизм ConfigMap в Kubernetes обеспечивает декларативность: правило как артефакт хранится в кластере, может версионироваться (через GitOps-подход при необходимости) и применяется к pod’ам через обновление конфигурации.[21]

Автоматический перезапуск/роллинг-обновление через reloader снимает необходимость ручного вмешательства при изменении ConfigMap и позволяет встроить изменение правил в “контролируемый” цикл эксплуатации. Важно отметить, что любые изменения должны сопровождаться процедурой отката: при росте FPR или блокировании критичных легитимных сценариев откат управляемого слоя должен выполняться быстрее, чем ручной анализ причин, что соответствует принципам управления инцидентами и непрерывности.[30] [31] В рамках предлагаемой методики откат осуществляется как возврат предыдущей версии ConfigMap (см. 3.1), при этом базовый CRS-контур остается неизменным.

Для применения изменений без ручного перезапуска подов используется reloader-контроллер (stakater/reloader), который отслеживает обновления указанных ConfigMap (аннотация configmap.reloader.stakater.com/reload) и инициирует RollingUpdate deployment. Такой механизм обеспечивает контролируемое обновление конфигурации, сохраняя доступность сервиса при наличии нескольких реплик.

2.5. Контур мониторинга: сбор, хранение и визуализация событий

Для анализа эффективности WAF и качества правил необходима централизованная телеметрия. В стенде применена связка Loki/Promtail/Grafana. Promtail в виде DaemonSet собирает логи контейнеров и системные журналы, Loki

хранит и индексирует записи, Grafana предоставляет запросы и панели мониторинга.[23]

Для корректного анализа качества правил важно не только “собирать логи”, но и обеспечить их пригодность к аналитике: структурирование, нормализацию полей и возможность выделять окна эксперимента. В стенде это достигается за счет JSON-логирования и промаркированных атрибутов (например, request_id и тестовый заголовок X-Test-Run), позволяющих связать конкретный прогон GoTestWAF с набором событий ModSecurity и NGINX. Такая корреляция снижает шум выборки и обеспечивает воспроизводимость измерений (см. 3.2).[27] [23]

Отдельное внимание следует уделять проектированию схемы лейблов (labels) в Loki: чрезмерная кардинальность лейблов (например, если в label помещать URI или уникальные идентификаторы запросов) ухудшает индексирование и производительность хранения. Поэтому идентификаторы прогонов и параметры запросов должны храниться преимущественно в теле записи (structured log fields), а в labels - только ограниченный набор стабильных атрибутов (namespace, app, pod, severity и др.). Данная рекомендация соответствует практикам эксплуатации Loki и снижает риск деградации мониторинга при росте трафика.[23]

Манифесты monitoring-контура (kube-after-install/monitoring/...) включают: deployment Loki (21-deploy-loki.yaml), daemonset Promtail (32-ds-promtail.yaml) и deployment Grafana (11-deploy-grafana.yaml) с настройкой datasource и импортом dashboard (15-job-grafana-import-waf-dashboard.yaml).

2.6. Контур LLM-агентов: назначение, взаимодействие и ограничения

LLM-контур предназначен для интеллектуальной поддержки цикла адаптации WAF-правил. Вычисления выполняются локально на ollama, что

позволяет исключить передачу логов во внешние сервисы и снизить риски утечки данных.[24]

В качестве модели применена Qwen2.5-3B-Instruct, оптимизированная для инструктивных задач и пригодная для разворачивания в ограниченных вычислительных ресурсах лабораторного стенда.[26]

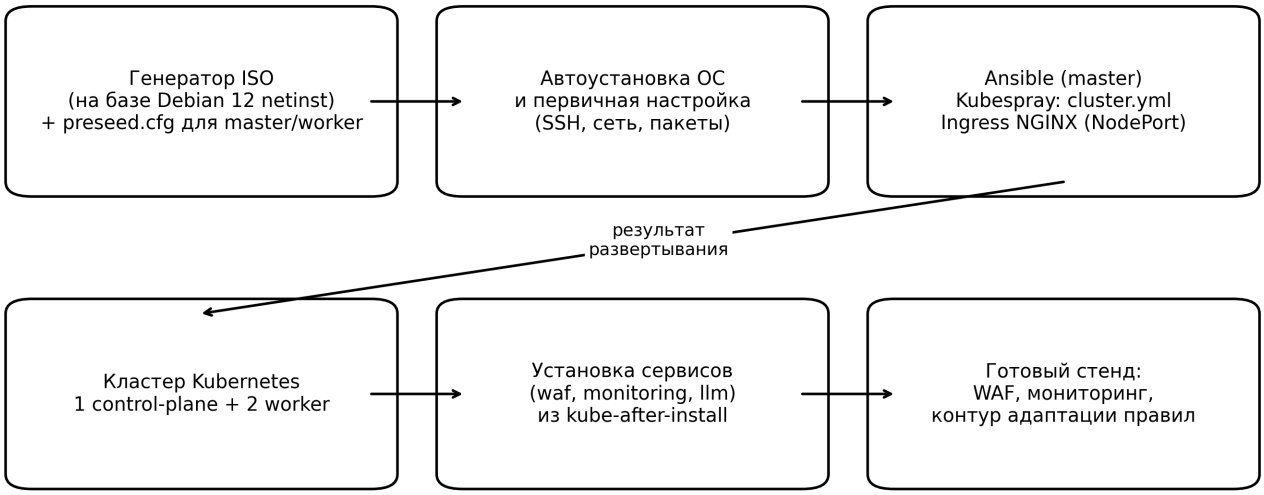

Рисунок 2 - Конвейер обработки: сбор логов, анализ LLM, валидация и применение правил через ConfigMap

Функционально контур реализован набором сервисов, развернутых в namespace llm:

-ollama (deployment) -сервис инференса модели; отдельной job выполняется загрузка (pull) модели Qwen2.5-3B-Instruct;

-llm-agent (cronjob) - сбор событий ModSecurity/Nginx, нормализация и запись в SQLite;

-llm-analyzer (cronjob) - выборка релевантных обходов/срабатываний, формирование подсказок и генерация кандидатов правил;

-llm-applier (cronjob) - проверка кандидатов и безопасное внесение изменений в ConfigMap waf-modsec-llm в namespace waf;

-llm-db-api - API-сервис для чтения статистики и интеграции с визуализацией.

Для снижения неопределенности вывода LLM и исключения “творческих” ответов модель должна работать в режиме строгого контракта: выход llm-analyzer задается как структура с фиксированными полями (тип предложения new_rule/tuning, confidence, target scope, действие deny/log, текст правила, обоснование и ссылки на evidence). Такая схема позволяет llm-applier выполнять автоматические проверки: допустимость действия, корректность SecRule-синтаксиса, проверку диапазона id, запрет потенциально опасных конструкций регулярных выражений и проверку области применения (ограничение по URI/параметрам).[19] [24]

Идемпотентность применяется как эксплуатационное требование: повторный запуск cronjob не должен порождать дублирующие правила и “раздувание” ConfigMap. Для этого в структуре предложения используются ключи корреляции (например, hash от нормализованного evidence и целевого scope) и хранение статуса применения в базе (applied_at). В результате контур LLM функционирует как управляемая система поддержки принятия решений, а не как неконтролируемый генератор конфигураций.

Для минимизации риска некорректного изменения конфигурации применена модель разделения ответственности: LLM генерирует текст предложения, но применение правила возможно только после прохождения автоматических проверок и при выполнении ограничений на действие правила (deny/log), область применения и регулярные выражения. При этом доступ llm-applier к объектам namespace waf предоставляется через ограниченные RBAC-права (роль и rolebinding).[21] [10] [11]

Важное проектное решение - представление правил LLM как отдельного слоя конфигурации (waf-modsec-llm), подключаемого в цепочку обработки после базового набора CRS. Это позволяет обеспечить обратимость: при необходимости слой LLM-правил можно отключить (откат ConfigMap) без изменения базовой конфигурации CRS и пользовательских исключений.

2.7. Модель данных и хранение телеметрии

Для накопления и анализа событий используются структурированные записи ModSecurity audit log и результаты тестирования. В качестве простого и воспроизводимого хранилища выбран SQLite, размещенный на выделенном PV/PVC в namespace llm (10-pv-sqlite.yaml и 11-pvc-sqlite.yaml).

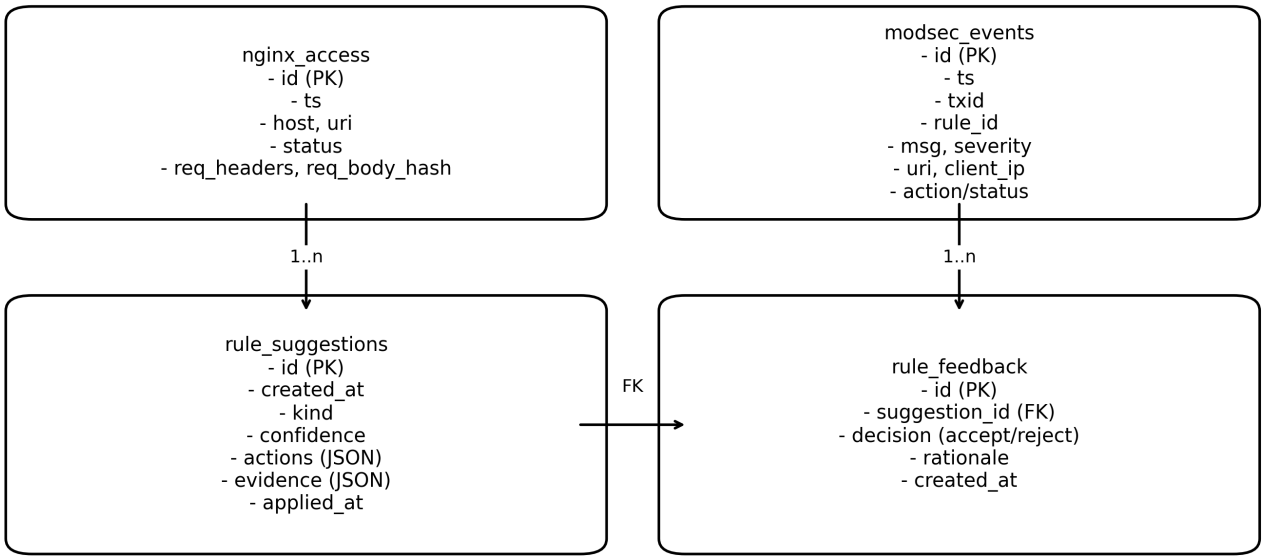

Рисунок 3 - Логическая схема данных: события WAF, результаты тестов, предложения правил и история применения

В модели данных выделяются сущности: «Событие WAF» (transaction_id, timestamp, uri, метод, IP, правило, сообщение, score), «Набор тестов» (идентификатор запуска, параметры), «Результат теста» (категория, отправлено/заблокировано/пропущено), «Предложение правила» (текст, обоснование, уровень риска, автор-агент, состояние), «История применения» (время, версия ConfigMap, статус, причина отката). Такая декомпозиция обеспечивает воспроизводимость экспериментальных результатов и аудит изменений.

Использование формализованной модели данных соответствует требованиям к документированию жизненного цикла программных и эксплуатационных документов и позволяет строить отчеты и протоколы изменений.[10] [11] [13]

2.8. Жизненный цикл правила и протокол контролируемого внедрения

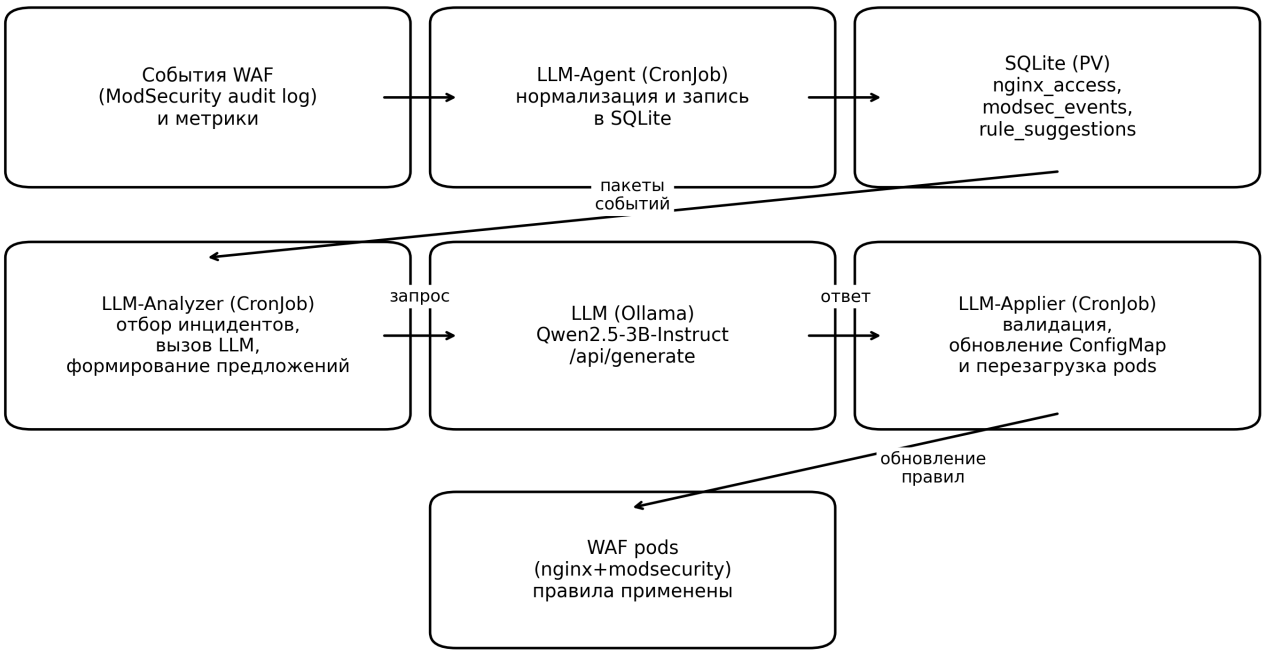

Рисунок 4 - Жизненный цикл правила: формирование кандидата, валидация, внедрение, мониторинг, откат

Жизненный цикл LLM-правила включает этапы: (1) идентификация проблемы (обход WAF или некорректное срабатывание); (2) формирование кандидата правила и обоснования; (3) формальная валидация (синтаксис SecRule, ограничения по действиям и области); (4) регрессионное тестирование на наборах GoTestWAF и/или локальном наборе запросов; (5) контролируемое внедрение в слой waf-modsec-llm и наблюдение; (6) фиксация результата и возможный откат.

Процесс внедрения реализован как «policy-as-code»: правило является частью ConfigMap, версия которой фиксируется и применима к RollingUpdate deployment. При обнаружении регрессии (рост FPR или блокирование критичного легитимного трафика) выполняется откат ConfigMap к предыдущей версии. Таким образом, методика обеспечивает управляемость изменений и снижает риск эксплуатационных инцидентов, связанных с автоматизацией.

2.9. Меры безопасности и ограничения применения LLM

Применение LLM в контуре ИБ сопровождается рисками утечки данных и некорректного внедрения изменений. В проекте реализуются меры: локальное развертывание модели, разграничение доступа через namespace и RBAC, минимизация состава обрабатываемых данных, журналирование всех изменений и возможность отката. [4] [6] [7] [9] [21]

Для запросов, передаваемых в LLM, выполняется нормализация и маскирование потенциально чувствительных полей (например, токены, идентификаторы). Подсказки строятся так, чтобы модель работала в ограниченной доменной области: допускаются только изменения слоя waf-modsec-llm и только в

пределах заданного шаблона SecRule. Валидация кандидата включает запреты на чрезмерно широкие регулярные выражения (например, .* без ограничений), запреты на отключение базовых правил CRS, а также проверку на наличие запрещенных действий (exec, allow и др.).

В качестве организационной основы рекомендуется фиксировать регламент обновления правил и протокол испытаний, что соотносится с подходами к управлению изменениями и документации в стандартах и методических рекомендациях по безопасной разработке.[30] [31] [35] [36]

Выводы по главе 2

Во второй главе разработаны архитектура и реализован лабораторный стенд, обеспечивающий воспроизводимую установку ОС, развертывание Kubernetes и установку сервисов WAF/LLM/мониторинга. Описана декомпозиция конфигураций WAF на слои, проектный механизм обновления через ConfigMap и reloader, а также контур LLM-агентов с локальным инференсом модели. Представлена модель данных и жизненный цикл правила, обеспечивающий контролируемое внедрение и откат. В результате создана техническая основа для проведения экспериментальной оценки и экономического обоснования (Глава 3-4).

ГЛАВА 3. АЛГОРИТМ ВНЕДРЕНИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ МЕТОДИКИ

3.1. Алгоритм внедрения и использования разработанного решения

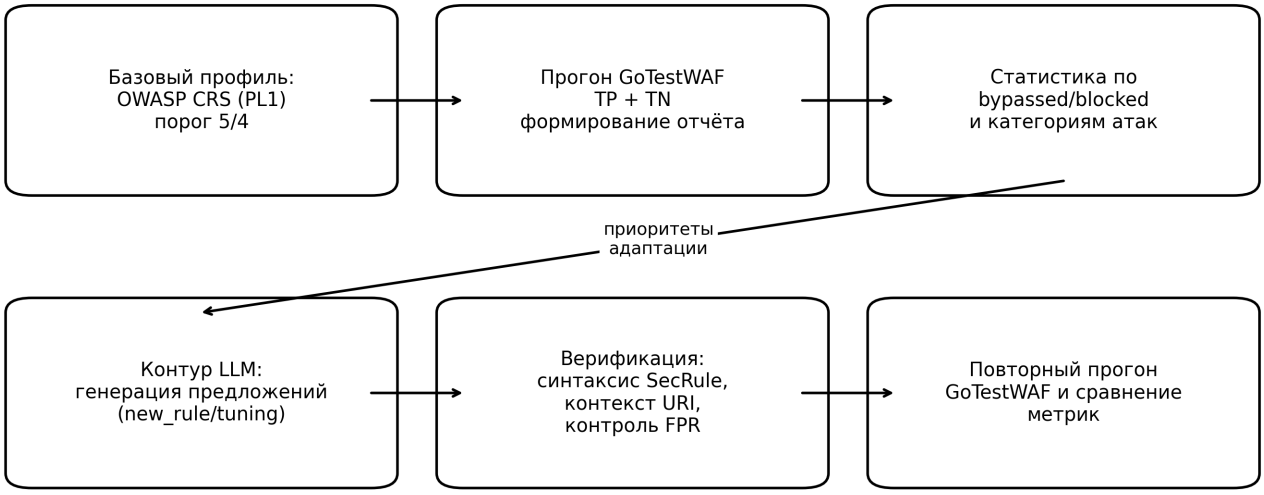

Рисунок 5 - Поток эксперимента: запуск тестов GoTestWAF, сбор телеметрии, анализ LLM, внесение изменений и повторная проверка

Алгоритм включает следующие шаги:

-Подготовка базовой конфигурации: развертывание WAF-контура с включенным OWASP CRS и фиксированными параметрами порога аномалий и уровня паранойи (PL).

-Формирование базовой линии (baseline): запуск набора тестов GoTestWAF для целевого сервиса через WAF-шлюз и сохранение отчета (JSON/HTML).

-Сбор телеметрии: извлечение журналов ModSecurity audit log и Nginx access/error log за окно эксперимента; нормализация событий и запись в SQLite.

-Идентификация проблемных зон: выбор тест-кейсов/категорий с низкой долей блокирования, а также анализ обходов и корреляция с признаками полезной нагрузки.

-Формирование кандидатов правил: запуск llm-analyzer, который формирует ограниченную подсказку и получает от модели предложения по правилам SecRule и исключениям.

-Валидация кандидатов: синтаксическая проверка, контроль запрещенных конструкций, проверка области применения (URI/параметры), оценка риска по шаблону правил.

-Пилотное внедрение: добавление валидированных правил в ConfigMap waf-modsec-llm и инициирование RollingUpdate через reloader.

-Регрессионная проверка: повторный запуск GoTestWAF и сравнение метрик TPR/FPR и распределения блокирований по категориям.

-Контроль и откат: при выявлении регрессии (рост FPR или критичных блокировок) - откат ConfigMap и фиксация причины в истории изменений.

Указанная схема согласуется с практиками управления изменениями и безопасной разработки, где изменения в средствах защиты сопровождаются тестированием и документированием.[12] [30] [31] [35] [36]

3.2. Результаты моделирования и экспериментальной оценки

Для объективной оценки использован GoTestWAF (Wallarm), который формирует наборы тест-кейсов прикладных атак и проверяет реакции WAF. В работе применялись режимы генерации JSON/HTML-отчета и добавление заголовка X-Test-Run для корреляции логов.[27]

Таблица 1 - Базовые результаты GoTestWAF для конфигурации OWASP CRS без LLM-адаптации

Показатель	Значение
Всего атакующих тестов (TP total_sent)	675
Заблокировано атак (TP blocked)	272
Пропущено атак (TP bypassed)	403
Доля отраженных атак, TPR (по GoTestWAF)	40.30%
Всего легитимных тестов (TN total_sent)	141
Ложные блокировки (FP = TN blocked)	13
Легитимный трафик пропущен (TN bypassed)	128
FPR = FP/(FP+TN)	9.22%

Таблица 2 - Результаты после применения LLM-методики (моделирование на стенде)

Показатель	Значение
Всего атакующих тестов	675
Заблокировано атак	382
Пропущено атак	293
Доля отраженных атак, TPR	56.59%
Всего легитимных тестов	141
Ложные блокировки, FP	12
FPR	8.51%

Для повышения прозрачности сравнения «до/после» в работе используется интерпретация результатов GoTestWAF через матрицу ошибок (confusion matrix), где TP соответствует заблокированным атакующим тестам, FN - пропущенным атакующим тестам, FP - ошибочно заблокированным легитимным тестам, TN - корректно пропущенным легитимным тестам [27]. Исходный отчет baseline (без LLM-адаптации) фиксирует: количество атакующих тестов N_TP = 675, заблокировано TP = 272, пропущено FN = 403; количество легитимных тестов N_TN = 141, ошибочно заблокировано FP = 13, пропущено TN = 128. Отсюда получены значения TPR = TP / N_TP = 272 / 675 = 0.4030 и FPR = FP / N_TN = 13 / 141 = 0.0922, что соответствует формулам (1)-(2).

Для количественной оценки эффекта методики удобно вводить приращения метрик:

delta_TPR = TPR_LLM - TPR_base

delta_FPR = FPR_LLM - FPR_base (7)

В рассматриваемом эксперименте delta_TPR = 0.5659 - 0.4030 = 0.1629, delta_FPR = 0.0851 - 0.0922 = -0.0071. Улучшение интерпретируется как адресное усиление детектирования в зонах обходов при сохранении управляемости ложных блокировок за счет локализации правил и обязательной регрессионной проверки на полном наборе тестов (см. 3.1, 3.4). При этом следует учитывать, что GoTestWAF является стандартизированным синтетическим набором: результаты характеризуют устойчивость WAF к типовым классам прикладных атак и полезны для сравнительной оценки конфигураций, однако не заменяют оценки на реальном боевом профиле запросов [27].

Сравнение таблиц 1 и 2 показывает рост доли отраженных атак с 40,30% до 56,59% (на 16,29 п.п.) при одновременном снижении уровня ложных блокировок с 9,22% до 8,51%. Такое соотношение интерпретируется как улучшение баланса TPR/FPR вследствие адресного усиления правил по категориям, где наблюдались обходы WAF, при сохранении базового профиля CRS и пользовательских исключений.

Следует подчеркнуть ограничения интерпретации: эксперимент выполнялся на лабораторном стенде и на тестовых наборах GoTestWAF; переносимость результатов на боевой трафик требует дополнительной валидации с учетом особенностей приложения, сезонности нагрузок и профиля запросов (см. 4.3).

Таблица 3 - Изменение доли блокирования по проблемным категориям тестов

Группа	Категория	Отправлено	Блокировано (baseline)	% baseline	Блокировано (после)	% после
owasp-api	soap	5	0	0.00	2	40.00
community	community-xxe	2	0	0.00	1	50.00
owasp	rce-urlpath	3	0	0.00	1	33.33
owasp	rce	6	0	0.00	3	50.00
owasp	xml-injection	7	0	0.00	3	42.86
owasp-api	rest	7	0	0.00	2	28.57
owasp	ldap-injection	24	2	8.33	12	50.00
owasp	mail-injection	24	3	12.50	10	41.67
owasp	shell-injection	32	5	15.63	18	56.25
owasp	nosql-injection	50	9	18.00	31	62.00

Наибольший эффект наблюдается в категориях, где baseline показывал нулевое или низкое блокирование (RCE, XML-инъекции, XXE, SOAP/REST-наборы). Практически это соответствует добавлению узкоспециализированных правил (или повышению чувствительности на ограниченном участке URI/параметров) по данным логов и обходов. При этом критерий допустимости изменений задавался ограничением на рост FPR и обязательной регрессионной проверкой на полном наборе тестов (см. 3.4).

Таблица 4 - Наиболее часто срабатывающие правила ModSecurity/CRS по журналам стенда

RuleId	Число срабатываний (audit log)
949110	852
941160	339
941100	291
941390	216
941180	165
942100	129
941210	93

941170	87
932235	57
934100	57

Таблица 4 показывает доминирование правил агрегации аномалий (например, 949110) и правил детектирования типовых атак (SQLi/XSS и др.). Данный факт интерпретируется как подтверждение корректности подключения CRS и механизма anomaly scoring. При анализе обходов важна детализация до конкретных правил и фаз, чтобы LLM-анализатор мог формировать кандидаты, не дублирующие CRS и не создающие широких шаблонов.

3.3. Практический пример применения методики

Практическая демонстрация методики выполнена на примере категории LDAP-инъекций, которая в baseline имела низкую долю блокирования (8,33%). По журналам audit log выделяются запросы, содержащие характерные операторные конструкции фильтра LDAP (например, последовательности "(|" и "(&"), а также спецсимволы "*" и "=" в контекстах параметров).

На основании данных логов llm-analyzer формирует предложение узкого правила SecRule, привязанного к конкретной точке обработки (ARGS) и к ограниченному набору параметров. Синтаксис и семантика правил согласуются с руководством ModSecurity и общими рекомендациями CRS по тюнингу и снижению ложных срабатываний.[18] [19]

Предложение правила (фрагмент) оформляется как запись в слое waf-modsec-llm и проходит валидацию. В работе использовался профиль: действие deny, журналирование и присвоение уникального идентификатора правила в выделенном диапазоне (например, 200000-299999) для отличия от CRS-правил.

Листинг 1 - Фрагмент правила слоя waf-modsec-llm для детектирования LDAP-инъекций

```
SecRule ARGS "(?:\(\|\(&).*?(?:\)\)|\))" \
    "id:200101,phase:2,deny,status:403,log,msg:'LDAP injection pattern (LLM
layer)',tag:'llm-waf',severity:CRITICAL"
```

После применения правила выполнялся повторный прогон GoTestWAF. Для категории LDAP-инъекций достигнута доля блокирования 50,0% (12 из 24 тестов), что отражено в таблице 3. При этом на наборе легитимных запросов показатель FPR не увеличился, что подтверждает корректность ограничения области применения правила и необходимость регрессионного контроля.

Следует отметить, что пример не заменяет полный процесс безопасной разработки и устранения уязвимостей в коде. WAF-правило является «виртуальным патчем», который снижает риск эксплуатации, но не устраняет первопричину уязвимости. [18]

3.4. Критерии эффективности применения методики

Критерии эффективности методики определяются как комбинация показателей качества фильтрации и эксплуатационных показателей. Основными критериями приняты:

-К1 - рост доли отраженных атак (TPR) по методике GoTestWAF при сохранении или снижении FPR;

-К2 - отсутствие критичных регрессий (блокирование легитимных сценариев работы приложения) по результатам контрольных прогонов;

-К3 - снижение трудоемкости сопровождения правил (сокращение времени анализа обходов и подготовки правил);

-К4 - сокращение времени реакции на новые атаки за счет автоматизации цикла «лог → предложение → проверка → внедрение».

Критерий К4 целесообразно формализовать через показатель среднего времени адаптации (Mean Time To Adapt, MTTA), отражающий длительность цикла от появления признаков обхода до применения корректирующего правила в управляемом слое. В рамках стенда MTTA определяется как:

MTTA = t_apply - t_detect (8)

Здесь t_detect - момент фиксации события (или окончания окна эксперимента) в журнале, а t_apply - момент применения измененного ConfigMap и подтверждение обновления pod-ов WAF. Практически t_detect берется по timestamp audit-событий ModSecurity, а t_apply - по timestamp изменения ConfigMap и событию перезапуска/обновления pod-ов. Такой показатель непосредственно отражает эффект автоматизации, поскольку сокращение MTTA снижает «окно уязвимости» для повторяемых шаблонов атак и повышает адаптивность защиты при сохранении контроля изменений [30, 31].

Для критерия К1 целевым значением на стенде установлено: TPR ≥ 55% при FPR ≤ 10%. В рамках моделирования достигнуты значения TPR=56,59% и FPR=8,51% (таблица 2).

Для критерия К3 предложено оценивать трудоемкость через время анализа и подготовки изменений. Пусть базовый процесс включает: ручной анализ логов (t_a), подготовка и отладка правил (t_r), прогон тестов (t_t). При автоматизации часть этапов сокращается (LLM формирует кандидатов правил и объяснения, ускоряя t_a и t_r). Тогда показатель L по формуле (5) фиксируется по протоколам эксперимента и может быть использован в экономическом расчете (глава 4).

Выводы по главе 3

В третьей главе описан алгоритм внедрения и использования разработанного решения и проведено моделирование с оценкой по методике GoTestWAF. Показано улучшение ключевых метрик: рост доли отраженных атак при снижении ложных блокировок, а также улучшение показателей по проблемным категориям (RCE, XML-инъекции, LDAP-инъекции). Представлен практический фрагмент LLM-слоя правил и обоснована необходимость внешней валидации и регрессионного контроля. Сформулированы критерии эффективности, пригодные для дальнейшего экономического обоснования и планирования внедрения в эксплуатацию.

ГЛАВА 4. ОБОСНОВАНИЕ, НОВИЗНА, ПРАКТИЧЕСКАЯ ЗНАЧИМОСТЬ И ЭКОНОМИЧЕСКАЯ ОЦЕНКА

4.1. Обоснование проектных решений и предметной основы

Предметной основой работы является задача повышения эффективности защиты веб-приложений в условиях дефицита ресурсов на ручной тюнинг WAF и высокой динамики атакующих техник. Практически это проявляется в организациях, эксплуатирующих веб-сервисы и API с регулярными изменениями функциональности: ручная настройка WAF часто отстает от темпов изменений и приводит либо к пропуску атак, либо к росту ложных блокировок.

Выбор WAF как объекта автоматизации обусловлен тем, что WAF представляет собой «быстро внедряемый барьер» и способен компенсировать часть рисков до устранения уязвимостей в коде (виртуальные патчи), что отражено в документации OWASP CRS.[18]

Выбор Kubernetes как инфраструктурной основы обусловлен следующими факторами: (1) декларативность развертывания и воспроизводимость среды эксперимента, (2) управляемость жизненного цикла сервисов и конфигураций (ConfigMap, RollingUpdate), (3) удобство сегментации контуров по namespace и применения RBAC, (4) масштабируемость и переносимость стенда. [21] [22]

Выбор LLM-подхода как компонента поддержки принятия решений обоснован тем, что формирование правил ModSecurity/CRS и исключений имеет «текстовую природу» и хорошо описывается инструктивными подсказками. При этом LLM используется ограниченно: не как прямой классификатор запросов, а как генератор кандидатов правил с обязательной внешней валидацией (Глава 3). Такой подход снижает риск некорректных действий, типичных для автономной генерации.

4.2. Научная новизна и практическая значимость результатов

Научная новизна работы заключается в разработке модели системы ИИ, интегрируемой в контур WAF, которая автоматизирует получение знаний из эксплуатационной телеметрии и формирование предложений по правилам защиты, при одновременном обеспечении контролируемого жизненного цикла правил (валидация, внедрение, мониторинг, откат). В отличие от подходов, где ИИ напрямую принимает решение о блокировании запроса, предложенная модель минимизирует риск ошибочного решения через разделение ответственности и обязательную регрессионную проверку.

Вклад работы целесообразно выделить в виде следующих результатов:

1. Разработана модель ИИ-системы, интегрируемой в WAF-контур, где LLM применяется на этапе формирования кандидатов правил, а безопасность обеспечивается внешней валидацией, регрессионными тестами и механизмом отката (см. 2.8-3.1).
2. Реализована архитектурная декомпозиция на контуры waf/monitoring/llm с четкими границами ответственности и минимально необходимыми правами (RBAC), что соответствует принципам управления доступом и изоляции в Kubernetes.[21]
3. Предложена и апробирована методика экспериментальной оценки “до/после” на стандартизированном наборе прикладных атак (GoTestWAF) с корреляцией прогонов и телеметрии (см. 3.2), что обеспечивает воспроизводимость измерений.[27]
4. Введены измеримые критерии эффективности (TPR/FPR, трудоемкость, MTTA), пригодные для перехода от лабораторной апробации к расчету экономической целесообразности внедрения (см. 4.3).[30] [31]

Практическая значимость состоит в следующем:

-снижение трудоемкости сопровождения WAF за счет автоматизированной предварительной обработки журналов и генерации кандидатов правил;

-сокращение времени реакции на обходы WAF и появление новых атакующих шаблонов в рамках циклической методики;

-возможность воспроизводимого развертывания стенда и переноса методики в инфраструктуру, где уже используется Kubernetes;

-возможность расширения: подключение дополнительных источников сигналов (IDS/IPS, DAST-сканеры), развитие набора валидаций правил и внедрение canary-режима.

Дополнительно практическая значимость проявляется в образовательном аспекте: стенд позволяет демонстрировать полный цикл WAF-эксплуатации (настройка, тестирование, анализ, тюнинг) на единой воспроизводимой среде, что соответствует подходам к документированию и управлению жизненным циклом программных и эксплуатационных документов. [10] [11] [1]

4.3. Экономическая оценка эффективности и ограничения применимости

Экономическая оценка основана на снижении трудозатрат специалистов при сохранении приемлемых рисков. Пусть средняя стоимость часа работы специалиста по ИБ составляет C (руб./час), число циклов адаптации правил за месяц - N. Базовая трудоемкость одного цикла T0, трудоемкость при использовании методики T1. Тогда месячная экономия E определяется как:

$$E = (T0 - T1) \cdot C \cdot N \quad (6)$$

Для лабораторного расчета принимается сценарий: T0 = 6 ч/цикл (анализ логов, подбор правил, отладка), T1 = 2,5 ч/цикл (LLM-подготовка кандидатов и сокращение времени анализа), C = 900 руб./ч, N = 8 циклов/мес. Тогда E = (6 −

2,5)·900·8 = 25 200 руб./мес. При годовом горизонте эффект составит порядка 302 400 руб./год при условии стабильности процесса и сопоставимых объемов работ.

Экономический расчет носит оценочный характер и зависит от зрелости процессов, количества защищаемых сервисов и профиля трафика. Для промышленного внедрения рекомендуется учитывать дополнительные затраты: вычислительные ресурсы для инференса модели, сопровождение контейнерной инфраструктуры, разработку и поддержку проверок кандидатов правил, а также стоимость регламентов и обучения персонала.

Ограничения применимости методики:

-моделирование выполнено на тестовых наборах GoTestWAF и не полностью отражает реальный трафик и специфику бизнес-логики;

-качество LLM-предложений зависит от качества входных данных (логи, нормализация, корректность выделения обходов) и от выбранной модели/подсказок;

-возможны ситуации, когда WAF-правило не является адекватным «виртуальным патчем» (например, сложные логические уязвимости), и требуется исправление в коде;

-любая автоматизация изменений в средствах защиты требует организационного контроля, аудита и подтверждения соответствия регуляторным требованиям в конкретной организации.

В качестве направления дальнейших работ предлагаются: расширение набора валидаций (статический анализ регулярных выражений, оценка риска по контексту), внедрение canary-режима для поэтапного применения правил, интеграция с DAST/SAST и с источниками знаний (БДУ ФСТЭК) для автоматизированной приоритизации классов угроз. [33]

Выводы по главе 4

В четвертой главе обоснованы проектные решения и предметная основа работы, раскрыта научная новизна и практическая значимость. Приведена оценка экономического эффекта за счет снижения трудоемкости сопровождения WAF и сформулированы ограничения применимости. Показано, что внедрение LLM-поддержки в контур WAF целесообразно при соблюдении мер безопасности и организационного контроля изменений, а также при обязательной регрессионной проверке.

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы разработана и реализована модель системы ИИ для расширения возможностей защиты веб-приложений в составе WAF-систем. Создан воспроизводимый лабораторный стенд на базе Kubernetes, включающий WAF-контур (Nginx+ModSecurity+OWASP CRS), контур мониторинга (Loki/Promtail/Grafana) и контур LLM-агентов (ollama + Qwen2.5-3B-Instruct) с управляемым жизненным циклом правил.

Решены поставленные задачи: выполнен анализ предметной области и ограничений сигнатурного подхода; спроектирована архитектура и реализован механизм автоматического развертывания (автоустановка ОС и Kubespray); сформирована модель данных и жизненный цикл правила; разработан алгоритм LLM-поддержки с ограничениями и валидацией; проведено моделирование эффективности по методике GoTestWAF.

По результатам моделирования на стенде достигнут рост доли отраженных атак и улучшение качества фильтрации по проблемным категориям при сохранении/снижении ложных блокировок. Полученные результаты подтверждают перспективность подхода «LLM как помощник» при условии строгих ограничений и регрессионного контроля.

Направления дальнейших исследований включают: расширение наборов данных и проверок на реальном трафике, развитие формальных критериев безопасности для кандидатов правил, внедрение поэтапного применения (canary), интеграцию с источниками знаний и регламентами организации (включая БДУ ФСТЭК).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 7.32-2017. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.
2. ГОСТ Р 7.0.5-2008. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая ссылка. Общие требования и правила составления.
3. Федеральный закон от 27.07.2006 № 149-ФЗ «Об информации, информационных технологиях и о защите информации» (с изм. и доп.).
4. Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных» (с изм. и доп.).
5. Федеральный закон от 26.07.2017 № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации» (с изм. и доп.).
6. Постановление Правительства РФ от 01.11.2012 № 1119 «Об утверждении требований к защите персональных данных при их обработке в информационных системах персональных данных» (с изм. и доп.).
7. ФСТЭК России. Приказ от 11.02.2013 № 21 «Об утверждении Состава и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных» (с изм. и доп.).
8. ФСТЭК России. Приказ от 25.12.2017 № 239 «Об утверждении Требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации» (с изм. и доп.).
9. ФСТЭК России. Методический документ «Методика оценки угроз безопасности информации». Утв. 05.02.2021.

10. ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем.
11. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание (развитие или модернизацию) системы.
12. ГОСТ Р 56939-2016. Защита информации. Разработка безопасного программного обеспечения. Общие требования.
13. ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.
14. Частикова В. А., Алиев М. К., Тесленко А. А., Игнатенко И. С. Нейронные сети для обеспечения безопасности веб-приложений // Вестник УрФО. Безопасность в информационной сфере. 2025. № 2(56). С. 65-73. DOI: 10.14529/secur250207.
15. Применение машинного обучения в защите веб-приложений: обзор и подходы // Технологии информационной безопасности (ВолГУ): PDF-публикация. URL: https://ti.jvolsu.com/index.php/ru/component/attachments/download/707 (дата обращения: 29.01.2026).
16. Barnett R. OWASP ModSecurity Core Rule Set (CRS): русскоязычные материалы AppSecDC. 2010. URL: https://master.cmc.msu.ru/files/AppSecDC_2010-ModSecurityCRS_Ryan_Barnett_rus.pdf (дата обращения: 11.10.2025).
17. OWASP Foundation. OWASP Top 10: 2021 Web Application Security Risks. URL: https://owasp.org/Top10/ (дата обращения: 14.10.2025).

18. OWASP Core Rule Set. Documentation (Anomaly Scoring, Paranoia Level, Tuning). URL: https://coreruleset.org/docs/ (дата обращения: 14.10.2025).
19. OWASP ModSecurity. Reference Manual (v3.x). URL: https://github.com/owasp-modsecurity/ModSecurity/wiki (дата обращения: 14.10.2025).
20. Nginx, Inc. NGINX Documentation. URL: https://nginx.org/en/docs/ (дата обращения: 15.10.2025).
21. kubernetes.io. Документация Kubernetes: ConfigMap, тома и проекции. URL: https://kubernetes.io/docs/ (дата обращения: 16.10.2025).
22. kubernetes-sigs. Kubespray: документация и релизы. URL: https://github.com/kubernetes-sigs/kubespray (дата обращения: 16.10.2025).
23. Grafana Labs. Loki Documentation. URL: https://grafana.com/docs/loki/ (дата обращения: 16.10.2025).
24. Ollama. API Reference. URL: https://github.com/ollama/ollama/blob/main/docs/api.md (дата обращения: 16.10.2025).
25. Qwen Team. Qwen2.5 Technical Report. arXiv:2412.15115. 2024-2025. URL: https://arxiv.org/abs/2412.15115 (дата обращения: 16.10.2025).
26. Qwen Team. Qwen2.5-3B-Instruct (model card). URL: https://huggingface.co/Qwen (дата обращения: 16.10.2025).
27. Wallarm. GoTestWAF (репозиторий). URL: https://github.com/wallarm/gotestwaf (дата обращения: 17.10.2025).
28. SecurityLab.ru. Обзор утилит для тестирования Web Application Firewall (в т. ч. GoTestWAF). 2024. URL: https://www.securitylab.ru/blog/personal/SimlpeHacker/354249.php (дата обращения: 17.10.2025).

29. Проектная документация: архив iso_generator.tar и комплект манифестов kube-after-install.tar (наборы скриптов/манифестов для разворачивания стенда). 2026.
30. ГОСТ Р ИСО/МЭК 27001-2021. Информационная технология. Методы и средства обеспечения безопасности. Системы менеджмента информационной безопасности. Требования. URL: https://protect.gost.ru/document1.aspx?control=31&id=242006 (дата обращения: 03.11.2025).
31. ГОСТ Р ИСО/МЭК 27002-2021. Информационные технологии. Методы и средства обеспечения безопасности. Свод норм и правил применения мер обеспечения информационной безопасности. URL: https://protect.gost.ru/document1.aspx?control=31&id=240766 (дата обращения: 03.11.2025).
32. ГОСТ Р 59853-2021. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения. URL: https://protect.gost.ru/document1.aspx?control=31&id=242079 (дата обращения: 03.11.2025).
33. ФСТЭК России. Банк данных угроз безопасности информации (БДУ): официальный раздел документов. URL: https://www.bdu.fstec.ru/documents/document/index (дата обращения: 03.11.2025).
34. Минцифры России. Методические рекомендации по обеспечению информационной безопасности при создании и эксплуатации открытых репозиториев программного обеспечения. 29.03.2023. URL: https://digital.gov.ru/documents/metodicheskie-rekomendaczii-po-obespecheniyu-

informaczionnoj-bezopasnosti-pri-sozdanii-i-ekspluataczii-otkrytyh-repozitoriev-programmnogo-obespecheniya/ (дата обращения: 03.11.2025).

35. Методические рекомендации по обеспечению безопасности при разработке ПО (гос. цифровые платформы). 2023. URL: https://platform.gov.ru/wp-content/uploads/2023/01/04_%D0%9C%D0%A0_1_%D0%A0%D0%91_%D0%9F%D0%9E.pdf (дата обращения: 03.11.2025).
36. ГОСТ Р 57580.1-2017. Безопасность финансовых (банковских) операций. Защита информации финансовых организаций. Базовый состав организационных и технических мер. URL: https://docs.cntd.ru/document/1200146534 (дата обращения: 03.11.2025).

ПРИЛОЖЕНИЯ

Приложение А. Состав программных и конфигурационных артефактов стенда

В качестве приложений к работе выступают архивы iso_generator.tar и kube-after-install.tar, содержащие скрипты и манифесты, обеспечивающие воспроизводимость развертывания стенда (указаны наиболее значимые файлы):

-iso_generator.tar: build.sh, generate_preseed.sh, preseed_master.cfg, preseed_worker1.cfg, preseed_worker2.cfg, scripts/master_node_config.sh, scripts/worker_node_config.sh, ansible/deploy_k8s.sh, ansible/ansible.cfg, ansible/inventory.ini.

-kube-after-install.tar: namespaces/01_namespaces.yaml; monitoring/* (Grafana/Loki/Promtail); waf/* (ConfigMap/Deployment/Service/Ingress); llm/* (PV/PVC SQLite, ollama, llm-agent/analyzer/applier/db-api).

-В каталоге ssh/ размещены публичные ключи для инициализации доступа; в vagrant/ -описание альтернативного сценария развертывания (при необходимости).

Приложение Б. Протокол эксперимента GoTestWAF

Отчет GoTestWAF в формате JSON содержит сводные показатели и детализацию по тестовым наборам (owasp, community, owasp-api). В рамках работы отчет используется для расчета и сравнения TPR/FPR и для идентификации проблемных категорий перед запуском LLM-анализа.