ПРОБЛЕМЫ ТЕОРЕТИЧЕСКОЙ ИНФОРМАТИКИ, ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И СИСТЕМЫ

УДК 004.9:338.48

РАЗРАБОТКА TELEGRAM-БОТА ДЛЯ ФОРМИРОВАНИЯ ПЕРСОНАЛИЗИРОВАННЫХ ТУРИСТИЧЕСКИХ МАРШРУТОВ ПО ВОРОНЕЖУ

Т. В. Азарнова, Е. С. Макогонова

Воронежский государственный университет

Аннотация. В статье рассматривается проблема автоматизации планирования туристических маршрутов для города Воронежа. Предложена архитектура интеллектуального Telegram-бота, способного формировать индивидуальные экскурсионные маршруты. Научно-технический вклад работы заключается в исследовании и адаптации алгоритмов рекомендаций для узкой предметной области, а также в проектировании масштабируемой архитектуры программного обеспечения. В результате исследования сформулированы технические требования к системе и обоснован выбор технологического стека для ее реализации, включая фреймворк Aiogram и СУБД MySQL. Разрабатываемое решение направлено на повышение туристической привлекательности города за счет предоставления современных цифровых сервисов.

Ключевые слова: Telegram-бот, персонализация, туристические маршруты, алгоритмы рекомендаций, контентная фильтрация, архитектура системы, Воронеж.

Введение

Современный этап развития туристической индустрии характеризуется растущим спросом на персонализированные услуги. Традиционные групповые экскурсии уступают место индивидуальным форматам, позволяющим учитывать личные интересы, временные рамки и физические возможности туристов [1]. Город Воронеж, обладающий значительным историко-культурным потенциалом, сталкивается с проблемой недостаточной автоматизации процессов туристического ориентирования. Существующие решения, такие как универсальные картографические сервисы, не предоставляют функционала для автоматического построения тематических маршрутов, что создает существенные сложности для самостоятельных туристов.

Целью данного исследования является анализ, проектирование и разработка архитектуры интеллектуальной системы, способной формировать персонализированные туристические маршруты по г. Воронежу с использованием платформы Telegram. Научно-исследовательская составляющая работы заключается в исследовании применимости алгоритмов контентной фильтрации для задач маршрутизации и в проектировании оптимальной архитектуры системы. Для достижения поставленной цели решаются следующие задачи: анализ предметной области и существующих аналогов; разработка алгоритма персонализации на основе контентной фильтрации; проектирование модульной архитектуры системы; обоснование выбора технологий реализации.

1. Анализ предметной области и постановка задачи

1.1. Современное состояние рынка туристических сервисов

Проведенный анализ показал, что несмотря на наличие множества туристических приложений, их можно разделить на две основные категории: справочники с информацией о достопримечательностях (TripAdvisor) и картографические сервисы (Яндекс.Карты). Первые предоставляют обширные базы данных, но не предлагают инструментов для построения связных маршрутов. Вторые оптимизированы для логистики, но не учитывают тематические предпочтения пользователей [2].

Для г. Воронежа ситуация усугубляется отсутствием специализированных цифровых продуктов, ориентированных именно на его туристическую специфику. Существующие путеводители зачастую представляют собой статичные списки достопримечательностей без учета их взаимного расположения и логистических возможностей для построения эффективных пешеходных маршрутов.

1.2. Постановка задачи разработки

Основной задачей разработки является создание системы, способной на основе ограниченного набора входных параметров (интересы пользователя, доступное время, начальная точка) сгенерировать оптимальный персонализированный маршрут. Формально задача может быть представлена в виде:

Пусть $P = \{p_1, p_2, ..., p_n\}$ — множество достопримечательностей, каждая из которых характеризуется набором атрибутов $A_i = \{a_{i1}, a_{i2}, ..., a_{im}\}$ (историческая ценность, архитектурный стиль, временной интервал для посещения и т. д.). Пользовательский запрос Q включает подмножество релевантных атрибутов $A_q \subset A$ и ограничения C (время, бюджет). Тогда задача системы — найти упорядоченное подмножество $R \subset P$, максимизирующее функцию релевантности $F(R, Q)$ при соблюдении ограничений C.

2. Методы и алгоритмы персонализации

2.1. Выбор подхода к персонализации

Для решения поставленной задачи был проведен сравнительный анализ методов рекомендательных систем. Коллаборативная фильтрация, показывающая высокую эффективность в электронной коммерции [3], требует значительного объема данных о поведении пользователей, что делает ее неприменимой на начальном этапе работы системы. Алгоритмы, основанные на машинном обучении, также требуют репрезентативной обучающей выборки.

В связи с этим для первоначальной реализации был выбран метод контентной фильтрации, который позволяет строить рекомендации на основе сравнения атрибутов контента (достопримечательностей) с профилем пользовательских предпочтений. Данный подход не требует исторических данных и обеспечивает достаточное качество рекомендаций для нишевых систем [4].

2.2. Алгоритм формирования маршрута

На первом этапе осуществляется сбор пользовательских предпочтений через интерактивное меню Telegram, где пользователь выбирает интересующие категории (архитектура, музеи, парки, религиозные объекты) и указывает доступное временное окно. После этого выполняется первичный отбор достопримечательностей из базы данных по критерию пересечения тегов объектов с выбранными пользователем категориями.

На следующем этапе для каждого отобранного объекта вычисляется вес релевантности на основе количества совпадающих тегов. Затем система упорядочивает отобранные объекты с учетом их географической близости для минимизации времени перемещения между ними. Завершающим этапом является валидация по времени, когда проверяется соответствие суммарного времени посещения и перемещения доступному временному интервалу пользователя.

Математически функция релевантности может быть выражена как:

$$rel(p_i, U) = \frac{\left|T(p_i) \cap T(U)\right|}{\left|T(U)\right|} \cdot w_{geo}(p_i, loc_U), \quad (1)$$

где $rel(p_i, U)$ — релевантность достопримечательности p_i для пользователя U;

$T(p_i)$ — множество тегов достопримечательности p_i;

$T(U)$ — множество тегов, выбранных пользователем U;

w_{geo} — весовой коэффициент, учитывающий расстояние от текущего местоположения пользователя loc_U.

Формула (1) позволяет количественно оценить соответствие объекта интересам пользователя, комбинируя тематическое сходство и географическую близость.

3. Архитектура системы и реализация

3.1. Технологический стек и компоненты системы

Для реализации системы выбран следующий технологический стек:

– Backend: Python 3.9+ с асинхронным фреймворком Aiogram для эффективной обработки запросов Telegram Bot API.

– База данных: MySQL для хранения атрибутивной информации о пользователях, достопримечательностях и маршрутах.

– Внешние API: Яндекс.Карты API для геокодирования и построения пешеходных маршрутов.

– Архитектура системы включает следующие модули:

– Модуль взаимодействия с Telegram — обработка входящих сообщений и команд.

– Модуль диалогового управления — управление сценарием опроса пользователя.

– Модуль рекомендаций — реализация алгоритма контентной фильтрации.

– Модуль работы с данными — ORM-взаимодействие с базой данных MySQL.

– Модуль картографической визуализации — генерация статических карт с маршрутом.

Включение схемы архитектуры (рис. 1) является необходимым, так как оно предоставляет наглядное и структурированное представление о компонентах системы и их взаимодействии. Это повышает техническую наглядность работы и облегчает восприятие сложной модульной структуры разрабатываемого решения.

3.2. Проектирование базы данных

Спроектирована реляционная модель данных в СУБД MySQL, включающая следующие основные сущности (табл. 1).

Таблица 1

Основные сущности базы данных

Сущность	Атрибуты	Назначение
User	user_id, telegram_id, preferences	Хранение данных пользователя
PointOfInterest	poi_id, name, description, coordinates, tags	Информация о достопримечательностях
Route	route_id, user_id, created_at, route_path	История сформированных маршрутов

Заключение

В результате проведенного исследования разработана архитектура интеллектуальной системы для формирования персонализированных туристических маршрутов по г. Воронежу на основе платформы Telegram. Ключевыми результатами работы, подчеркивающими ее научно-технический вклад, являются:

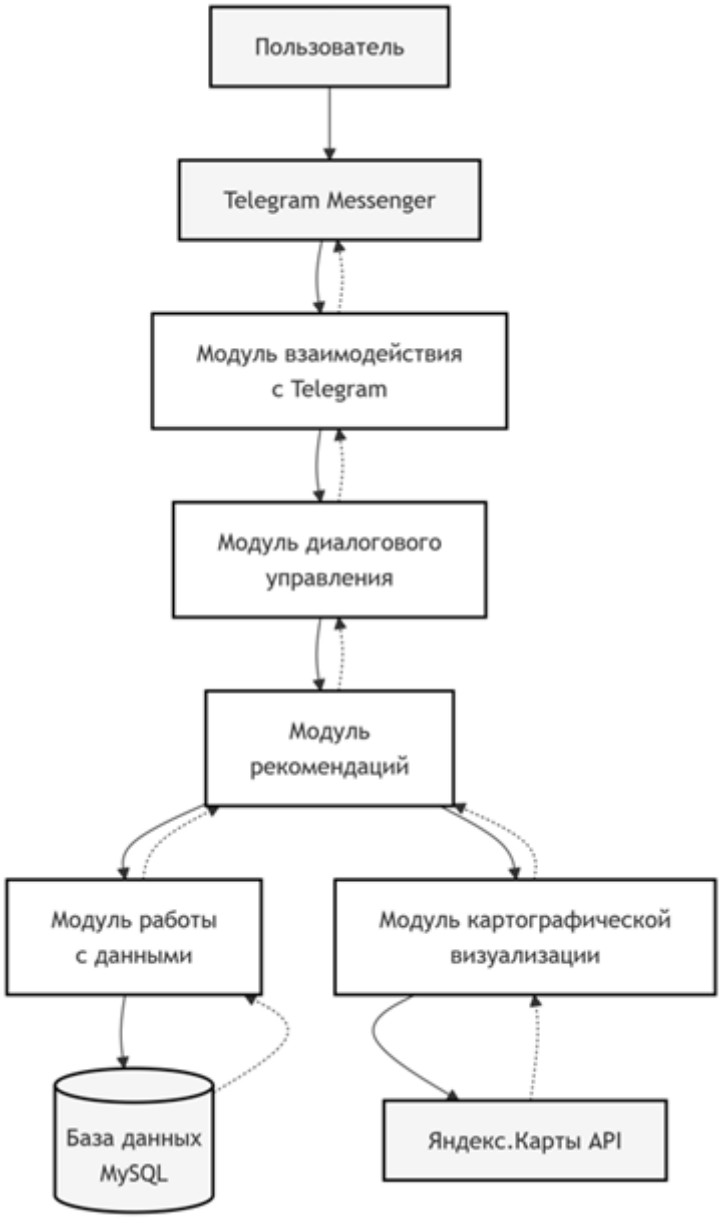

Рис. 1. Архитектура Telegram-бота для формирования туристических маршрутов

– Проведен анализ предметной области, выявивший отсутствие на рынке специализированных решений для г. Воронежа.

– Исследована и адаптирована для конкретной задачи методика контентной фильтрации, предложен и математически формализован алгоритм персонализации маршрутов.

– Спроектирована модульная архитектура системы, обеспечивающая масштабируемость и гибкость разработки, и обоснован выбор технологического стека, включая СУБД MySQL.

– Разработана структура базы данных, обеспечивающая хранение семантически размеченной информации о достопримечательностях.

Перспективы дальнейших исследований включают внедрение гибридных алгоритмов рекомендаций с элементами коллаборативной фильтрации по мере накопления пользовательских данных, а также интеграцию с системами бронирования билетов для обеспечения сквозного туристического опыта.

Литература

1. *Алексахин А. Н.* Цифровая трансформация в сфере туризма: тенденции и перспективы развития // Современные проблемы сервиса и туризма. – 2024. – № 2. – С. 15–24.

2. *Кудряшов А. А.* Развитие цифровых технологий сферы туризма в Российской Федерации // Туризм и индустрия гостеприимства. – 2024. – № 3. – С. 33–42.

3. *Восколович Н. А.* На пути к цифровому туризму // SPA Journal. – 2024. – Т. 12, № 1. – С. 5–14.

4. *Гущина Е. Г.* Цифровизация как новая реальность в управлении туристским рынком России // Информационные системы и технологии в экономике. – 2024. – Т. 8, № 4. – С. 78–86.

5. *Сереброва А. А.* Современные информационные технологии в туристической отрасли // Молодой ученый. – 2024. – № 3 (502). – С. 90–92.

6. *Чуваткин П. П., Левченко К. К.* Роль цифровых технологий в повышении инновационной активности туристских организаций // Технологии бизнеса и сервиса. – 2023. – Т. 9, № 2. – С. 55–64.

7. *Шкирова К. В.* Использование цифровых технологий в индустрии гостеприимства как инструмент популяризации и охраны культурного и природного наследия России // Институт наследия. – 2025. – № 1. – С. 23–31.

8. *Стародубцев И. М., Ширитов А. А.* Цифровизация в сфере туризма // Информационные технологии и математическое моделирование в управлении сложными системами. – 2024. – Т. 1(21). – С. 11–20.

9. *Савельева О. В.* Цифровизация в сфере туризма в Российской Федерации // Экономика и управление в сфере услуг. – 2023. – № 4. – С. 40–49.

10. *Тухватуллина В. Р.* Цифровизация в туризме: технологические подходы и персонализация туристических предложений // Журнал цифровой экономики. – 2025. – № 2. – С. 65–73.

УДК 004.056.5

БЕЗОПАСНОЕ ИСПОЛЬЗОВАНИЕ ЗАИМСТВОВАННЫХ КОМПОНЕНТОВ В ПРОЦЕССЕ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Е. Г. Батлук

Воронежский государственный университет

Аннотация. Статья посвящена решению проблемы обеспечения информационной безопасности при использовании заимствованных компонентов в разработке программного обеспечения. Проанализированы основные типы лицензий и риски, связанные с интеграцией сторонних решений. На основе ГОСТ Р 58412-2019 и современных исследований в области безопасности программного обеспечения предложена комплексная стратегия управления уязвимостями, объединяющая технологические и организационные меры защиты. Рассмотрены практические аспекты внедрения автоматизированного мониторинга уязвимостей, методов статического и динамического анализа кода, ведения реестра компонентов (SBOM) и контроля лицензионной совместимости.
Ключевые слова: заимствованные компоненты, информационная безопасность, управление уязвимостями, разработка программного обеспечения, открытое программное обеспечение, лицензионная совместимость, SBOM, статический анализ кода, динамический анализ кода, Software Composition Analysis.

Введение

Современная разработка программного обеспечения активно использует заимствованные компоненты, что ускоряет процесс и снижает затраты. Однако интеграция сторонних решений сопровождается серьёзными угрозами безопасности, поскольку компоненты могут содержать уязвимости, ошибки или вредоносный код.

В статье представлена стратегия управления уязвимостями при работе с заимствованными компонентами, включающая технические инструменты контроля, организационные меры и правовые аспекты использования стороннего кода.

1. Заимствованные компоненты в разработке ПО

Заимствованные компоненты — это элементы программного обеспечения (например, код, фреймворки, библиотеки, архитектурные шаблоны), созданные отдельно от основного проекта и интегрированные в него для ускорения разработки, сокращения затрат и повышения функциональности готового продукта. Часто речь идет о модулях, разработанных вне конкретной организации или проекта, но пригодных для повторного использования.

Заимствованные компоненты часто сопровождаются различными видами лицензий, регулирующими условия их использования, изменения и распространения. К основным типам лицензий относятся:

1. Открытые лицензии (Open Source). Большинство из них разрешает коммерческое использование, изменение и распространение программного обеспечения при условии указания авторства. Однако некоторые виды таких лицензий обязывают раскрывать исходный код производных продуктов.

2. Проприетарные лицензии. Эти лицензии накладывают ограничения на изменение, распространение и использование программного обеспечения в коммерческих продуктах.

3. Лицензии Creative Commons (CC). Данный тип лицензий применяется для графики, музыки, текстов, шрифтов и дизайна. Лицензии Creative Commons различаются условиями ис-

пользования: одни разрешают использовать контент только в некоммерческих целях, другие требуют, чтобы любые доработанные материалы распространялись на тех же условиях, что и оригинальные.

Согласно авторитетному исследованию Census III of Free and Open Source Software [3], проведённому экспертами Гарвардской школы бизнеса и Лаборатории инновационных наук Гарварда в партнёрстве с The Linux Foundation и OpenSSF, открытые компоненты стали неотъемлемой основой практически любого современного приложения. Зачастую эти компоненты выполняют не просто вспомогательную функцию, а образуют структурное ядро всей системы. Яркими примерами служат операционные системы на базе Linux, фреймворки вроде React или Django, а также среды выполнения, такие как Node.js.

2. Заимствованные компоненты как источник угроз информационной безопасности

Интеграция компонентов с открытым исходным кодом является общепринятой практикой в мировой индустрии разработки программного обеспечения. Данный подход сформировался как реакция на ограничения, присущие проприетарным продуктам коммерческих организаций, характеризующимся закрытым кодом и строгими условиями лицензирования.

Ключевые преимущества Open Source проявляются на различных уровнях разработки. На техническом уровне открытый код обеспечивает ускорение процесса создания программных продуктов за счет использования готовых, отлаженных решений. На экономическом уровне наблюдается значительное снижение затрат на разработку, поскольку исключается необходимость реализации стандартных функций с нуля. На архитектурном уровне открытые компоненты способствуют стандартизации подходов к проектированию и повышению надежности программных систем благодаря коллективной верификации кода. Тем не менее, использование таких компонентов обусловливает возникновение угроз информационной безопасности.

В частности, пункт 5.3.2 ГОСТ Р 58412-2019 идентифицирует угрозу внедрения уязвимостей программы путем использования заимствованных у сторонних разработчиков программного обеспечения уязвимых компонентов [1]. Угроза состоит в интеграции в разрабатываемое ПО компонентов сторонних разработчиков (библиотек, модулей, исходного кода), которые содержат уязвимости. В дальнейшем приобретенные уязвимости могут быть использованы с целью выполнения компьютерных атак на информационные системы пользователей, применяющих ПО. Угроза вызвана недостатками в системе безопасности процесса разработки: слабым контролем доступа и контролем целостности, применяемых к объектам среды разработки ПО, неполной проверкой компонентов на уязвимости, несовершенным управлением конфигурацией, недостаточным обучением сотрудников и нерегулярным поиском уязвимостей.

3. Стратегия управления уязвимостями при использовании заимствованных компонентов

Эффективное управление уязвимостями в заимствованных компонентах требует реализации комплексного подхода, объединяющего технические и организационные меры [2].

Одним из основных шагов при обеспечении безопасной разработки ПО является внедрение автоматизированного мониторинга с использованием инструментов анализа состава программного обеспечения (Software Composition Analysis). Данное решение позволяет осуществлять непрерывное сканирование зависимостей с подключением к общедоступным базам уязвимостей, таким как National Vulnerability Database (NVD), что обеспечивает своевременное обнаружение потенциальных угроз.

Существенную роль играет реализация регулярных процедур сканирования, включающих статический (SAST) и динамический (DAST) анализ кода. Статический анализ представляет собой процесс анализа исходного кода программы с целью выявления дефектов кода, в частно-

сти потенциально опасных конструкций. Статический анализ необходимо проводить как для исходного кода разрабатываемой программы, так и для заимствованных компонентов — за исключением доверенных, уже прошедших такую проверку у своих разработчиков. Динамический анализ кода представляет собой процесс выявления уязвимостей в режиме выполнения (run-time) программного обеспечения. Основные методы динамического анализа включают общую сборку путей выполнения, полносистемный анализ и фаззинг-тестирование. В отличие от статического анализа, DAST проводится для готового программного комплекса, включающего все интегрированные сторонние компоненты. При этом критически важным является проведение динамического анализа на регулярной основе — при каждом обновлении как собственного кода, так и используемых сторонних компонентов, поскольку изменения в их поведении могут порождать новые уязвимости.

Критически важным представляется ведение детального реестра компонентов (Software Bill of Materials). SBOM — перечень библиотек, конкретных файлов и зависимостей, имеющих отношение к разрабатываемому программному обеспечению. Он необходим, чтобы дать исчерпывающее представление о программных компонентах, их версиях, существующих лицензиях и т.д. Также в перечне можно найти способы проверки зависимостей и подтверждения подлинности источников, из которых заимствованы фрагменты программного кода. SBOM может применяться к любым компонентам программного обеспечения — как проприетарным, так и с открытым кодом.

Программное обеспечение защищено авторским правом согласно статье 1261 Гражданского кодекса Российской Федерации. Согласно статье, программой для ЭВМ является объективно выраженная совокупность данных и команд. Ее назначение — обеспечивать функционирование ЭВМ и других компьютерных устройств для получения определенного конечного результата. Важно отметить, что данное определение охватывает не только саму программу, но и подготовительные материалы, созданные в процессе ее разработки, а также аудиовизуальные отображения, которые программа порождает. Использование стороннего кода без соблюдения лицензии считается нарушением (ст. 1301 ГК РФ). Следовательно, дополнительно необходима проверка лицензионной совместимости компонентов для минимизации юридических рисков, так как даже open-source компоненты могут требовать выполнения условий лицензии. Разные открытые лицензии предъявляют свои требования к пользователям:

Пермиссивные (Apache, BSD). Код разрешается свободно использовать, изменять и распространять, в том числе в коммерческих целях. Однако требуется указывать авторов и сохранять информацию о лицензии.

Копилефтные слабые (MIT, LGPL). Эти лицензии также требуют указания авторов и условий. Главное отличие в том, что если вы вносите изменения в программное обеспечение, распространяемое по такой лицензии, то эти изменения также должны быть доступны под аналогичной открытой лицензией.

Копилефтные сильные (GPL). Использование кода под этой лицензией означает, что вы обязаны раскрыть исходный код всего вашего программного обеспечения, которое включает в себя GPL-компонент. Это делает их непригодными для большинства коммерческих продуктов, где хотят сохранить закрытость кода.

Помимо внедрения технических средств контроля, важным представляется реализация организационных мер, формирующих культуру безопасности в процессе разработки. Ключевыми элементами данной стратегии являются: регулярное обучение разработчиков современным практикам безопасного программирования и актуальным методам идентификации уязвимостей; формализация политики управления зависимостями, устанавливающей четкие процедуры выбора, верификации и обновления сторонних компонентов.

Заключение

Интеграция заимствованных компонентов в процесс разработки программного обеспечения представляет собой неотъемлемый элемент современной индустрии, однако сопряжена с существенными рисками информационной безопасности. В рамках исследования была предложена комплексная стратегия, сочетающая технологические и организационные меры защиты. Разработанная стратегия применима в корпоративной разработке, государственных информационных системах, IoT-устройствах и образовательных программах. Ключевыми элементами данной стратегии выступают автоматизированный мониторинг уязвимостей, систематическое применение методов статического и динамического анализа, ведение детализированного реестра компонентов, а также строгий контроль лицензионной совместимости. Особое значение имеет формирование организационной культуры безопасности, включающей регулярное обучение разработчиков и реализацию формализованных процедур управления зависимостями. Представленный подход позволяет достичь баланса между операционной эффективностью и необходимым уровнем защищенности программных продуктов.

Литература

1. ГОСТ Р 58412-2019. Защита информации. Разработка безопасного программного обеспечения. Угрозы безопасности информации при разработке программного обеспечения : национальный стандарт Российской Федерации : дата введения 2019-11-01 / Федеральное агентство по техническому регулированию. – Изд. официальное. – Москва : Стандартинформ, 2019. – 7 с.

2. О безопасности заимствованных компонентов Open Source // InformationSecurity : [сайт]. – 2024. – URL: https://www.itsec.ru/articles/o-bezopasnosti-zaimstvovannyh-komponentov-open-source (дата обращения: 23.11.2025).

3. Census III of Free and Open Source Software// THE LINUX FOUNDATION : [сайт]. – 2024. – URL: https://www.linuxfoundation.org/research/census-iii (дата обращения: 18.11.2025).

УДК 004.8

ИНТЕЛЛЕКТУАЛЬНАЯ НАДСТРОЙКА В АВТОМАТИЗАЦИИ ТЕСТИРОВАНИЯ

Д. М. Башлыков, И. Е. Воронина

Воронежский государственный университет

Аннотация. Рассматривается проблема ограничений классической UI-автоматизации и предлагается подход к построению интеллектуальной надстройки. Показано, что рост сложности программных систем делает традиционные автотесты хрупкими, трудоёмкими в сопровождении и слабо связанными с бизнес-логикой. Представлена возможная структура интеллектуальной надстройки, включающая модель предметной области, движок вывода и генератор сценариев, обеспечивающих осмысленное тестирование и выбор релевантных тестов после изменений. Представлены возможности автоматической генерации сценариев, анализа связей между сущностями и сокращения регрессионного набора. Выполнено обоснование того, что интеграция интеллектуальной надстройки повышает устойчивость автоматизации, снижает стоимость поддержки и улучшает качество тестового покрытия.

Ключевые слова: UI-тестирование, автоматизация тестирования, база знаний, интеллектуальная надстройка, бизнес-правила, регрессионное тестирование, Selenium, Playwright, генерация тестов, движок вывода, предметная область, семантические связи, CI/CD.

Введение

Современные программные продукты становятся всё более сложными: растёт количество экранов, пользовательских сценариев, зависимостей и интеграций. Изменения в одном модуле часто затрагивают другие части системы, а требования к качеству и скорости поставки функциональности постоянно повышаются. В таких условиях ручное тестирование уже не способно обеспечивать необходимый уровень регресса (основная идея регрессионного тестирования заключается в подтверждение того, что модификации не нарушили существующую функциональность [1]): оно занимает слишком много времени, зависит от человеческого фактора и плохо масштабируется. Поэтому автоматизация UI-тестирования стала не только актуальным процессом стандартом для большинства команд разработки. Такие инструменты, как Selenium, Selenide, Playwright и Cypress, позволяют воспроизводить реальные действия пользователя в браузере: открывать страницы, заполнять формы, выбирать элементы, проверять отображение данных. Это делает проверку стабильной, повторяемой и независимой от конкретного тестировщика.

Автоматизированные тесты легко становятся частью непрерывного процесса разработки: они интегрируются в CI/CD-конвейеры, запускаются автоматически при каждом изменении кода и позволяют выполнять регрессионные проверки без участия человека. Благодаря возможности параллельного запуска и высокой повторяемости, автотесты обеспечивают быструю обратную связь для команды разработки и значительно повышают устойчивость и предсказуемость продукта на длительной дистанции [2].

Таким образом, автоматизация UI-тестирования — это не просто удобный инструмент, а необходимый компонент современной разработки. Однако, несмотря на все преимущества, классический подход имеет ряд ограничений. Эти ограничения подталкивают к поиску новых способов усиления автоматизации, например, к использованию интеллектуальной надстройки для существующих фреймворков.

1. Проблемы классической UI-автоматизации

Рассмотрим ограничения, с которой сталкивается традиционная UI автоматизация. Они особенно заметны в процессе развития продукта и возрастания количества автотестов.

1. Хрупкость тестов при изменениях в интерфейсе.

UI-тесты напрямую завязаны на DOM-структуру, локаторы и визуальные элементы. Даже небольшие изменения — новый класс, перестроенная разметка, перенос кнопки — могут привести к сбоям. В результате тесты требуют постоянной поддержки, а часть регресса оказывается заблокирована до исправления.

2. Изменения UX вызывают каскадные падения.

Любое изменение пользовательского пути, структуры экранов или названий элементов приводит к падению десятков тестов. При этом сама бизнес-логика может остаться неизменной — меняется только внешний слой, а тесты реагируют как на критическую ошибку.

3. Дублирование тестовых сценариев.

При большом количестве тестов одни и те же шаги, условия и проверки повторяются в разных сценариях. Это приводит к разрастанию кода, усложняет поддержку и увеличивает время на анализ поломок.

4. Нет ясности, что именно покрыто тестами.

Классические UI-тесты проверяют конкретные действия: «нажать кнопку», «ввести текст». Но они не отражают связь с реальной бизнес-логикой. Нельзя легко ответить на вопрос:

- покрыто ли конкретное бизнес-правило;
- проверяются ли все обязательные ограничения формы;
- какие сценарии будут затронуты конкретным изменением в коде.

Анализ покрытия сводится к набору технических метрик — и мало что говорит о полноте проверки предметной области.

5. Сложно определить какие тесты запускать после изменения продукта.

Если разработчик изменил один модуль, приходится запускать весь набор UI-тестов, так как нет понимания, какие сценарии связаны с этой функциональностью [3]. Это делает регресс долгим и тяжёлым.

6. Отсутствие единых правил и осмысленной модели приложения.

В UI-автотестах нет встроенного представления о том:

- какие сущности есть в системе;
- какие у них свойства;
- какие ограничения действуют;
- какие сценарии допустимы или запрещены.

Автотест — всего лишь код, который исполняет действия. Он не «понимает» систему, а лишь воспроизводит заранее «зашитые» шаги.

Все вышеперечисленные проблемы имеют общий корень: классическая UI-автоматизация не обладает знанием о приложении. Она проверяет поведение интерфейса, но не опирается на формализованную модель бизнес-логики. Отсюда и возникает необходимость в более интеллектуальном подходе, который позволит сделать тесты осмысленными, адаптивными и менее зависимыми от визуальных изменений.

2. Интеллектуальная надстройка

Чтобы преодолеть ограничения классической UI-автоматизации, поверх существующих инструментов можно создать интеллектуальную надстройку, которая не просто исполняет тестовые шаги, а понимает, что именно она проверяет. Ключевая идея заключается в том, что

автотесты должны опираться не только на код и локаторы, но и на формализованную модель знаний о продукте. Для этого необходимо разработать собственную базу знаний, что позволит:

- интерпретировать бизнес-правила;
- анализировать связи между сущностями;
- выбирать релевантные тесты при изменениях;
- автоматически формировать новые сценарии.

Фактически, тестовая инфраструктура перестаёт быть набором скриптов и превращается в систему рассуждений, где UI-тесты становятся исполнителями, а логика тестирования — результатом анализа знаний.

В общем виде архитектуру можно представить следующим образом:

- UI-автотесты — «руки»: кликают, вводят, проверяют.
- Selenium/Playwright — «мышцы»: выполняют действия.
- База знаний + логический движок — «мозг»: решает, *что* тестировать.

Функции интеллектуальной надстройки:

- анализирует модель приложения;
- выбирает релевантные сценарии;
- генерирует проверки;
- сбрасывает действия в привычный слой автотестов.

Таким образом, надстройка позволит тестам быть устойчивыми к визуальным изменениям, обеспечить осмысленный анализ покрытия и превратить автоматизацию в гораздо более адаптивный и управляемый процесс.

3. Возможности интеллектуальной надстройки

Рассмотрим основные направления работы интеллектуальной надстройки

3.1. Генерация тестовых сценариев по правилам

База знаний хранит не только сущности и их свойства, но и правила и ограничения.

На основе этих правил система может автоматически создавать тестовые случаи без ручного написания кода.

Пример правила:

Если статус заявки = Черновик — разрешено редактирование и удаление, публикация запрещена

Сгенерированные тесты:

1. Пользователь открывает заявку в статусе Черновик → проверка доступности кнопки «Редактировать».
2. Попытка удалить заявку → ожидаемое успешное удаление.
3. Попытка опубликовать → проверка ошибки или недоступности кнопки.

Таким образом, тесты создаются на основе логики продукта, а не на наборе случайных комбинаций действий.

3.2. Выбор релевантного тестового набора

После изменения функциональности приложения надстройка может определить, какие тесты необходимо выполнить.

Надстройке известны зависимости между сущностями и экранами, например, при изменении модуля X, кроме того автоматически формируется набор тестов, который покрывает

только пострадавшие элементы, экономя время на регресс. Таким образом, предотвращаются бессмысленные запуски всех тестов и для разработчиков сокращается цикл обратной связи.

3.3. Семантические связи

Компоненты надстройки:

• сущности (например, Пользователь, Заявка, Рабочее пространство);

• их свойства и ограничения (обязательные поля, допустимые статусы);

• разрешённые действия и переходы между состояниями.

Пример.

Правило *Поле Email обязательно* позволяет автоматически генерировать негативные кейсы:

• оставить поле пустым → проверить ошибку валидации;

• ввести некорректный формат → проверить корректную обработку ошибки.

Тем самым уменьшается доля ручного планирования сценариев и повышается полнота тестового покрытия.

3.4. Представление знаний

База знаний может использовать:

• Продукционные правила: простые проверки условий → действие/результат;

• Фреймы: описания шаблонов сущностей, их свойств и связей;

• Семантические сети: графы зависимостей между сущностями и функциями.

Пример.

• Фрейм «Заявка» содержит поля: статус, тип, дата создания.

• Правило *Если статус = Ожидание → редактирование запрещено* автоматически порождает тест, проверяющий доступность кнопок.

• Семантические зависимости позволяют определить, какие другие сущности могут быть затронуты изменением определенного компонента системы.

4. Разделение ролей

Взаимодействие проекта автоматизации UI с интеллектуальной надстройкой строится по принципу разделения ролей: UI-тесты выполняют действия, а интеллектуальная система решает, что и как проверять.

• База знаний: хранит модель предметной области, бизнес-правила и связи между сущностями;

• Движок вывода: анализирует правила и зависимости, определяет, что и какие тесты нужно выполнить;

• Генератор тестов: превращает правила из базы знаний в конкретные пошаговые сценарии для UI-тестов;

• UI Test Runner (Selenium/Playwright): исполняет тестовые сценарии: клики, ввод данных, проверку отображения элементов;

• CI/CD: автоматически запускает тесты при изменениях в коде, обеспечивает параллельное выполнение и интеграцию с конвейером разработки.

В этой архитектуре UI-слой не меняется: локаторы, страницы, методы Page Object остаются прежними. Меняется только логика тестирования: вместо ручного написания сценариев, тесты получают последовательность действий от «мозга» системы — базы знаний и движка вывода.

Заключение

Использование интеллектуальной надстройки для классической UI-автоматизации приносит ряд ощутимых преимуществ, которые повышают эффективность тестирования и качество продукта:

• Снижение расходов на поддержку.

Тесты формируются автоматически на основе правил, поэтому уменьшается необходимость ручного переписывания сценариев при изменениях интерфейса или бизнес-логики.

• Минимизация хрупкости тестов.

Поскольку тесты опираются на модель предметной области, а не на конкретные локаторы, изменения в DOM или UX не приводят к массовым падениям.

• Интеллектуальный выбор тестов после изменений.

Система анализирует, какие правила и сущности затронуты модификациями, и выбирает только релевантные сценарии для выполнения, экономя время и ресурсы.

• Быстрая генерация большого числа сценариев.

На основе правил и связей в базе знаний можно создавать десятки или сотни тестов автоматически, покрывая разнообразные комбинации данных и условий.

• Осмысленная проверка покрытия.

Покрытие измеряется не на уровне строк кода, а на уровне бизнес-правил и сущностей. Это позволяет видеть, какие правила полностью проверены, а какие требуют дополнительных сценариев.

• Ускоренный регресс.

Тесты выполняются быстрее и более целенаправленно, что сокращает цикл обратной связи для команды разработки.

• Тестовая система становится «знающей», а не «исполняющей».

Автоматизация превращается из набора скриптов в осмысленную систему, которая понимает структуру приложения, связи между сущностями и правила проверки, что делает тестирование более устойчивым и управляемым.

Литература

1. *Канер С.* Тестирование программного обеспечения. Практическое руководство / С. Канер, Д. Бах, Б. Петтикорд. – Санкт-Петербург : Питер, 2004. – 320 с.

2. *Дастин Э.* Автоматизированное тестирование программного обеспечения / Э. Дастин, Д. Рашка, Дж. Пол. – Москва : Лори, 2003. – 544 с.

3. *Майерс Г.* Искусство тестирования программ / Г. Майерс, К. Сандлер, Т. Баджетт. – 3-е изд. – Москва : Диалектика, 2019. – 256 с.

УДК 004.4

ДИНАМИЧЕСКОЕ МАСШТАБИРОВАНИЕ ВИРТУАЛИЗИРОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ РЕСУРСОВ ИНФОРМАЦИОННО-ТЕХНОЛОГИЧЕСКИХ СЕРВИСОВ НА БАЗЕ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

Ю. А. Воронцов, А. В. Калач

МИРЭА – Российский технологический университет

Аннотация. В работе рассмотрен подход к динамическому масштабированию ИТ-сервисов с учетом ограничений на обрабатываемые задачи и вычислительную инфраструктуру. Представлен подход к решению задачи масштабирования, состоящий из трёх этапов: прогноза периода максимального потребления ресурса, основанного на терминах «дефицита» и «профицита» вычислительного ресурса, вычисления влияния периода максимального потребления ресурса на качество работы программного сервиса и формирования набора типовых решений для уменьшения времени принятия решения об изменении конфигурации вычислительной инфраструктуры.
Ключевые слова: виртуальная вычислительная инфраструктура, микросервисная архитектура, ИТ-сервисы, динамическое масштабирование, контейнеризация, оркестрация контейнеров, дефицит ресурсов, прогноз загруженности, SLO, качество обслуживания, конфигурация инфраструктуры.

Введение

Активное развитие программных приложений, построенных по принципам микросервисной архитектуры, и реализующих информационно-технологические (ИТ) сервисы, привели к повсеместному внедрению технологий виртуализации вычислительных ресурсов (контейнеризации и оркестрации) для их развертывания. В свою очередь это привело к росту потребности в формировании гибких подходов к управлению вычислительными ресурсами. Обуславливается эта потребность особенностями нагрузки со стороны пользователей на информационно-технологические сервисы, которая характеризуется следующими пунктами: выраженные сезонности (суточные/недельные), нерегулярность, а также длительные периоды повышенного потребления вычислительных ресурсов, чья продолжительность и амплитуда заранее неизвестны. Существующие исследования предлагают различные подходы к решению задачи динамического выделения ресурсов, вместо применения статических квот, ведущих либо к значительному перерасходу ресурсов, либо к нарушению эксплуатационных соглашений (SLO) по времени отклика и пропускной способности.

Базовым способом масштабирования является реактивный подход [1, 2], предполагающий управление экземплярами приложений на основе правил. Его развитием стали способы масштабирования с очереде-ориентированными и SLO-ориентированными условиями, сместившими фокус в сторону показателей непосредственно связанные с качеством обслуживания [1, 3]. Дальнейшие развитие способов масштабирования подразумевает применение более сложных теорий, например, теории управления, при помощи которой ИТ-сервис трактуют как объект регулирования с целью стабилизации целевой метрики, например, задержки, в окрестности заданного значения [4, 5]. Помимо исследования текущего состояния сервиса также рассматриваются подходы на основе краткосрочного прогноза нагрузки и заблаговременного изменяют мощность [1, 6]. Для устранения недостатков прогностических методов также предлагается использовать их гибриды с реактивным подходом [5]. Отдельно стоит выделить подходы на основе машинного обучения и обучения с подкреплением [5, 7].

Все из перечисленных подходов показывают свою эффективность и актуальность для инфраструктур разного масштаба. При этом существуют пробелы в предлагаемых решения, выраженные в следующих пунктах. Исходя из этого необходимо разработать подход динамического масштабирования виртуализированных ресурсов, позволяющий снизить стоимость использования ресурсов при соблюдении (улучшении) SLO за счёт прогноза длительности периодов пикового потребления ресурсов и выбора конфигурации с учетом возможных рисков и ошибок. Разрабатываемый подход должен содержать следующие этапы:

1. Прогноз продолжительности периода максимального использования выделенного ресурса.

2. Выбор оптимального объёма ресурсов и числа обработчиков.

3. Формирование набора типовых решений, исходя из эффективности выбранных конфигураций.

Будем учитывать также следующие ограничения. Сервисы должны обслуживать однотипные задачи или задачи должны быть кластеризированы по ресурсоемкости; сервисы должны предоставлять телеметрию очереди и/или времени обработки задач; временные задержки масштабирования и запуска новых экземпляров приложений могут быть замерены; возможны предсказуемые сезонности и/или нестационарные сегменты нагрузки. Дополнительно к ограничениям необходимо отнести наличие нестабильной производительности из-за повышенной нагрузки от развернутых на той же инфраструктуре сервисов и ограничения вертикального масштабирования ресурсов на физических узлах.

1. Прогноз продолжительности периода максимального использования выделенного ресурса

В качестве основного показателя при прогнозе предлагается использовать *объем вычислительного ресурса, требуемого для выполнения поступающих задач* — $\hat{r}(t)$. Вычислительный ресурс в каждый момент времени также характеризуется *максимальным доступным объемом* — $r_{\max}(t)$. В каждый момент времени t вычислительный ресурс может быть характеризован разницей между $r_{\max}(t)$ и $\hat{r}(t)$, описывающей пребывание в одном из двух состояний: «*дефицит ресурса*» — $D(t)$ и «*профицит ресурса*» — $P(t)$. Благодаря этим состояниям можно говорить о том, испытывает ли в момент времени t рассматриваемый сервис необходимость в изменении для каждого экземпляра приложения *числа обработчиков* c и $r_{\max}$, чтобы каждому из обработчиков выделялся требуемый объем ресурсов, или же *количества экземпляров приложения* d. Следовательно, $D(t)$ и $P(t)$ будут являть одними из основных параметров для управления. Определим:

$$\hat{r}(t) = \lambda(t) \cdot \hat{r}_{task}(t), \tag{1}$$

где $\lambda(t)$ — *интенсивность поступления задач*, $\hat{r}_{task}(t)$ — *объем ресурса, необходимый на выполнение одной задачи*. Объем ресурса для одной задачи может быть вычислен исходя из данных мониторинга как $\hat{r}_{task}(t) = r_{fact}(t) \div \mu_{eff}(r_{\max}(t))$, где $r_{fact}(t)$ — фактическое потребление ресурса, полученное от системы мониторинга, $r_{\max}(t)$ — выделенный объем ресурса, $\mu_{eff}(r_{\max}(t))$ — *эффективная производительность*, определяет количество выполняемых обработчиком задач с учетом доли полноценно функционирующих обработчиков на момент времени t, т. к. часть из них может находиться в состоянии запуска или, наоборот, в состоянии завершения.

Прогноз продолжительности периода максимального использования выделенного вычислительного ресурса подразумевает наличие дефицита ресурса в течении определенного интервала времени. Чем длительнее интервал дефицита, тем длительнее выделенный ресурс будет находиться в состоянии максимального потребления в будущем. Определим суммарный дефицит ресурса на момент времени t как интегральный показатель:

$$D_{sum}(t) = \int_{t^*}^{T} \hat{r}(t)dt, \tag{2}$$

где t^* и T — моменты времени начала и конца периода дефицита ресурса, при этом T может рассматриваться как текущий момент времени. На рис. 1 графически представлено вычисление суммарного дефицита ресурса.

Полученные данные могут быть использованы для вычисления *периода максимального использования выделенного вычислительного ресурса* в будущем, поскольку период дефицита описывает перечень задач, которые не были обработаны в момент поступления, и были отложены на выполнения в один из следующих моментов времени. Можно говорить, что *объем дефицита ресурса* $D_{sum}(t)$ напрямую связан с длительностью периода в будущем, в котором $\hat{r}(t)$ будет принимать значения максимально близкие к $r_{\max}(t)$, т. е. ресурс будет находится в состоянии максимального потреблению.

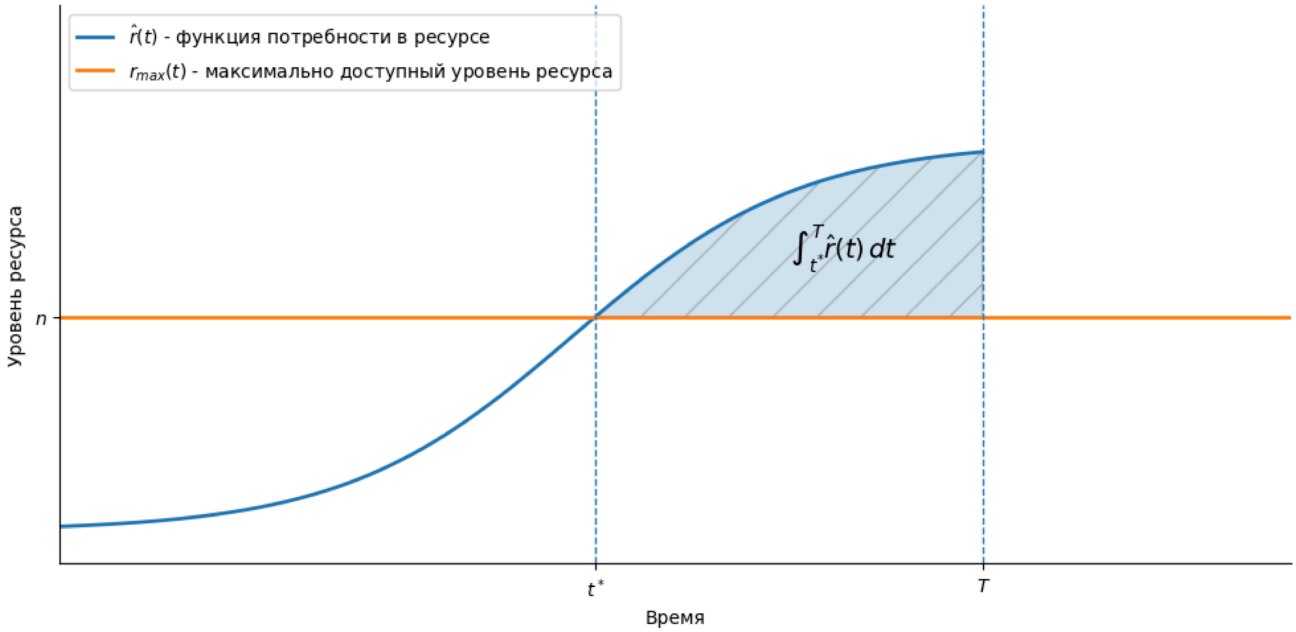

Рис. 1. Графическое представления поиска дефицита ресурса

2. Вычисление влияния периода максимального потребления ресурса на качество работы программного сервиса

В процессе выполнения поставленных задач программное обеспечение может потреблять различные ресурсы. Конкретный паттерн потребления зависит от типа решаемых задач и наиболее значимого для этой задачи ресурса, однако, всегда необходимо понимать, какой конкретно ресурс или набор ресурсов является ограничением по производительности. Необходимо определять, какой ресурс или набор ресурсов находится в состоянии дефицита и негативно сказывается на эффективности обработки поступающих задач. Управление данным ресурсом/набором ресурсов будет являться наиболее приоритетной задачей, а продолжительность периода его максимальной загрузки в будущем непосредственно влияет на продолжительность выполнения задач и может быть использована для проверки соблюдения критериев SLO для новых поступающих задач, а также для вычисления размеров штрафов, при нарушении этих критериев.

Определение продолжительности периода максимальной загрузки дает нам достаточное количество данных, чтобы предполагать, через какой период времени будет исполнена поступившая задач. Нахождение ресурса в периоде дефицита подразумевает его нехватку для обра-

ботки всех поступивших в момент времени t задач за время $\tau \le \Delta t$, где Δt — приращение $\hat{r}(t)$. То есть поступившие задачи не будут полностью исполнены за время между получением данных о загруженности ресурсов, при условии, что ресурс может быть полностью израсходован при текущей конфигурации обработчиков за этот период. Тогда время выполнения поступившей задачи будет равно $ET(t) = W(t) + S(t)$, где $W(t)$ — время ожидания в очереди, а $S(t)$ — время обработки задачи.

Поскольку $D_{sum}(t)$ является интегральным показателем и имеет накопительный эффект, можно говорить, что каждая новая поступающая в момент времени t задача будет исполнена через $W(t) = D_{sum}(t) \div \hat{r}(t)$. В то время как $S(t) = 1 \div \mu_{eff}(r_{\max}(t))$, где за 1 принимается объем выполняемой работы при решении поступающей задачи.

При получении значения $ET(t)$ можно говорить о том, удовлетворяет ли текущее время обработки критериям, прописанным в SLO, или же нет и необходимо ли при этом предпринимать какие-либо действия. Таким образом $ET(t)$ может служить основным критерием для изменения количества экземпляров приложения, обработчиков и объема выделенных для этих обработчиков ресурсов.

3. Формирование набора типовых решений, исходя из эффективности применяемых конфигураций

Нагрузка на ИТ-сервис в большинстве случает имеет повторяющиеся сменяющие друг друга паттерны. У программного обеспечения можно вывести сформированную зависимость между производительностью и количеством его экземпляров, обработчиков в каждом экземпляре с учетом выделяемых лимитов по ресурсам. На основании этого можно вывести зависимость между паттерном нагрузки со стороны пользователей и наиболее эффективной конфигурацией программного обеспечения, удовлетворяющей требованиям SLO и уровню затрат на обслуживаемую инфраструктуру.

Первоначальный набор решений может формироваться исходя из нагрузочного тестирования программного сервиса на приближенной к реальной конфигурации вычислительной инфраструктуры. При этом тестирование должно производиться с учетом гетерогенности вычислительной инфраструктуры и всех возможных конфигураций аппаратного обеспечения, на которых будут разворачиваться обработчики. В процессе работы системы эти конфигурации могут дополнительно корректироваться исходя из полученных значений эффективности обработки в реальном времени.

Заключение

Предлагаемый подход позволит повысить эффективность масштабирования программных сервисов на базе микросервисной архитектуры, основываясь на связи показателей потребления вычислительных ресурсов с показателями качества работы сервиса. Дальнейшие исследования будут направлены на развитие заложенного в данной работе подхода.

Литература

1. *Zou D., Lu W., Zhu Z., Lu X., Zhou J., Wang X., Liu K., Wang H., Wang K., Sun R.* OptScaler: A Collaborative Framework for Robust Autoscaling in the Cloud // Proceedings of the VLDB Endowment. – 2024. – V. 17, No 12. – P. 4090–4103. – DOI: 10.14778/3685800.3685829.

2. Kubernetes Documentation. Horizontal Pod Autoscaling // Kubernetes : [сайт]. – URL: https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/ (дата обращения: 02.11.2025).

3. KEDA Documentation. Event-driven autoscaling for Kubernetes // KEDA : [сайт]. – URL: https://keda.sh/docs/2.18/scalers/ (дата обращения: 03.11.2025).

4. Nguyen T.-T., Yeom Y.-J., Kim T., Park D.-H., Kim S. Horizontal Pod Autoscaling in Kubernetes for Elastic Container Orchestration // Sensors. – 2020. – V. 20, No 16. – St. 4621. – DOI: 10.3390/s20164621.

5. Autoscaling Strategies: rule-based vs PID // Mantis – Netflix Open Source : [сайт]. – URL: https://netflix.github.io/mantis/operate/autoscalingstrategies/ (дата обращения: 03.11.2025).

6. *Alharthi S., Alshamsi A., Alseiari A., Alwarafy A.* Auto-Scaling Techniques in Cloud Computing: Issues and Research Directions // Sensors. – 2024. – V. 24, No 17. – St. 5551. – DOI: 10.3390/s24175551.

7. *Xu M., Song C., Ilager S., Gill S. S., Zhao J., Ye K., Xu C.* CoScal: Multifaceted Scaling of Microservices with Reinforcement Learning // IEEE Transactions on Network and Service Management. – 2022. – V. 19, No 4. – P. 3995–4009. – DOI: 10.1109/TNSM.2022.3210211.

УДК 510.52, 004.423.25

О ЗАДАЧЕ 1-ВЫВОДИМОСТИ СЛОВ В КОНТЕКСТНО-СВОБОДНОЙ ГРАММАТИКЕ

С. М. Дудаков

Тверской государственный университет
НИУ Высшая школа экономики

Аннотация. Ранее нами рассматривалась задача тотальной выводимости в грамматиках разного вида, то есть когда в выводе требуется применить все правила хотя бы по одному разу. Эта задача первоначально возникла для аксиоматизации некоторых форм исчисления Ламбека, но она же может интерпретироваться и как, например, задача о поведении событийно-ориентированной системы. В настоящей работе мы рассматриваем противоположную ситуацию, когда есть верхние ограничения на количества применений правил. Наш основной результат: даже в простейшем случае, когда каждое правило можно применить не более чем единожды, а выводимое слово пусто, такая задача будет *NP*-полной, если не делать ограничений на количество нетерминалов в грамматике. Если последнее ограничение есть, то есть алгоритм, решающий задачу за полиномиальное время.
Ключевые слова: контекстно-свободная грамматика, нетерминал, выводимость с ограничениями, пустое слово, разрешимость за полиномиальное время, класс *NP*, *NP*-полная задача, задача о «бартерных сделках», линейная логика, мультипликативный фрагмент.

Введение

Мы изучаем контекстно-свободные грамматики (КС-грамматики) и выводы в них. Классическое определение вывода — последовательная замена в слове нетерминала на цепочку из терминалов и нетерминалов по одному из правил вывода. При этом не предполагается ограничений на количество применений правил. Алгоритмы, проверяющие выводимость, хорошо известны, эта задача может быть эффективно решена за полиномиальное время.

Однако, если интерпретировать вывод в КС-грамматике, как последовательность событий в системе [3], то логичными будут вопросы о том, какое именно количество применений каждого из правил в выводе было использовано. В наших ранних работах [2, 3] мы показали, что если потребовать, чтобы каждое из правил было применено не менее одного раза, то даже для пустого слова задача о его выводимости становится NP-полной. Первоначально такая задача была поставлена в [2] для задачи аксиоматизации некоторого варианта исчисления Ламбека, но, как мы отметили, она может быть интерпретирована и как задача о поведении системы, управляемой событиями. Для этого достаточно рассмотреть нетерминалы, как события, терминалы, как действия, а правила — как возможные варианты реации системы на различные события (триггеры).

Естественным будет задать противоположное ограничение: какой будет ситуация, если мы установим верхние ограничения для применений правил? Такая постановка задачи тоже может быть интерпретирована на практике. Если, например, каждое применение правил связано с использованием тех или иных ресурсов и количество этих ресурсов ограничено, то вопрос о выводимости с ограничениями превращается в разновидность известной задачи о «бартерных сделках» [1]. Заметим, что последняя задача тоже первоначально была сформулирована в терминах исчисления для линейной логики Жирара, точнее, её мультипликативного фрагмента.

В настоящей работе мы рассматриваем задачу такого вида: можно ли вывести пустое слово в КС-грамматике, если каждое правило может быть применено не более одного раза. Заметим, что эта задача отличается от той, которая изучена в работе [1]. В линейной логике каждая импликация при выводе может быть использована в точности один раз, в то время как наша задача допускает, что какие-то правила могут не быть использованы вообще.

Основным нашим результатом является следующий: если количество нетерминалов грамматики ограничено, то изучаемая задача имеет решение в полиномиальном времени, если же такого ограничения нет, то она является NP-полной. Посленее означает, что в предположении $PTIME \neq NP$ задача не может быть решена с использованием полиномиального времени.

1. Основные определения

Контекстно-свободной грамматикой (КС-грамматикой) называется набор из четырёх элементов $G = (\Sigma, N, P, S)$. Здесь Σ — это множество терминальных символов, из которых состоят слова языка, N — множество нетерминальных символов, P — множество правил вывода, а S — начальный нетерминал. Правилом вывода называется слово вида $A \to x_1 \dots x_n$, в котором $A \in N$ и $x_1, \dots, x_n \in \Sigma \cup N$. **Выводом** в грамматике G называется конечная последовательность слов $(w_0, \dots, w_k)$, первое из которых равно S, а каждое из следующих слов w_{i+1} получается из предыдущего применением одного из правил из P. Последнее означает, что если правило имеет вид $A \to x_1 \dots x_n$ то одно из вхождений символа A в слове w_i заменено в слове w_{i+1} словом $x_1 \dots x_n$ Вывод $(w_0, \dots, w_k)$ в грамматике G — это **вывод слова** w_k.

Мы называем вывод $(w_0, \dots, w_k)$ в грамматике G **1-ограниченным**, если каждое из правил вывода применяется не более одного раза. Если существует 1-ограниченный вывод слова w в грамматике G, то мы будет называть слово w **1-выводимым** в G. Задача **1-выводимости**: по грамматике G и слову w определить, является ли w 1-выводимым в грамматике G.

Сразу отметим тривиальный факт. Наша задача имеет два входных параметра — грамматику G и слово w. В ряде случаев имеет смысл фиксировать один из параметров, чтобы рассмотреть более частный вариант задачи, который может быть проще, чем в общем случае. Если мы зафиксируем грамматику G, то наша задача действительно становится тривиальной: если заранее знать множество правил P, то мы также заранее знаем конечное множество 1-ограниченных выводов, и, следовательно, конечное же множество 1-выводимых слов.

Поэтому если мы хотим рассмотреть частные случаи нашей задачи, то фиксировать имеет смысл второй аргумент — входное слово. Мы рассматриваем самый простой вариант, когда зафиксировано пустое слово, обозначаемое ϵ.

2. Главный результат

Теорема 1. *Если количество нетерминалов грамматики G заранее ограничено константой K, то задача 1-выводимости пустого слова может быть решена за полиномиальное время.*

Доказательство. Прежде всего отметим, что любой вывод можно переупорядочить так, что все правила вида $A \to \epsilon$ будут использованы в самом конце, то есть до этого момента вывод будет несокращающим.

Поскольку каждое правило вида $A \to \epsilon$ мы можем применить не более одного раза, то если в ходе вывода будет получено слово длинее K, то далее пустое слово мы получить уже не сможем.

Далее, из K нетерминалов можно построить не более K^2 цепных правил, то есть правил вида $A \to B$, где A и B нетерминалы. А если общее количество правил равно N, то в выводе, длина которого тоже не превосходит N, эти цепные правила могут быть размещены не более чем N^{K^2} способами. Поэтому мы можем последовательно рассмотреть каждый из этих способов и для каждой последовательности цепных правил $A \to \dots \to B$ сделать замену в правиле $C \to \dots A \dots$, по которому получено первое A, на B. В результате мы получим задачи о 1-выводимости пустого слова для N^{K^2} КС-грамматик без цепных правил.

Чтобы решить каждую из последних задач, мы рассматриваем выводы длины не более K, так как теперь каждое применение правила, кроме последних правил вида $A \to \epsilon$, ведёт к уве-

личению длины слова, а как мы уже отметили ранее, эта длина не может превосходить константу K. Следовательно, общее количество таких выводов не превосходит N^K, поэтому их тоже можно перебрать за полиномиальное время и проверить, можно ли с помощью хотя бы одного из них получить пустое слово.

Заметим, что эта теорема легко обобщается на случай любого конкретного слова длины M. Отличие будет только в том, что теперь максимальное слово, которое может возникнуть на промежуточном этапе вывода будет $K+M$, что тоже является константой.

Теорема 2. *В общем случае задача 1-выводимости пустого слова в КС-грамматике G является NP-полной.*

Доказательство. Принадлежность задачи классу NP очевидна: угадываем вывод, длина которого не превышает количества правил, проверяем, что каждое из слов получено из предыдущего по одному из правил вывода, что правила не повторяются, что первым словом является S, а последним w. Если все условия выполнены, то ответом будет «да», в противном случае — «нет».

Чтобы доказать NP-трудность, мы сведём задачу о выполнимости 3-КНФ к задаче 1-выводимости пустого слова.

Пусть нам дана 3-КНФ $\Phi = d_1 \wedge \cdots \wedge d_m$, то есть конъюнктивная нормальная форма, в которой каждая элементарная дизъюнкция d_i состоит из трёх литералов: $d_i = l_i^1 \vee l_i^2 \vee l_i^3$. Каждый из литералов l_i^j является пропозициональной переменной z_l или её отрицанием $\neg z_l$. Полагаем, что всё множество пропозициональных переменных — это $\{z_1,\ldots,z_n\}$. Для каждой пропозициональной переменной z_l пусть C_l^+ — это количество её вхождений в Φ без отрицания, а C_l^- — с отрицанием. Без ограничения общности можно считать, что каждая элементарная дизъюнкция d_i содержит каждую переменную z_l не более чем один раз.

Составим множество нетерминалов N следующим образом. Включим в него начальный нетерминал S. Для каждой пропозициональной переменной z_l добавим нетерминалы Z_l, Z'_l, $Z^*_l, Y_{l,1},\ldots,Y_{l,C_l^+}, Y'_{l,1},\ldots,Y'_{l,C_l^-}$. Для каждой элементарной дизъюнкции d_i добавим нетерминал D_i.

Рассмотрим грамматику G, которая содержит следующие правила вывода:

1. $S \to Z^*_1 \ldots Z^*_n D_1 \ldots D_m$;

2. $D_i \to L_i^j$ для $i = 1,\ldots,m$, $j = 1,2,3$. С помощью L_i^j обозначаем нетерминал Z_l, если $l_i^j = z_l$, и Z'_l, если $l_i^j = \neg z_l$;

3. $Z_l^* \to Y_{l,1} \ldots Y_{l,C_l^+}$ для $l = 1,\ldots,n$;

4. $Z_l^* \to Y'_{l,1} \ldots Y'_{l,C_l^-}$ для $l = 1,\ldots,n$;

5. $Z_l \to Y_{l,p}$ для $l = 1,\ldots,n$, $p = 1,\ldots,C_l^+$;

6. $Z'_l \to Y'_{l,p}$ для $l = 1,\ldots,n$, $p = 1,\ldots,C_l^-$;

7. $Y_{l,p} \to \epsilon$ для $l = 1,\ldots,n$, $p = 1,\ldots,C_l^+$;

8. $Y'_{l,p} \to \epsilon$ для $l = 1,\ldots,n$, $p = 1,\ldots,C_l^-$.

Покажем, что пустое слово можно 1-вывести в грамматике G тогда и только тогда, когда исходная КНФ Φ выполнима.

Пусть в грамматике G пустое слово имеет 1-ограниченный вывод. Он начинается с применения правила 1. Далее, для каждого нетерминала Z_l^* должно быть применено одно из правил 3 или 4. Для получившихся на предыдущем шаге Y или Y' должны быть применены правила 7 или 8 соответственно. Заметим, что в силу 1-ограниченности вывода после этого шага правила 7 или 8 соответственно применятся не могут. Из этого получается, что не могут быть применены и правила 5 или 6 соответственно. Таким образом, мы окончательно получаем, что если для Z_l^* было применено правило 3, то далее для D_i мы не можем применять правила $D_i \to Z_l$, поскольку далее невозможно применить правила 5. Точно так же, если для Z_l^* было

применено правило 4, то далее для D_i мы не можем применять правила $D_i \to Z'_l$, поскольку далее невозможно применить правила 6.

Построим интерпретацию I так, что $I(z_l)$ равно 1, если для Z_l^* применено правило 4, и $I(z_l)$ равно 0, если для Z_l^* применено правило 3. Тогда для каждого D_i выбрано правило 2 согласно ограничениям из предыдущего абзаца. Таким образом, если для D_i применено правило вида $D_i \to Z_l$, то $I(z_l) = 1$, где литерал z_l входит в элементарную дизъюнкцию d_i, поэтому $I(d_i) = 1$. И наоборот, если для D_i применено правило вида $D_i \to Z'_l$, то $I(z_l) = 0$, где литерал $\neg z_l$ входит в элементарную дизъюнкцию d_i, поэтому снова $I(d_i) = 1$.

Окончательно получаем, что $I(d_i) = 1$ для всех $i = 1, \ldots, m$, поэтому $I(\Phi) = 1$, то есть КНФ Φ выполнима.

Пусть теперь, наоборот, КНФ Φ выполнима, то есть $I(\Phi) = 1$ для некоторой интерпретации I. Это означает, что в каждой элементарной дизъюнкции d_i есть литерал $l_i^{j_0}$, для которого $I(l_i^{j_0}) = 1$. Построим тогда вывод в грамматике G следующим образом. Применим к S первое правило, для каждого D_i применим правило $D_i \to L_i^{j_0}$, а для каждого Z_l^* применим правило 3, если $I(z_l) = 0$, или правило 4 в противном случае. В результате у нас окажется слово, в котором каждый $Y_{i,p}$ и каждый $Y'_{i,p}$ встречается не более чем по одному разу, поэтому, используя правила 7 и 8 не более чем по одному разу, мы сможем превратить всё слово в пустое.

Как и теорема 1 теорема 2 может быть обобщена на случай произвольного слова w, для этого достаточно дописать его в конец правила 1.

Заключение

Мы показали возможность полиномиального решения проблемы 1-выводимости для конкретного слова, если ограничить количество нетерминалов. Возникает естественный вопрос о сложности задачи выводимости не для конкретного слова, а когда оно тоже может меняться. Очевидно, что алгоритм, предложенный в теореме 1, в этом случае уже не будет полиномиальным.

Другой вопрос: мы видели, что задача о выводимости пустого слова превращается в *NP*-полную, как только мы ограничиваем количество применений каждого правила единицей. Доказательство легко модифицировать, заменив единицу на любую константу Q, достаточно соответственно умножить количество Y и Y' в правых частях правил. Возникает вопрос: существует ли какое-то более слабое ограничение, при котором задача всё ещё будет *NP*-полной?

Литература

1. Concurrency problem for horn fragment of girard's linear logic / D. A. Archangelsky, M. I. Dekhtyar, E. Kruglov [et al.] // Lecture Notes in Computer Science. – 1994. – Vol. 813 LNCS. – P. 18–22. – DOI 10.1007/3-540-58140-5_3. – EDN XMUBQJ.

2. Сложность исчислений Ламбека с модальностями и тотальной выводимости в грамматиках / С. М. Дудаков, Б. Н. Карлов, С. Л. Кузнецов, Е. М. Фофанова // Алгебра и логика. – 2021. – Т. 60, № 5. – С. 471–496. – DOI 10.33048/alglog.2021.60.502. – EDN SOVAEC.

3. *Дудаков С. М.* О сложности проблемы тотальной выводимости в неукорачивающих и контекстно-свободных грамматиках / С. М. Дудаков, Б. Н. Карлов // Доклады Российской академии наук. Математика, информатика, процессы управления. – 2025. – Т. 524, № 1. – С. 11–18. – DOI 10.7868/S3034504925040024. – EDN QJGJGT.

УДК 004.272, 004.312

РАЗРАБОТКА АРХИТЕКТУРЫ АППАРАТНОЙ РЕАЛИЗАЦИИ 2D ГРАФИЧЕСКОГО УСКОРИТЕЛЯ

И. А. Дудкин, Д. М. Старухин, В. В. Черкасов

Воронежский государственный университет

Аннотация. В начале освещается вопрос причины и актуальности разработки. Ставится задача разработки архитектуры аппаратной реализации 2D графического ускорителя. Приводятся различия требований к графическим ускорителям в зависимости от устройств их использующих. Обсуждается подход к разработке архитектуры. Описываются основные блоки, из которых состоит реализация GPU, их назначение. Проводится детальный анализ архитектуры вычислительных ядер. Подводятся итоги проделанной работы.

Ключевые слова: графический ускоритель, архитектура, аппаратная реализация, сложно-функциальный блок, потоки, регистры, схема, вычислительная ядра, разработка, GPU.

Введение

Под 2D графическим ускорителем понимается один или комплекс цифровых аппаратных блоков, специализированных для эффективного выполнения операций с двухмерной графикой. Основная область применения двухмерной графики помимо очевидных 2D игр — реализация графического интерфейса пользователя (GUI), так как целевым пользователем приложений или каких-либо других систем, например умной кофеварки, часто не является технически образованный специалист. Использование других видов интерфейсов, например, интерфейса командной строки, будет создавать затруднения в работе.

Однако использование мощностей только процессора для работы с графикой не целесообразно. Процессор создается как гибкое и программируемое аппаратное решение, а потому он лишен специализированных оптимизаций для конкретной области. Эту проблему решают аппаратные блоки ускорителей, в отличие от процессора они предназначены для выполнения одного типа операций эффективно по времени и ресурсам. Несмотря на то, что такие решения уступают в гибкости, выигрыш в скорости значительно выше, чем установка дополнительного ядра на чип.

1. Постановка задачи

Компанией АО «ПКК» Миландр была поставлена задача разработать аппаратную реализацию 2D графического ускорителя. Перед началом аппаратной разработки необходимо описать архитектуру, которой должен соответствовать сложно-функциональный (СФ) блок.

При разработке архитектуры учитываются проделанные ранее обзор и анализ 2D графических ускорителей [1].

Для разработки архитектуры необходимо учитывать контекст, в котором будет использоваться аппаратный блок. Основной целью разработки графического ускорителя (GPU) является снижение нагрузки на центральный процессор (CPU). Этого можно добиться путём вычисления сложных матричных операций на GPU, чтобы CPU мог заниматься более приоритетными задачами. Это сделает систему в целом более быстрой и отзывчивой.

2. Актуальность разработки

Не смотря на узкую специализацию применения аппаратного блока, на самом деле подобные графические ускорители используются почти повсеместно, при условии наличия у устройства дисплея высокого разрешения. Ввиду того, что изделия решают разные задачи, каждое из них имеет свои требования к GPU, в зависимости от сложности задачи.

Ниже представлена таблица, содержащая несколько примеров того, какие именно могут быть требования к GPU, в зависимости от устройства, которое будет его использовать.

Таблица 1

Требование к GPU

Требование	Пример устройств
Минимальное энергопотребление и низкая стоимость	Умные часы, термостат, простая панель управления
Высокая надёжность, работа с несколькими слоями изображения и плавность анимации	Автомобильный экран (приборная панель, медиацентр), кассовый аппарат, промышленный контроллер
Высокая производительность интерфейсов и продвинутые эффекты	Бюджетный смартфон, планшет

3. Подход к разработке

При разработке архитектуры как правило используют подход, при котором систему описывают иерархично сверху вниз. Этот подход позволяет последовательно уточнять детали реализации, давая уже на первых этапах основное понимание архитектуры.

4. Верхнеуровневая модель-схема GPU

Исходя из описанного иерархичного подхода к разработке опишем верхний уровень GPU.

На рис. 1 изображено схематичное представление верхнего уровня СФ-блока. Блок Device Control Register отвечает за верхнеуровневую настройку СФ-блока, которая заключается в конфигурировании количества параллельно исполняющихся потоков на ядрах.

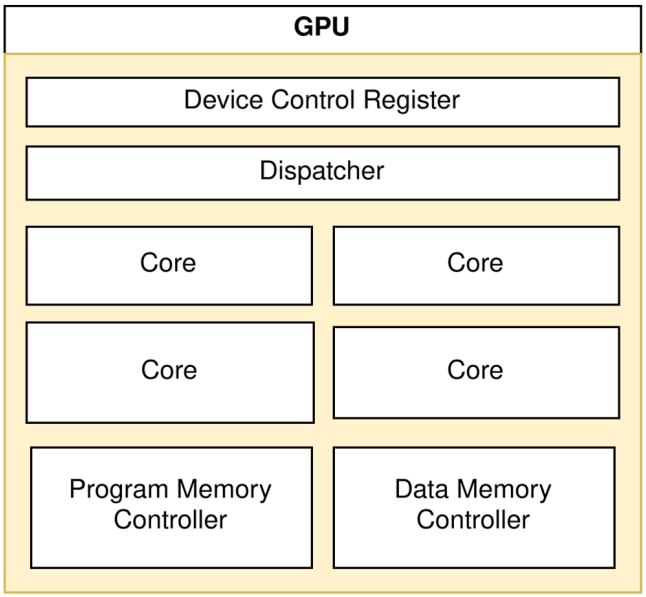

Рис. 1

Следующим ключевым подблоком является Dispatcher, целью которого является назначение ядрам вычислительных задач и выставление флагов завершения вычислений.

Следующими элементами схемы являются непосредственно вычислительные ядра. Каждое ядро может заниматься вычислением одной программы. Внутреннее устройство ядра описывается в разделе 5. Также на верхнем уровне блока располагаются контроллеры памяти, функция которых заключается в том, чтобы обрабатывать запросы к внешней памяти, основываясь на её пропускной способности, от всех ядер и возвращать ядрам ответ.

5. Структура вычислительного ядра GPU

Опускаясь на уровень ниже мы можем описать внутреннее устройство вычислительного ядра.

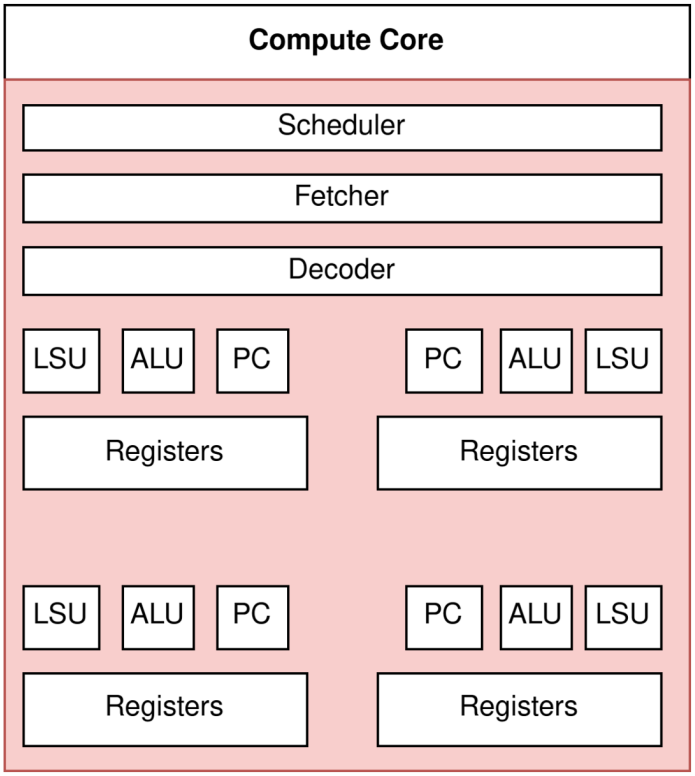

Рис. 2. Структурная блок-схема ядра

На рис. 2 изображено схематичное устройство одного вычислительного ядра. Ниже приведено описание каждого подблока.

5.1. Scheduler

СФ-блок Scheduler отвечает за весь поток управления одного вычислительного ядра, обрабатывающего 1 блок данных. Задано несколько активных фаз, в ходе которых ядро совершает различные действия. Ниже в табл. 2 представлено описание состояний ядра.

Каждое ядро обладает собственным планировщиком, который помогает в едином потоке управления проводить вычисления нескольких нитей исполнения.

Таблица 2

Описание состояний ядра

Состояние	**Описание**
Fetch	Извлечение инструкции по текущему счетчику инструкций (PC) из памяти программ
Decode	Декодирование инструкции в управляющие сигналы
Request	Если есть инструкция, обращающаяся к памяти, инициировать асинхронные запросы к памяти от LSU
Wait	Ожидание завершения всех асинхронных запросов к памяти
Execute	Выполнение вычислений с полученными данными из регистров/памяти
Update	Обновление значений регистров и счетчика команд

5.2. Fetcher

Каждое ядро имеет свой собственный Fetcher, который извлекает инструкцию по текущему счетчику инструкций из глобальной памяти данных. Хранит инструкцию после извлечения из памяти и ожидает начала этапа декодирования для перехода в режим ожидания.

5.3. Decoder

Каждое ядро обладает своим индивидуальным Decoder, который преобразует инструкции в управляющие сигналы, необходимые для её выполнения.

Ниже представлена таблица поддерживаемых инструкций:

Таблица 3

Описание инструкций

Название инструкции	**Описание**
BRnzp	Инструкция перехода на другую строку памяти программы, если регистр NZP соответствует условию nzp в инструкции
CMP	Сравнение значений двух регистров и сохранение результата в регистре NZP для использования в следующей инструкции BRnzp
ADD, SUB, MUL, DIV	Базовые арифметические операции для работы с тензорами
LDR	Загрузка данных из глобальной памяти
STR	Сохранение данных в глобальной памяти
CONST	Загрузка константы в регистр
RET	Сигнал о завершении выполнения текущего потока

Здесь регистром NZP называют регистр, значение которого устанавливается инструкцией CMP. Условные переходы осуществляются благодаря флагам в регистре NZP.

5.4. LSU (Load-Store Unit)

Каждый поток в ядре имеет свой собственный LSU. При получении запроса на доступ к памяти осуществляет чтение ячейки памяти по заданному адресу, после чего переходит в режим ожидания. Когда блок получает подтверждение, что чтение из памяти завершилось, то блок отдаёт полученные данные и переходит в состояние ожидания обновления. При получении запроса на запись в память осуществляет запись заданного значения в ячейку памяти по пе-

реданному адресу, после чего переходит в режим ожидания. Когда блок получает подтверждение, что запись в память завершилась, то блок переходит в состояние ожидания обновления. Данный подблок отвечает за обработку инструкций LDR и STR из табл. 2.

5.5. ALU (Arithmetic-Logic Unit)

Каждый поток в ядре имеет свой ALU, который выполняет вычисления со значениями регистров. Данный подблок реализует выполнение таких инструкций как: ADD, SUB, MUL, DIV, подробно описанных в табл. 2.

5.6. PC (Program Counter)

Вычисляет следующий PC для каждого потока, до которого нужно обновиться. Каждый поток в каждом ядре имеет свой собственный расчет следующего PC. Значение регистра NZP устанавливается инструкцией CMP (на основе операций сравнения: >, =, <) для инициирования инструкции BRnzp, описанной в табл. 2.

5.7. Registers

Каждый поток в ядре имеет свой собственный набор с 13 свободными регистрами и 3 регистрами только для чтения. Регистры только для чтения содержат значения %blockIdx, %blockDim, %threadIdx, которые важны для архитектуры SIMD [3].

Заключение

На основе представленных данных была разработана архитектура для проектирования синтезируемого RTL-описания 2D графического ускорителя для компании-заказчика АО «ПКК Миландр». В качестве основного подхода к разработке была использована иерархичная модель представления и описания сложных систем. В ходе разработки учитывались проделанные ранее обзор и анализ 2D графических ускорителей.

Литература

1. *Дудкин И. А. Старухин Д. М. Черкасов В. В.* Обзор и анализ 2D графических ускорителей. Межвузовская научная конференция молодых ученых и студентов // Математика, информационные технологии, приложения. – 2025.

2. Advanced Micro Devices, Inc. Southern Islands Series Instruction Set Architecture. – URL: https://www.amd.com/content/dam/amd/en/documents/radeon-tech-docs/instruction-set-architectures/southern-islands-instruction-set-architecture.pdf (дата обращения: 26.11.2025).

3. *Flynn Michael J.* Some Computer Organizations and Their Effectiveness // IEEE Transactions on Computers.

УДК 004.056:005.8

МЕТОДЫ АВТОМАТИЗАЦИИ ПРОЦЕССОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В КОРПОРАТИВНЫХ СЕТЯХ

А. А. Карташов

Воронежский государственный университет

Аннотация. В данной статье рассматриваются основные направления автоматизации процессов ИБ, методы, применяемые в современных организациях, а также преимущества и ограничения внедрения автоматизированных решений. Статья может быть полезна руководителям начинающих компаний, системным администраторам, разработчикам и интеграторам, а также студентам и исследователям, изучающим автоматизацию процессов защиты корпоративных сетей.
Ключевые слова: информационная безопасность, корпоративные сети, автоматизация процессов, управление уязвимостями, мониторинг, анализ, инструменты автоматизации.

Введение

Современные корпоративные сети становятся всё более сложными и одновременно с этим стремительно увеличивается число киберугроз, направленных на нарушение конфиденциальности, целостности и доступности информации. Увеличение объёмов данных, постоянное изменение инфраструктуры и усложнение нормативных документов требуют внедрения автоматизированных механизмов контроля и оперативного реагирования. Традиционные подходы к обеспечению информационной безопасности, основанные на ручной обработке событий, проверке конфигураций и анализе инцидентов, уже не позволяют оперативно и эффективно противодействовать современным атакам. Автоматизация процессов информационной безопасности позволяет значительно снизить зависимость от человеческого фактора, повысить скорость обработки событий и обеспечить непрерывное выполнение регламентов по защите информационных систем.

1. Особенности обеспечения информационной безопасности в корпоративных сетях

Современные корпоративные сети отличаются наличием множества сегментов, облачных сервисов, виртуализированных платформ и мобильных устройств. Ключевые особенности таких сетей включают гибридную архитектуру, которая объединяет локальные ресурсы и облачные сервисы, динамическую масштабируемость с быстрым изменением числа узлов, широкий спектр используемых протоколов и технологий, постоянный обмен трафиком, требующий в идеале непрерывного мониторинга, а также поддержку удалённого доступа, увеличивающего поверхность атаки.

Организации сталкиваются с разнообразными угрозами, включая внешние и внутренние атаки (DDoS, фишинг, вредоносные вложения), эксплуатацию уязвимостей в программном обеспечении и конфигурациях, несанкционированный доступ пользователей и устройств, компрометацию учетных записей, распространение вредоносного ПО внутри сети, утечку данных и ошибки конфигурации, вызванные человеческим фактором.

Ручное управление ИБ сталкивается с рядом трудностей. Это большой объём данных журналов и событий, требующий постоянной обработки, высокая вероятность ошибок при ручном анализе, низкая скорость реагирования на инциденты, сложность поддержания единых политик безопасности, необходимость постоянного контроля конфигураций и обновлений, а также рост

требований нормативных стандартов (ГОСТ, ISO, НПА РФ). Все эти факторы подчёркивают необходимость перехода к автоматизированным методам защиты корпоративных сетей.

2. Основные направления автоматизации процессов информационной безопасности

Для того чтобы обеспечить наибольший уровень безопасности в корпоративной сети, необходимо автоматизировать следующие процессы:

– Мониторинг и анализ событий безопасности
– Выявление угроз и аномалий
– Управление уязвимостями
– Контроль конфигураций и политик безопасности
– Управление доступом
– Реагирование на инциденты
– Резервное копирование и контроль целостности
– Отчётность и документооборот в сфере ИБ

Рассмотрим эти процессы подробнее, а также методы и инструменты, которые могут использоваться для их реализации.

Мониторинг и анализ событий безопасности: одно из ключевых направлений развития корпоративной защиты — это автоматическое отслеживание событий безопасности. Для этого применяются системы класса SIEM (Security Information and Event Management), которые собирают данные со всех устройств сети, приводят их к единому формату и сопоставляют между собой, выявляя подозрительные активности. Благодаря этому анализ происходит быстрее, а специалисты SOC (Security Operations Center) могут сосредоточиться на наиболее критичных инцидентах, а не на ручной сортировке огромного объема журналов.

Выявления угроз и аномалий: Важной частью защиты являются системы, способные самостоятельно обнаруживать угрозы. Системы IDS/IPS для анализа сетевого трафика и решения контроля взаимодействий внутри сети выявляют как известные типы атак, так и новые аномальные действия. Поведенческие методы анализа позволяют фиксировать нетипичное поведение пользователей и устройств, что существенно повышает вероятность своевременного обнаружения скрытых инцидентов.

Управление уязвимостями: здесь должны использоваться автоматические сканеры уязвимостей, которые регулярно исследуют инфраструктуру, классифицируют найденные угрозы и формируют отчёты о рисках. Такие системы помогут быстро выявить слабые элементы, поддерживать актуальность конфигураций и предоставлять рекомендации по устранению нарушений. В сочетании с инвентаризацией инфраструктуры это позволяет системно контролировать уровень защищенности.

Контроль конфигураций и политик безопасности: Единообразие конфигураций — важная часть устойчивой ИБ. Системы NAC, GPO и инструменты управления конфигурациями обеспечивают применение политик безопасности ко всем устройствам, отслеживают любые изменения и предотвращают отклонения. Сюда относится проверка устройств на соответствие политике доступа, автоматическое исправление неправильных конфигураций, централизованное управление параметрами безопасности, аудит состояния инфраструктур.

Управление доступом: Системы IAM (Identity and Access Management) позволяют контролировать права пользователей и обеспечивать защиту учётных записей. Это включает управление жизненным циклом учётных записей, назначение ролей и использование многофакторной аутентификации.

Реагирования на инциденты: платформы SOAR (Security Orchestration, Automation and Response) берут на себя выполнение рутинных операций при реагировании на инциденты. Они собирают информацию, анализируют данные по заранее подготовленным сценариям, блокиру-

ют подозрительные IP-адреса или учетные записи и могут автоматически изолировать заражённые хосты. Это ускоряет процесс реагирования и снижает нагрузку на команды безопасности.

Резервное копирование и контроль целостности: Резервное копирование обеспечивает возможность восстановления данных после сбоев или атак, включая шифровальщиков. FIM-системы отслеживают изменения в критически важных файлах и системных компонентах, что позволяет оперативно выявлять вредоносные модификации.

Отчётность и документооборот: Системы автоматизации отчётности позволяют поддерживать соответствие требованиям регуляторов и нормативных актов. Они формируют отчёты, ведут журналы, фиксируют выполнение процедур и облегчают документирование инцидентов. Сюда относятся формирование отчётов по стандартам ISO, ГОСТ, 152-ФЗ, PCI DSS, ведение электронных журналов, управление регламентами обработки инцидентов, контроль сроков выполнения задач ИБ.

3. Преимущества внедрения систем автоматизации

Внедрение современных автоматизированных механизмов защиты позволяет значительно повысить уровень безопасности корпоративной сети. К преимуществам можно отнести:

– снижение влияния человеческого фактора за счёт минимизации ручных действий;

– ускорение реагирования на инциденты благодаря автоматическому выполнению заранее определённых сценариев;

– повышение точности контроля за счёт анализа больших объёмов данных и корреляции событий из разных источников;

– оптимизация затрат за счёт сокращения рутинных операций и эффективного распределения ресурсов.

Однако нужно учитывать и следующие факторы, по причине которых до сих пор не все системы эффективны и полностью соответствуют требованиям безопасности.

Сложность интеграции автоматизированных решений с существующей инфраструктурой и устаревшими системами, так же высокая стоимость внедрения и обслуживания SIEM, SOAR, IAM и других платформ и самое значимое, что не у каждой организации имеется достаточно квалифицированных специалистов, способных обслуживать и корректировать подобные системы.

Заключение

Бизнес постепенно переходит от ручного управления информационной безопасностью к автоматизации процессов и внедрению ИБ-политик. Однако, как показывает практика политики плохо исполняются и часто являются формальными. Для построения зрелой и эффективной защиты важно оценить возможности организации, спланировать действия с учётом специфики корпоративной среды и при необходимости привлекать внешних экспертов. Такой подход позволит организации работать в условиях неопределённости, снизить угрозы и повысить устойчивость корпоративной инфраструктуры.

Литература

1. *Кубаренко А. С.* Модернизация корпоративной компьютерной сети предприятия // Концепт. – 2016. – Т. 11. – С. 3131–3135.

2. Автоматизация политик информационной безопасности в корпоративных сетях. – URL: https://ict-online.ru/it_class/Avtomatizatsiya-politik-informatsionnoi-bezopasnosti-v-korporativnykh-setyakh-319176 (дата обращения: 22.11.2025).

3. Корпоративные решения безопасности | Лаборатория Касперского. – URL: https://www.kaspersky.kz/business/enterprise-security (дата обращения: 15.11.2025).

УДК 004.912

ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА СЕМАНТИЧЕСКОГО АНАЛИЗА ОБРАЗОВАТЕЛЬНЫХ ПРОГРАММ НА ОСНОВЕ RUBERT И ОНТОЛОГИЧЕСКИХ ГРАФОВ

И. С. Кожевников, Е. Г. Бергер

МИРЭА – Российский технологический университет

Аннотация. В работе представлена интеллектуальная система семантического анализа образовательных программ, основанная на модели RuBERT и онтологических графах. Сформулировано фундаментальное противоречие между ростом знаний и скоростью их формализации, проявляющееся в компетентностных разрывах и онтологических деформациях образовательных программ. Обоснован выбор ОП как управляемого объекта адаптации в двухконтурной модели анализа: внутреннего (оценка онтологической целостности) и внешнего (оценка соответствия требованиям рынка труда). Разработаны две методики и интегрированы в архитектуру СППР, обеспечивающей автоматизированное сопоставление компетенций, выявление разрывов и формирование рекомендаций по модернизации ОП. Представлены результаты апробации, включающие выявленные зоны риска и рекомендации по обновлению компетентностного каркаса. Показано, что предложенный подход сокращает время анализа программ и повышает точность выявления несоответствий.

Ключевые слова: RuBERT, онтологический граф, образовательная программа, семантический анализ, компетентностный разрыв, онтологическая целостность рынок труда, интеллектуальная система, СППР, NLP, анализ вакансий, адаптация образовательных программ.

Введение

Мир развивается через преодоление противоречий. Их выявление, формализация и разрешение лежат в основе научного прогресса: от физических теорий до социальных систем. В современных условиях именно информатика сталкивается с наиболее острыми и фундаментальными противоречиями, поскольку цифровая среда становится центральным механизмом хранения, переработки и распространения знаний. Рост объёма данных, усложнение цифровых процессов, ускорение обновления технологий и методов создают качественно новые вызовы, связанные с необходимостью структурировать и интерпретировать быстро растущие массивы информации.

Одним из ключевых противоречий современной информатики является несоответствие между темпом накопления новых знаний и возможностями их структурирования, формальной обработки и передачи. Это противоречие универсально: оно затрагивает управление организациями, цифровые платформы, государственные информационные системы и — в наиболее концентрированном виде — систему высшего образования [6, 9].

Высшее образование выступает ярким примером того, как фундаментальное информационное противоречие проявляется на практике. С одной стороны, современная индустрия формирует новые компетенции быстрее, чем образовательные структуры могут их интегрировать. С другой стороны, сами образовательные программы (ОП) часто оказываются внутренне несогласованными — между уровнями ФГОС (Федеральные государственные образовательные стандарты) → ОП → РПД (Рабочая программа дисциплины) возникают смысловые, логические и структурные несоответствия, нарушающие целостность компетентностного каркаса [1,5].

Таким образом, проблема адаптации образовательных программ является частным случаем фундаментального противоречия информатики: темп появления новых компетенций превышает возможности их формальной фиксации, описания и передачи в образовательных системах [2].

Хронология исследования и постановка задачи

Выбор образовательной программы (ОП) в качестве центрального объекта анализа и адаптации обусловлен особенностями структуры нормативно-методической документации высшего образования. ФГОС задаёт лишь инвариантные рамочные требования и обновляется с большими интервалами, вследствие чего не может оперативно реагировать на быстрые изменения рынка труда. РПД, напротив, формируются на нижнем уровне и зачастую являются фрагментарными документами, зависящими от отдельных преподавателей. Коррекция РПД без корректировки ОП приводит к точечным изменениям и не устраняет системных противоречий.

Основная образовательная программа занимает промежуточное, системообразующее положение: она определяет целевые компетенции, структуру подготовки, профилизацию и связи между дисциплинами. Именно ОП обладает достаточной управляемостью и гибкостью для модернизации: её содержание утверждается на уровне факультета или университета и может корректироваться без изменения нормативной рамки ФГОС. При этом обновление ОП автоматически транслируется на РПД, обеспечивая каскадный эффект согласования.

Первоначальный исследовательский вопрос был связан с оценкой соответствия образовательных программ требованиям рынка труда. На этом этапе анализ текстов вакансий и компетенций показал наличие устойчивого компетентностного разрыва: покрытие востребованных компетенций по техническим направлениям составляло около 54–71 %, а дефицит цифровых и междисциплинарных навыков был особенно выраженным.

На данном этапе ключевым техническим решением стало использование современных языковых моделей. Первые эксперименты показали, что RuBERT обеспечивает устойчивые и интерпретируемые векторные представления компетенций, позволяя сопоставлять элементы образовательных программ с текстами вакансий на уровне смысловых сходств, а не отдельных ключевых слов. Это стало отправной точкой для разработки первой версии методики внешнего анализа.

Однако дальнейшее исследование выявило, что проблема не ограничивается только «внешним контуром» — сопоставлением с рынка труда. Анализ структуры образовательных программ показал наличие внутренних несогласованностей, которые препятствуют корректной интерпретации компетенций ещё до выхода на сравнение с вакансиями. Терминологические расхождения между уровнями ФГОС, ОП и РПД, несоответствия между компетенциями и индикаторами, нарушения в структуре целей дисциплин — всё это формировало онтологическую деформацию ОП, затрудняющую её обновление и развитие [7].

Чтобы формализовать эти разрывы, в исследование была добавлена вторая технологическая компонента — онтологическое моделирование. Представление ОП в виде графа компетенций, индикаторов и целей дисциплин позволило выявлять структурные и логические несогласованности, которые невозможно обнаружить только средствами NLP. Таким образом, использование графов стало естественным продолжением применения языковых моделей, дополнив их структурной интерпретируемостью.

Это привело к ключевому выводу: прежде чем адаптировать образовательную программу к внешним требованиям, необходимо обеспечить её внутреннюю логико-семантическую согласованность. Только после устранения онтологических деформаций возможно корректное и достоверное сопоставление программы с требованиями рынка труда.

Предлагаемое решение

На основании проведенного анализа был сформирован двухконтурный подход:

Внутренний контур — оценка онтологической целостности образовательной программы (согласованность ФГОС → ОП → РПД) [1, 5, 8].

Введено понятие «онтологическая целостность образовательной программы», определяемое как степень логико-семантической согласованности элементов многоуровневой структуры образовательных программ (ФГОС — ОП — РПД).

Для количественного выражения логико-семантической целостности образовательных программ разработан показатель онтологической целостности (IOC, Index of Ontological Coherence), интегрирующий три взаимодополняющих критерия:

• семантическую согласованность — степень смысловой близости между элементами разных уровней;

• структурную связность — полноту и замкнутость онтологического графа;

• логическую непротиворечивость — отсутствие конфликтов и противоречий в логическом выводе онтологической модели.

Интегральный показатель рассчитывается по формуле (1):

$$IOC = W_1 \cdot \overline{S_{sem}} + W_2 \cdot D_{str} + W_3 \cdot (1 - E_{log}), \quad (1)$$

где $\overline{S_{sem}}$ — среднее значение семантической близости между связанными элементами (компетенциями, индикаторами и целями дисциплин), вычисляемое на основе метрик cosine similarity и soft cosine similarity;

D_{str} — коэффициент структурной связности онтологического графа, определяемый как отношение числа установленных рёбер к максимально возможному числу связей;

E_{log} — доля логических несогласованностей, выявляемых средствами логического вывода (reasoning) в среде Protégé или GraphDB;

W_1, W_2, W_3 — весовые коэффициенты, отражающие значимость соответствующих компонентов.

Для интерпретации значения показателя IOC предложена шкала оценки онтологической целостности образовательной программы:

Высокая (IOC 0,8–1,0): программа онтологически согласована, структура устойчива

Средняя (0,6 < IOC < 0,79): частичная целостность, присутствуют отдельные смысловые или структурные разрывы

Низкая (IOC < 0,6): онтологическая фрагментация, требуется пересмотр логико-семантических связей.

Полученное значение индекса отражает уровень онтологической целостности образовательной программы и используется в качестве критерия для последующей диагностики разрывов.

Введено понятие «онтологический разрыв», характеризующий нарушение указанной целостности и проявляющийся в утрате или искажении смысловых связей между уровнями нормативной документации. Для выявления и локализации зон несогласованности разработан показатель онтологического разрыва (IOD, Index of Ontological Discontinuity), позволяющий количественно определить участки, где нарушена смысловая или логическая связность между элементами. Показатель вычисляется для каждой пары связанных сущностей (компетенция — индикатор, индикатор — цель дисциплины) по формуле (5).

$$IOD_{ij} = 1 - S_{ij}, \quad (2)$$

где S_{ij} — семантическая близость между элементами, полученная с использованием метрик cosine similarity и soft cosine similarity.

Среднее значение $IOD_{avg} = 1 - S_{sem}$ отражает общий уровень разрывности онтологической структуры, а взаимосвязь между показателями целостности и разрыва определяется зависимостью $IOC = 1 - IOD_{avg}$.

Анализ разрывов проводится на трёх уровнях структуры образовательной программы:

1) между компетенциями и индикаторами (ФГОС — ОП) — разрыв между нормативными требованиями и их операционализацией;

2) между индикаторами и целями дисциплин (ОП — РПД) — несоответствие планируемых результатов и содержания учебных дисциплин;

3) внутри РПД — нарушение внутренней логической согласованности целей и разделов дисциплины.

Пороговые значения показателя IOD_{ij} позволяют дифференцировать зоны риска: значения менее 0,25 характеризуют устойчивые связи, диапазон 0,25–0,40 указывает на частичное несоответствие, а превышение 0,40 свидетельствует о выраженном онтологическом разрыве.

Индекс IOC интерпретируется как интегральная мера системной согласованности образовательной программы, объединяющая семантическую, структурную и логическую компоненты, а индекс IOD — как мера локальной разрывности онтологической структуры.

Предложена типология онтологических разрывов, включающая терминологические, логические и структурные несоответствия, возникающие при проектировании и реализации образовательных программ.

Внешний контур — оценка соответствия образовательной программы требованиям рынка труда.

На основе матрицы семантических соответствий вычисляются показатели покрытия (C_{cov}) и избыточности (C_{red}) характеризующие степень соответствия компетенций образовательной программы требованиям рынка труда. Показатель C_{cov} отражает долю компетенций образовательной программы, имеющих смысловые аналоги в профессиональном корпусе, а C_{red} — долю компетенций, не находящих семантически близких соответствий в текстах рынка труда.

Введен интегральный показатель релевантности образовательной программы (IRL), который определяется как взвешенная сумма этих коэффициентов и средней семантической близости между соответствующими компетенциями образовательного и профессионального корпусов:

$$IRL = W_1 \cdot \overline{S_{sem}} + W_2 \cdot C_{cov} - W_3 \cdot C_{red}, \quad (3)$$

Для интерпретации интегрального показателя релевантности (IRL) введена система пороговых значений:

– $IRL \geq 0{,}75$ — высокая релевантность; возможны лишь слабые смысловые расхождения и единичные дефициты компетенций;

– $0{,}50 \leq IRL < 0{,}75$ — средняя релевантность; наблюдается сочетание дефицитных и смысловых разрывов, возможна локальная избыточность;

– $IRL < 0{,}50$ — низкая релевантность; характерен выраженный дефицит ключевых компетенций и структурные расхождения с требованиями рынка.

Избыточность компетенций определяется отдельно на основе доли компетенций ОП, не находящих прямого соответствия в требованиях рынка труда.

Для реализации двухконтурного подхода разработаны две методики:

Методика 1: семантическая оценка онтологической целостности ОП;

Использует:

- RuBERT-эмбеддинги,
- онтологические графы OWL/RDF,
- метрики semantic similarity,
- структурно-логический анализ (разрывы: терминологические, логические, структурные).

Методика 2: интеллектуальный анализ соответствия ОП требованиям рынка труда.

Использует:

- RuBERT,
- косинусное и soft-cosine сходство,
- кластеризацию,
- интегральный показатель релевантности IRL.

Обе методики объединяются в интеллектуальную систему поддержки принятия решений (СППР) [4], где RuBERT [2,3,8] работает как ядро смысловой обработки, а графовые модели обеспечивают интерпретацию связей.

Архитектура СППР реализует модульный принцип построения (рис. 1) и включает следующие основные подсистемы:

Модуль импорта, аннотации и лемматизации данных

Обеспечивает загрузку исходных документов (ФГОС, ОП, РПД, профессиональные стандарты, вакансии), их нормализацию, токенизацию и аннотацию сущностей, связанных с компетенциями и трудовыми функциями.

Подсистема анализа онтологической целостности (внутренний контур)

Выполняет диагностику логико-семантической согласованности образовательных программ между уровнями ФГОС – ОП – РПД.

В качестве методов используются: векторизация текстов с помощью RuBERT, построение онтологических графов (OWL/RDF) и анализ связности на основе метрик cosine и soft cosine similarity.

Результатом является вычисление значений показателей *IOC*, *IOD* и *IRL* характеризующих степень внутренней согласованности и локальных разрывов.

Подсистема анализа соответствия требованиям рынка труда (внешний контур)

Осуществляет семантическое сопоставление образовательной программы с корпусом профессиональных описаний (вакансии, профстандарты).

Для вычисления степени соответствия применяются метрики семантической близости и показатели покрытия (C_{cov}), избыточности (C_{red}) и интегральный индекс релевантности (IRL).

Семантическое ядро (RuBERT + Онтологический граф Neo4j)

Обеспечивает единое пространство представления данных и реализацию семантического вывода.

RuBERT отвечает за контекстную векторизацию текстов компетенций, индикаторов и целей дисциплин, а Neo4j — за хранение и обработку онтологических связей между сущностями.

Хранилище данных и результатов анализа

Содержит корпус исходных текстов, векторные представления, результаты семантических вычислений (*IOC*, *IOD*, *IRL*) и историю предыдущих итераций анализа, что обеспечивает возможность повторного обращения к данным без повторной обработки.

Модуль визуализации и экспертной верификации

Предназначен для отображения графовых структур, зон онтологических разрывов и интегральных показателей в наглядной форме.

Эксперт получает возможность подтверждать или отклонять автоматически сформированные заключения, формируя базу прецедентов и корректировок.

Модуль формирования управленческих рекомендаций

На основании показателей *IOC*, *IOD*, *IRL* система извлекает из базы знаний типовые рекомендации.

Рекомендации представляют собой формализованные предложения по корректировке компетенций, индикаторов и содержания дисциплин в соответствии с выявленными несогласованностями.

Интеграция внутреннего и внешнего контуров анализа обеспечивает целостное представление образовательной программы в онтологическом пространстве.

СППР функционирует в итерационном режиме, обеспечивая циклический анализ образовательных программ.

На каждом цикле система:

- принимает исходные данные (ФГОС, ОП, РПД, профстандарты, вакансии);
- проводит внутренний и внешний анализ;

• формирует рекомендации;

• после внесения корректировок повторно проводит анализ, уточняя показатели IOC, IOD и IRL.

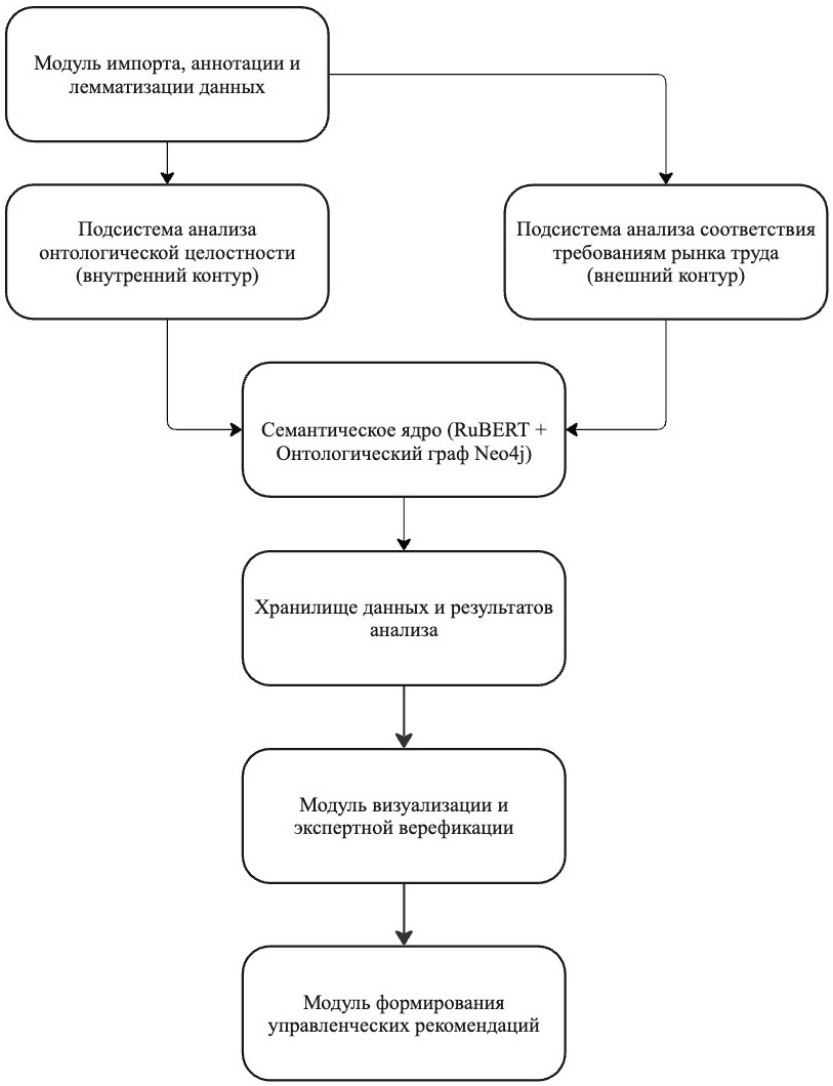

Рис. 1. Архитектура СППР

Пример работы СППР: результаты анализа образовательной программы и рекомендации по адаптации

Для демонстрации практической применимости разработанной системы поддержки принятия решений был проведён анализ образовательной программы по направлению 09.03.03 «Прикладная информатика» одного из технических университетов. Анализ выполнялся по двухконтурной схеме: внутренний контур (онтологическая целостность) и внешний контур (соответствие требованиям рынка труда).

Использовались тексты 1 ФГОС, 1 ОП, 3 рабочие программы дисциплины, а также корпус из 50 вакансий ИТ-профиля.

Выявленные несоответствия

Результаты внутреннего анализа (соответствие ОП требованиям рынка труда)

Показатели целостности

IOC = 0,74 (при рекомендуемом $\geq$ 0,85)

IOD = 0,26 (выше нормируемого порога 0,25)

Это означает: программа семантически неполна, обнаружены частичные несоответствия и деформация структуры компетентностного каркаса.

Результаты внешнего анализа (соответствие ОП требованиям рынка труда)

Для сопоставления использовался корпус вакансий ИТ-профиля. Все тексты были преобразованы моделью RuBERT в семантические эмбеддинги и сопоставлены с компетенциями ОП с использованием soft cosine similarity.

Покрытие компетенций

C_{cov} = 0,68 — покрытие актуальных компетенций рынка

C_{red} = 0,21 — обнаружена избыточность (устаревшие или нерелевантные компетенции)

Интегральная метрика соответствия

IRL = 0,61

Это означает: программа частично соответствует актуальным профессиональным требованиям, но нуждается в структурной и смысловой модернизации.

Заключение

В работе предложен и реализован двухконтурный подход к интеллектуальному анализу образовательных программ, основанный на интеграции трансформерной модели RuBERT и онтологически-графового моделирования. Показано, что фундаментальное противоречие между ростом объёма знаний и ограниченной скоростью их формализации проявляется в системе высшего образования в виде двух проблем: (1) внутренней несогласованности образовательных программ между уровнями ФГОС — ОП — РПД и (2) несоответствия содержания образовательных программ динамично изменяющимся требованиям рынка труда.

Предложенный подход обеспечивает научно обоснованный и технологически реализуемый механизм повышения качества образовательных программ в условиях динамичного рынка труда. Повышает качество подготовки ОП за счет внутренней согласованности и степени покрытия образовательных программ актуальными компетенциями рынка труда. Повышает эффективность подготовки ОП за счет сокращения сроков анализа, диагностики и актуализации образовательных программ.

Перспективы дальнейших исследований

Работа открывает ряд направлений для дальнейшего развития:

1. Автоматическая классификация типов онтологических разрывов

Сейчас определение типа разрыва осуществляется экспертно. Разработка обучаемых моделей для автоматической типизации (терминологический / логический / структурный) станет естественным продолжением исследования.

2. Расширение семантического ядра

- дообучение RuBERT на отраслевых корпусах,
- использование современных моделей (RuBERT-large, RuRoBERTa, ruGPT),
- внедрение моделей sentence-transformers для более точных эмбеддингов.

3. Расширение онтологии компетенций

Создание единой интегрированной ontosystem:

- компетенции ФГОС,
- индикаторы ОП,
- трудовые функции профстандартов,
- требования вакансий.

4. Интеграция СППР в учебные процессы

Система может использоваться:

• для регулярного мониторинга и обновления ОП,

• в практиках студентов по анализу образовательных документов,

• в процедурах внутренней аккредитации вузов.

5. Формирование базы корректирующих рекомендаций

Накапливаемые экспертные подтверждения позволят перейти от статического набора рекомендаций к обучаемой системе управления развитием ОП.

Литература

1. *Минаев Д. В.* Когнитивное управление образовательными программами на основе онтологических моделей // Вестник МГТУ им. Баумана. – 2021. – № 4. – С. 55–62.

2. *Ботов А. А.* Методы интеллектуальной поддержки проектирования образовательных траекторий с применением технологий машинного обучения // Информатика и образование. – 2022. – № 3. – С. 41–48.

3. *Kuratov Y., Arkhipov M.* Adaptation of BERT for Russian language: DeepPavlov Team Contribution // arXiv preprint arXiv:2010.15900. – 2020.

4. *Мухаметзянова Д. В., Ямалтдинов М. А.* Системы поддержки принятия решений в образовании: обзор и перспективы // Казанский педагогический журнал. – 2021. – № 5(147). – С. 45–52.

5. *Кудинов С. В., Исаев Е. А.* Онтологический подход к анализу компетенций в образовании // Наука и школа. – 2020. – № 4. – С. 34–41.

6. *Гасанова А. А., Никитина Н. А.* Цифровая трансформация образования: вызовы и возможности // Высшее образование в России. – 2021. – № 6. – С. 20–27.

7. *Беляев С. Ю.* Подход к формализации образовательных программ на основе онтологий // Вестник Томского государственного университета. – 2022. – № 478. – С. 56–63.

8. *Кожевников И. С.* Сравнительный анализ эффективности моделей обработки естественного языка для адаптации образовательных программ к требованиям рынка труда / И. С. Кожевников, В. Ф. Дубровский // Автоматизация в промышленности. – 2025. – № 4. – С. 55–60. – EDN FVPNAD.

9. *Кожевников И. С.* Системы поддержки принятия решений в цифровом обществе: преодоление компетентностного разрыва между образованием и рынком труда / И. С. Кожевников // «Вызовы современности и стратегии развития общества в условиях новой реальности» (шифр – МКВСС) : Сборник материалов XXXVIII Международной научно-практической конференции, Москва, 30 сентября 2025 года. – Москва: АНО ДПО «Университет ИТБО», 2025. – С. 141–150. – EDN IOQTOI.

УДК 004.056

СПОСОБЫ ЗАЩИТЫ ОБЛАЧНЫХ ХРАНИЛИЩ В УСЛОВИЯХ РОСТА КИБЕРУГРОЗ

Д. М. Кузнецов

Национальный исследовательский университет «МИЭТ»

Аннотация. В работе рассматривается общая информация о моделях обслуживания и развертывания облачных хранилищ, описаны отличия подходов к обеспечению безопасности в облаке от традиционных подходов. Основная цель данной работы – в условиях роста киберугроз рассмотреть угрозы безопасности в облачных технологиях и методы борьбы с ними.

Ключевые слова: облачное хранилище, модель обслуживания, модель развертывания, угроза безопасности.

Введение

Развитие облачных хранилищ началось в начале 2000-х годов и с тех пор стремительно ускоряется. Сегодня всё больше компаний переходят на модели IaaS (Infrastructure as a Service), PaaS (Platform as a Service) и SaaS (Software as a Service), чтобы повысить гибкость и масштабируемость ИТ-инфраструктуры и снизить расходы.

В России также усиливается тренд на внедрение облачных технологий. Этому способствует внешнеполитическая обстановка: уход зарубежных поставщиков, сложности с импортом оборудования и ПО, а также требования регуляторов по использованию отечественных решений. В связи с этим вопросы информационной безопасности в облаке становятся ключевыми при выборе модели хранения данных и переноса инфраструктуры.

Рост рынка облачных услуг сопровождается увеличением числа атак на облачные ресурсы. В отличие от локальных систем, облачная среда требует специфических подходов к защите данных, управлению доступом, мониторингу угроз и выполнению нормативных требований. Риски — утечки данных, несанкционированный доступ, DDoS-атаки и ошибки в настройках — могут привести к серьёзным финансовым и репутационным потерям.

Атака на инфраструктуру облачного провайдера особенно опасна: она может лишить клиентов критически важных данных, привести к их утечке или надолго остановить бизнес-процессы.

1. Модели обслуживания

Национальный институт стандартов и технологий (NIST) выделяет три основные модели обслуживания, описывающие ключевые категории облачных сервисов.

Платформа как услуга (PaaS) — модель, при которой провайдер предоставляет готовую виртуальную среду для разработки, тестирования и развертывания приложений. В неё могут входить почтовые серверы, системы управления базами данных, веб-серверы, среды разработки, резервные серверы и другие инструменты, уже настроенные для использования.

Инфраструктура как услуга (IaaS) обеспечивает доступ к базовым ресурсам вычислительной инфраструктуры, таким как виртуальные машины, сетевые компоненты и хранилища данных. В этом случае пользователи самостоятельно создают виртуальные серверы, устанавливают операционные системы и необходимое программное обеспечение [1].

Программное обеспечение как услуга (SaaS) представляет собой готовое приложение, полностью размещённое и обслуживаемое поставщиком. Пользователь получает к нему доступ через веб-браузер, мобильное приложение или лёгкий клиент.

SaaS подходит в тех случаях, когда необходимо просто использовать уже готовый функционал, не занимаясь управлением инфраструктурой или созданием собственных решений. Если с IaaS обычно работают системные администраторы, а с PaaS — разработчики, то SaaS ориентирован прежде всего на конечных пользователей [2].

2. Модели развертывания

Традиционно в облачных вычислениях принято выделять следующие основные модели развертывания:

Публичное облако. Данной инфраструктурой могут пользоваться все. Происходит это через интернет. Данная модель принадлежит той компании, которая предоставляет данные услуги.

Частное облако. Эту инфраструктуру использует только одна организация. Управляет ею сама компания или сторонний провайдер. При этом размещение происходит в компании или же за пределами организации.

Коммунальное облако. Данную инфраструктуру использует сразу несколько компаний, которые объединены какими-то интересами, например: единые требования безопасности, стандарты соответствия, политика. Управляют инфраструктурой участники этого сообщества или внешняя сторона. Расположение может быть внешним или локальным.

Гибридное облако. В данной структуре объединено два и более облака (публичные, частные, общественные). Эти системы независимые, но при этом их связывают проприетарные или стандартизированные технологии. Данная связка позволяет переносить приложения и данные. Также можно распределить нагрузку между различными средами.

В облачных вычислениях обычно выделяют четыре основных модели развертывания:

Публичное облако. Такая облачная инфраструктура открыта для широкого круга пользователей через интернет и принадлежит организации, предоставляющей облачные услуги.

Частное облако. Инфраструктура предназначена исключительно для одной организации. Управление может осуществляться как самой компанией, так и сторонним провайдером, а размещение возможно как внутри организации, так и за её пределами.

Коммунальное облако. Инфраструктура используется совместно несколькими организациями, объединёнными общими интересами — например, едиными требованиями безопасности, политиками или стандартами соответствия. Управление может выполняться участниками сообщества или внешней стороной, а расположение — как локальным, так и внешним.

Гибридное облако. Это комбинация двух или более облаков (частных, публичных или общественных), которые остаются независимыми системами, но связаны между собой стандартизированными или проприетарными технологиями. Такая связка обеспечивает переносимость данных и приложений, а также позволяет распределять нагрузки между разными облачными средами.

На рис. 1 приведено определение облака Национального института стандартов и технологий.

3. Отличие подходов к обеспечению безопасности в облаке от традиционных подходов

В настоящее время активно используются облачные вычисления. Это привело к тому, что привычный подход к безопасности изменился. Облачная модель, как более современное решение, существенно отличается от классических методов защиты [3].

Если используется традиционная схема, хранение данных предусмотрено в локальной сети. В облачных же инфраструктурах, которые распределены и обслуживают сразу многих клиентов, требуется защищать большое количество модулей и приложений, гибко настраивать доступ, обеспечивать безопасность сетевого взаимодействия и предотвращать утечки между виртуальными средами.

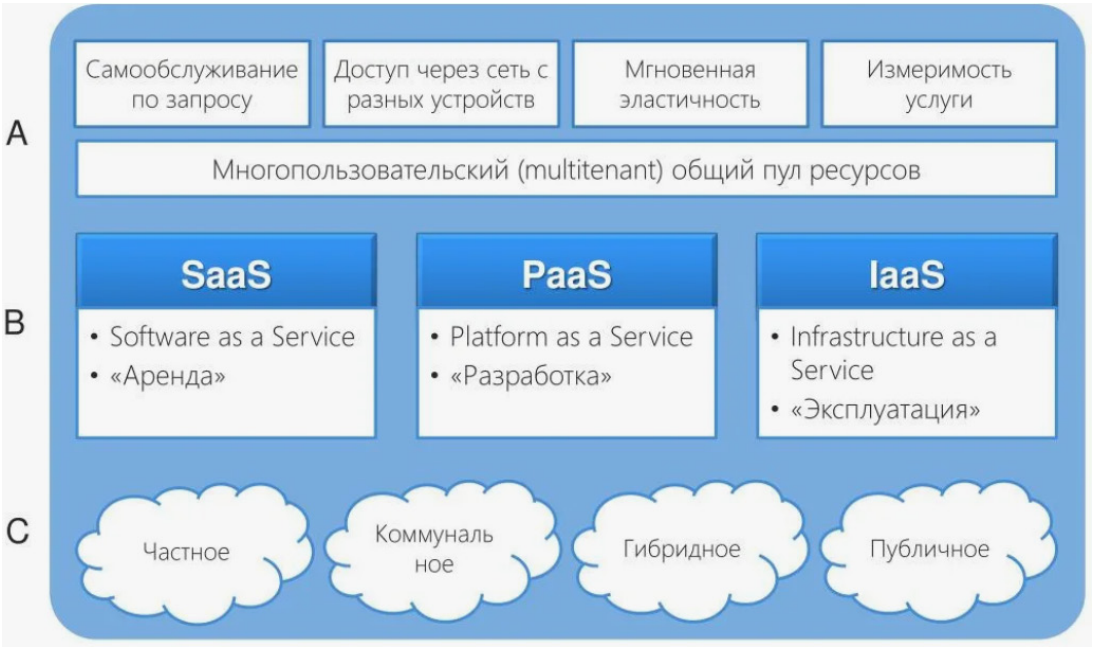

Рис. 1. Определение облака NIST: 5 характеристик(A), 3 модели(B), 4 способа реализации(C)

Масштабирование. Облачные системы легко расширяются благодаря модульной архитектуре. Однако постоянные обновления и быстрые изменения, связанные с ростом компании, требуют непрерывного контроля уровня безопасности.

Взаимодействие с конечным пользователем. Облачные сервисы интегрируются с множеством других систем, поэтому нужно обеспечивать защиту на всех уровнях — от устройств пользователей до приложений и сетей. Важную роль играет своевременное управление правами доступа и отслеживание уязвимостей, возникающих из-за некорректных настроек и небезопасных установок.

Близость к другим данным и системам. Поскольку облако постоянно связано с клиентскими системами, уязвимость, которая возникла в одном компоненте, вполне может компрометировать всю среду. Провайдерам, которые хранят у себя данные, приходится использовать серьезные меры безопасности, так как они сами являются объектами риска.

4. Угрозы безопасности в облачных технологиях

Уязвимости. Так как локальные и облачные серверы функционируют на тех же приложениях и операционных системах. Актуальны удаленные атаки и угрозы вредоносных программ. Применение большого количества виртуальных машин приводит к повышению риска компрометации. Для того чтобы обеспечить надежную защиту, требуется внедрение правил выявления угроз. Нужно блокировать вредоносную активность и регулярно проводить аудит безопасности.

Уязвимости виртуальной среды. Поскольку облачные и локальные серверы работают на тех же операционных системах и приложениях, угрозы вредоносного ПО и удалённые атаки остаются актуальными. Использование множества виртуальных машин повышает риск компрометации. Для защиты необходимо внедрять правила обнаружения угроз, применять механизмы блокировки вредоносной активности и регулярно проводить аудит безопасности.

Недостаточная сегментация и распределение ролей. Если зоны облачной среды плохо разделены, злоумышленник, получив доступ, может свободно перемещаться внутри инфраструктуры, увеличивая масштаб атаки. Каждая зона должна иметь собственные вычислительные,

сетевые ресурсы и ресурсы хранилища. Уровни управления и данных следует изолировать друг от друга, а обмен трафиком между зонами — ограничивать только разрешёнными путями.

API и интерфейсы. API — протоколы и стандарты. Через них идет взаимодействие пользователей с сервисами. Важно их надежно защитить. От этого зависит обеспечение безопасности платформы в целом, поэтому они должны соответствовать требованиям сертификации, иметь корректные механизмы аутентификации, контроля доступа и мониторинга, чтобы предотвратить угрозы вроде анонимного доступа, слабых токенов или недостаточного журналирования.

Инсайдерские угрозы. Доверенные сотрудники могут получить доступ к конфиденциальным данным и нанести ущерб организации — как намеренно, так и по неосторожности. Это может привести к снижению производительности или финансовым потерям.

Компрометация аккаунта или сервиса.

Если злоумышленнику удастся получить учетные данные, он будет использовать аккаунт для шпионажа и атак, изменит информацию или перенаправит пользователей те ресурсы, которые способны причинить вред.

Проблемы технического характерах. В многопользовательских системах разные клиенты используют общую инфраструктуру. Гипервизоры, которые обеспечивают изоляцию, могут быть уязвимы. Это приведет к тому, что атакующий получит доступ к виртуальным устройствам других пользователей.

Неопределенный уровень риска. Отсутствие обязательных процедур безопасности — обновлений, аудита, протоколирования — приводит к формированию неопределённого уровня риска, который может стать причиной серьёзных инцидентов [4].

Угрозы безопасности и методы борьбы с ними приведены в табл. 1.

Таблица 1

Угрозы и методы борьбы с ними

Угрозы безопасности	**Методы смягчения последствий**
Уязвимости в виртуальной среде	– Необходимо проводить систематическое сканирование на уязвимости и настроить процесс их устранения.
Недостаточное сегментирование и распределение ролей	– Важно, чтобы уровни данных и управления были в разных зонах. – Обмен данными между рабочими станциями в пределах одной зоны и между разными зонами должен быть ограничен разрешенными потоками трафика и путями. – Важно разграничить доступ непривилегированных и привилегированных пользователей. – Мониторинг сетевой активности и настройка межсетевых экранов.
API и интерфейсы	– Использование надежных механизмов аутентификации и контроля доступа. – Использование шифрования при передаче данных. – Анализ интерфейсов облачных провайдеров. – Правильное понимание цепочки зависимостей, связанных с API.
Инсайдеры	– Важно управлять имеющимися ресурсами. – Подготовить юридическое соглашение. – Строгое применение процедуры управления цепочкой поставок. – Обеспечение надлежащей ясности в вопросах безопасности и административных процессов.

Компрометация аккаунта или сервиса	– Правильное понимание политик безопасности и соглашений об уровне обслуживания. – Использование многофакторных методов аутентификации. – Строгий мониторинг для выявления несанкционированных действий. – Контроль обмена учетными данными между потребителями и службами.
Проблемы с технологией	– Применение современных методов контроля доступа и аутентификации. – Изучение среды на предмет изменений несанкционированного характера.
Неизвестный риски	– Раскрывать соответствующие журналы, детали и данные инфраструктуры.
Потеря данных	– Регулярное резервное копирование. – Надлежащие методы шифрования. – Реализация генерации, хранения и управления сильными ключами. – Законодательное указание методов усиления и обслуживания поставщиков.

Некоторые атаки на облачные хранилища атаки и методы борьбы с ними приведены в табл. 2.

Таблица 2

Атаки и методы борьбы с ними

Атаки	**Методы снижения воздействия**
SQL-инъекции	– Исключение динамически генерируемого SQL в коде. – Фильтрация для обеззараживания вводимых пользователем данных. – Архитектура на основе прокси для динамического обнаружения и извлечения пользовательского ввода.
Межсайтовый скриптинг (XSS)	– Активная фильтрация содержимого. – Технологии предотвращения утечки данных на основе контента. – Технологии обнаружения уязвимостей веб- приложений. – Подход, основанный на чертежах, позволяет минимизировать зависимость от веб-браузера. – Правильная настройка уровня защищенных сокетов (SSL). – Использование программ для защиты от вредоносного ПО.
Фишинг	– Идентифицировать спам-письма. – Установка специализированных средств защиты.
DNS-атаки	– Использовать меры безопасности DNS. Пример: расширения безопасности системы доменных имен (DNSSEC).

MITM-атаки	– Правильная настройка уровня защищенных сокетов (SSL). – Использование инструментов шифрования, например, Dsniff, Ettercap, Wsniff, Airjack и т. д.
DOS-атаки	– Использование совершенных схем аутентификации и авторизации. – Использование системы обнаружения вторжений (IDS)/системы предотвращения вторжений (IPS). А также других специальных средств защиты.
Sniffer Attacks	– Использование техники обнаружения снифферов на основе протокола разрешения адресов (ARP). – Использование техники обнаружения снифферов, основанной на времени передачи данных (Round-Trip Time, RTT).
Wrapping Attack	– Использование надлежащего механизма подписи. – Использование правильной конфигурации Secure Socket Layer (SSL).

Заключение

Облачные технологии являются мощным и эффективным инструментом для работы с информационными ресурсами. Использование современных защитных решений на разных уровнях виртуальной инфраструктуры позволяет значительно снизить количество потенциальных угроз.

Внедрение облачных систем способствует уменьшению расходов на программное обеспечение, повышает качество и эффективность бизнес-процессов, а также обеспечивает большую прозрачность и доступность данных.

Литература

1. О безопасности облачных сред. – 2024. [Электронный ресурс]. – URL: https://habr.com/ru/companies/otus/articles/818505/?ysclid=mido1ncxaf933483245 (дата обращения 2025-11-19).

2. Модели облачных сервисов: разница между IaaS, SaaS, PaaS и примеры. – 2024. [Электронный ресурс]. – URL: https://www.sim-networks.com/ru/blog/cloud-computing-service-models

3. *Зонова Д. Ю.* Способы защиты облачных хранилищ // Colloquium-journal. – 2024. – № 2 (195). URL: https://cyberleninka.ru/article/n/sposoby-zaschity-oblachnyh-hranilisch (дата обращения: 19.11.2025).

4. *Тонких А. С.* Угрозы безопасности в облачных технологиях и методы их устранения / А. С. Тонких, Е. Ю. Авксентьева // Международный журнал гуманитарных и естественных наук. – 2024. – № 1-2(88). – С. 232–238. – DOI 10.24412/2500-1000-2024-1-2-232-238. – EDN CZLBFQ.

УДК 002:001(470)

ВЛИЯНИЕ НЕКОНТРОЛИРУЕМОГО РАЗНООБРАЗИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАЗРАБОТКИ САЙТОВ НИИ НА УРОВЕНЬ ИНТЕГРАЦИИ ИХ ИНФОРМАЦИОННЫХ РЕСУРСОВ

В. И. Меденников

ФИЦ «Информатика и управление» РАН

Аннотация. Целью работы является оценка влияния неконтролируемого выбора программного обеспечения (ПО) разработки сайтов НИИ на примере аграрных на уровень интеграции их информационных ресурсов, диктуемой основными принципами цифровой экономики, научная реализация которых ведет к оформлению единой цифровой платформы научно-образовательных ресурсов. Показано, что одной из важных составляющих данной платформы является инструментальная в виде общесистемного ПО и электронного оборудования. Мониторинг сайтов аграрных НИИ выявил большие дезинтеграционные тенденции из-за нарастающего разнообразия ПО при разработке их сайтов, препятствующих интеграции научно-образовательных ресурсов.
Ключевые слова: интеграция, программное обеспечение, сайты, аграрные научно-исследовательские институты, информационные ресурсы, цифровая платформа.

Введение

Хотя мир вступил в эпоху цифровой экономики (ЦЭ), требующей глобальной интеграции информационных ресурсов (ИР), алгоритмического обеспечения по их обработке и инструментов цифровизации этих составляющих в виде программного обеспечения (ПО) и электронных приборов, технологии формирования сайтов научно-исследовательских институтов (НИИ) игнорируют данные мировые тенденции [1]. Проблема усугубляется вымыванием почти всех айтишников из науки и образования, непониманием Минобрнауки возможностей интеграционных технологий обеспечить реализацию с единых комплексных позиций исторически сложившиеся основные три функции мировой науки: эффективный инструмент трансфера научных знаний в производство; образовательная с целью адекватной восприимчивости инноваций всем населением; коммуникационная среди все более возрастающего числа научных и педагогических работников [2, 3].

В результате руководство НИИ при разработке сайтов в большинстве случаев отдает предпочтение готовым, примитивным, самое главное, бесплатным средствам, обладающим онтологической и информационной несовместимостью, которая, например не позволяет автоматически скачивать отчеты о самообследовании вузов на сайт Минобрнауки, предписанные приказом [4].

С другой стороны, резкое сокращение ИТ-специалистов в научных и образовательных организациях, место которых в публикациях по ЦЭ заняли менее знающие специалисты, наряду с акцентированием внимания Минобрнауки лишь на росте наукометрических показателей ожидаемо послужило причиной роста искаженной, недостоверной информации на сайтах НИИ. А это является одной из главных проблем применения искусственного интеллекта [5]. Поскольку ПО в такой ситуации играет важнейшую роль при создании сайтов, то в работе на примере аграрных НИИ исследуем ПО, используемое при разработке сайтов, на предмет требований ЦЭ по их совместимости.

1. Эволюционная необходимость внедрения интеграционных технологий

На рис. 1 представлена взаимосвязь составляющих элементов, рассмотренных выше интеграционных технологий. Хотя, конечно, на цифровизацию, как и на информатизацию, следуя тренду компьютеризации, существенно влияют следующие виды их обеспечения: организационное, информационное, математическое, программное, техническое, лингвистическое, метрологическое, правовое. В работе будем рассматривать в силу наибольшего влияния на интеграцию ресурсов лишь приведенное комплементарное трехкомпонентное множество [6].

Рис. 1. Основные виды обеспечения интеграционных технологий в ЦЭ

Как правило, разработка более совершенных алгоритмов мотивируется возможностями увеличения эффективности использования ИР, а все более объемные, подвергнутые структуризации, данные ведут к развитию новых, более совершенных алгоритмов. Такая комплементарность поддерживается инструментальной составляющей. В работе [7] показано, что с момента зарождения до настоящего момента в методах разработки ПО АСУ можно проследить четыре существенных этапа в их эволюции. Отличие одного этапа от другого характеризуется кардинальным изменением как механизмов всего цикла сбора, передачи, интеграции, накопления и последующей обработки информации, так и алгоритмов, реализованных в программном обеспечении (ПО). В работе [8] показана экономическая составляющая данной эволюции.

Начавшаяся цифровая трансформация производства, отнесенная к четвертому этапу, в силу потенциальной эффективности интеграционных технологий и своим возможностям с неизбежностью ведет к интеграции ИР, приложений и инструментария значительного большего числа экономических агентов из различных отраслей экономики, в единую информационно-управленческую среду. А формирование такой среды требует разработки соответствующих цифровых стандартов на указанные выше основные составляющие элементы ИКТ, поскольку настройка разработанных систем на резко возросшее число пользователей их становится крайне затратна. Об этом свидетельствуют исследования по цифровым двойникам [9, 10], когда, порой разработка такого двойника одного предприятия может достигать 100$ млрд.

2. Анализ программного обеспечения для разработки сайтов аграрных НИИ

Поскольку ПО играет большую роль при разработке сайтов, то проанализируем его состав и структуру на примере аграрных НИИ с целью его совместимости в интеграционных технологиях контента 183 их сайтов на основе результатов мониторинга в 2017г. [11] в сравнении с результатами аналогичного мониторинга, обработанного в 2023г. В таб. 1 приведен анализ сравнения 25 групп ПО, в которые разнесены 955 различных их видов, использованных на сайтах в первый мониторинг, и 42 групп, в которые разнесены уже 1585 использованных видов в 2023 г.

Таблица 1

Группы ПО на сайтах аграрных НИИ (2017/2023)

№ п/п	Наименование	Назначение	Число использований
1	CDN	Сервер технологии быстрой доставки контента страниц сайта	1/23
2	Мобильный фреймворк	ПО формирования мобильного варианта сайта	2/0
3	CMS	Сервер технологии автоматизированного управления контентом сайта	91/97
4	CRM	Системы управления интернет-магазином	3/1
5	Email	Сервисы электронной почты	0/3
6	JS-графика	Графический сервис сайтов	2/4
7	JS-библиотека	Библиотечный сервис с использованием языка JS	0/315
8	JS-фреймворк	Оформительские сервисы страниц на сайтах	182/45
9	PaaS	Сервисы облачных расчетов	0/2
10	SEO	Оптимизация процедуры поиска	0/8
11	Фреймворк UI	Сервисы дизайна в виде удобного фреймворка	0/81
12	Аналитика	Клиент-серверные технологии отслеживания динамики сайтов (посещения, состава, индексации и пр.)	145/168
13	Баг-трекер	Поисковые сервисы в части ошибок на сайте	0/2
14	БД	Хранение содержимого сайта в базе данных (БД)	1/41
15	Безопасность	Защитный сервис от произвольного посещения сайта	0/17
16	Блог	Сервисы типа CMS ведения дневника	20/44
17	Веб-сервер	Сервер, обслуживающий сайт	181/173
18	Веб-фреймворк	Каркас для написания веб-приложений	44/13
19	Видео-сервис	Сервис на сайте для показа видео	6/18
20	Виджет	Сервисы демонстрации программных вставок	12/16
21	Лендинг-генератор	Сервис для генерации спецстраниц	0/4
22	Статистические генераторы	Сервис для генерации статистики страниц	0/1
23	Карта	Сервисы показа карт в онлайн-режиме	2/6
24	Кеширование	Сервис для кеширования страниц с целью быстрой выдачи	3/3
25	Менеджер тегов	Сервисы для работы с тегами	0/9
26	Обратный прокси	Сервисы для ретрансляции запросов пользователей из другой сети	0/117
27	Онлайн-консультант	Сервисы для консультирования в онлайн-режиме	0/4
28	Операционная система	Рабочая операционная система сайта	39/22

29	Опечатки браузера	Сервисы для исправления опечаток на сайте в онлайн-режиме	0/2
30	Хостинг панель	Сервис-панель по управлению хостингом	7/4
31	Плагины для WordPress	Сервис-плагины для CMS WordPress	0/16
32	Платёжная система	Сервисы для оплаты на сайте	0/1
33	Производительность	Сервисы для увеличения производительности сайта	0/1
34	Прочее	ПО, не вошедшее ни в какую группу	2/31
35	Расширение веб-сервера	Дополнительные сервисы веб-сервера	0/7
36	Рекламная сеть	Рекламные сервисы на сайте	5/4
37	Ретаргетинг	Сервисы для автоматического направления онлайн-рекламы посетителям сайта	0/8
38	Автоматизация маркетинга	Сервисы для автоматической рассылки почтовых отправлений	4/3
39	Утилита для разработчиков	Сервисные утилиты для разработки сайтов	6/2
40	Шаблон для WordPress	CMS-шаблоны для WordPress	0/4
41	Шрифт	Набор дополнительных шрифтов оформления страниц	62/128
42	Электронная коммерция	Сервисы для продажи в онлайн-режиме	0/18
43	Язык программирования	Программный язык разработки сайта	127/120
44	wappalyzer_title_ web-server extensions	Сервер технологии Wappalyzer	7/0
45	Фотогалерея	Сервисы для демонстрации изображений	2/0
Итого			955/1585

Заключение

Анализ показал, что число различных видов ПО за пять лет выросло почти на 66 процентов одновременно и с ростом их групп на 68 процентов (25 и 42). При этом для разработки сайтов в большинстве случаев применяют бесплатные CMS, например Joomla и WordPress. Хотя проявилась применимость коммерческих CMS в виде 1C-Bitrix, интегрированной с несколькими широко известными мощными системами управления БД (СУБД), однако, как показал мониторинг, НИИ не использует данное преимущество при размещении своей информации на сайте, что, наряду с отсутствием требований (рекомендаций) со стороны Минцифры и Минобрнауки по выбору ПО, повлекших огромное разнообразие его, лишает возможности автоматически получать информацию с сайта для использования в других информационных системах (ИС), в том числе, и в ИС самой Минобрнауки и отдаляет возможность в перспективе формирования единой цифровой платформы информационных научно-образовательных ресурсов [12].

Литература

1. *Меденников В. И.* Математическая модель формирования цифровых платформ управления экономикой страны / В. И. Меденников // Цифровая экономика. – 2019. – №1(5). – С. 25–35.

2. *Зацаринный А. А.* Цифровая платформа для научных исследований / А. А. Зацаринный // Материалы Международной научной конференции «Математическое моделирование и информационные технологии в инженерных и бизнес-приложениях». – Издательский дом ВГУ, Воронеж. – 2018. – С. 104–113.

3. *Меденников В. И.* Комплементарность функций науки - необходимое условие эффективности развития страны / В. И. Меденников // Информатизация образования и науки. – 2022. – № 4(56). – С. 70–82.

4. Приказ Министерства образования и науки РФ от 14 июня 2013 г. N 462 «Об утверждении Порядка проведения самообследования образовательной организацией». – Москва. URL: https://base.garant.ru/70405358/ (дата обращения: 30.09.2025).

5. *Зацаринный А. А.* Интеграция приложений искусственного интеллекта в единую цифровую платформу АПК / А. А. Зацаринный, В. И. Меденников, А. Н. Райков // Информационное общество. – 2023. – № 1. – С. 127-138.

6. *Milgrom P.* The Economics of Modern Manufacturing: Technology, Strategy and Organization / P. Milgrom // American Economic Review. – 1990. – Vol. 80. – № 3. – P. 511–528.

7. *Алексеева Н. А.* Экономические и управленческие проблемы землеустройства и землепользования в регионе / Н. А. Алексеева, А. К. Осипов, В. И. Меденников [и др.]. – Ижевск : Общество с ограниченной ответственностью «Издательство «Шелест». – 2022. – 225 с.

8. *Брукс Ф.* Мифический человеко-месяц или как создаются программные системы / Ф. Брукс // – СПб.: Символ-Плюс. – 2006. – 304 с.

9. *Боровков А. И.* Цифровые двойники и цифровая трансформация предприятий ОПК / А. И. Боровков, Ю. А. Рябов, К. В. Кукушкин, В. М. Марусева [и др.]. // Оборонная техника. – 2018. – № 1. – С. 6–23.

10. *Меденников В. И.* Необходимость формирования единого цифрового двойника сельскохозяйственного предприятия / В. И. Меденников // Землеустройство, экономика и управление в агропромышленном комплексе в период глобальных вызовов : материалы V Всерос. научно-практической конф. (Ижевск, 01 марта 2023 г.) – Ижевск. – 2023. – С. 236–243.

11. *Меденников В. И.* Эффективность использования информационных интернет-ресурсов научно-исследовательских учреждений аграрного направления / В. И. Меденников, Л. Г. Муратова, С. Г. Сальников. – М. : Аналитик, 2018. – 235 с.

12. *Ерешко Ф. И.* Концепция формирования единого информационного интернет-пространства научно-образовательных ресурсов страны / Ф. И. Ерешко, В. И. Меденников, Ю. А. Флеров // Управление развитием крупномасштабных систем mlsd'2020 : Труды тринадцатой международной конференции. – Москва: Институт проблем управления им. В. А. Трапезникова РАН. – 2020. – С. 376–385.

УДК 519.7

О ЛИНЕЙНОЙ И 4-АДИЧЕСКОЙ СЛОЖНОСТИ ЧЕТВЕРТИЧНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ С ПЕРИОДОМ, РАВНЫМ ПРОИЗВЕДЕНИЮ ДВУХ ПРОСТЫХ ЧИСЕЛ

Р. А. Русаков, Ц. Цао, В. А. Едемский

Новгородский государственный университет им. Ярослава Мудрого

Аннотация. Исследована сложность четвертичных последовательностей с периодом, равным произведению двух простых чисел. Определена их линейная сложность над кольцом классов вычетов по модулю четыре, а также над конечным полем четвёртого порядка, оценена 4-адическая сложность рассматриваемых последовательностей. Метод исследования основан на применении основе обобщённых циклотомических классов и чисел для составного модуля.

Ключевые слова: четвертичные последовательности, линейная сложность, 4-адическая сложность, кольца вычетов, конечные поля, циклотомические классы.

Введение

В силу простоты их реализации бинарные и четвертичные последовательности представляют собой важный класс последовательностей, значимый для практических применений. Автокорреляция, линейная и 4-адическая сложности являются важными характеристиками непредсказуемости четвертичных последовательностей.

В [1] были предложены правила конструирования новых многофазных последовательностей с периодом pq, где $p>2$ и $q>2$ — различные простые числа. Было показано, что эти последовательности обладают хорошими автокорреляционными свойствами. Также в [1] изучена их линейная сложность над простыми полями, но линейная сложность четвертичных последовательностей над кольцом классов вычетов по модулю 4 не исследована.

В этой статье определяем линейную сложность этих последовательностей как над кольцом классов вычетов по модулю 4, так и над полем четвёртого порядка, а также оцениваем их 4-сложность, развивая результаты, полученные ранее в [5].

1. Определение последовательности

В этом разделе напомним определение четвертичных последовательностей, рассмотренных ранее в [1], а также определим их, используя обобщенные циклотомические классы.

Пусть p, q — различные простые числа такие, что $p \equiv q \equiv 1 \pmod 4$. Обозначим через ζ и η первообразные корни по модулям p и q соответственно. Тогда циклотомические классы четвёртого порядка по модулям p и q определяются как

$$H_i = \{\zeta^{i+4t} \bmod p, t = 0,1,\ldots,(p-1)/4-1\} \text{ и } G_i = \{\eta^{i+4t} \bmod q, t = 0,1,\ldots,(q-1)/4-1\}.$$

Пусть $P = \{p, 2p, \ldots, (q-1)p\}$ и $Q = \{q, 2q, \ldots, (p-1)q\}$. Рассмотрим четвертичную последовательность (s_i) периода pq, заданную по правилу

$$s_i = \begin{cases} (k+l) \bmod 4, & \text{если } i \bmod p \in H_k \text{ и } i \bmod q \in G_l, \\ 0, & \text{если } i \bmod pq \in P \cup \{0\}, \\ 2, & \text{если } i \bmod pq \in Q. \end{cases} \quad (1)$$

Согласно [1], она обладает хорошими автокорреляционными свойствами.

Для наших целей дадим альтернативное определение. Отображение $f(a) = (a \bmod p, a \bmod q)$ устанавливает изоморфизм колец $\mathbb{Z}_{pq} \cong \mathbb{Z}_p \times \mathbb{Z}_q$, где $\mathbb{Z}_{pq}$ — кольцо классов вычетов по модулю pq. Пусть $g \in \mathbb{Z}_{pq}^*$ — элемент, удовлетворяющий условиям $g \bmod p = \zeta$ и $g \bmod q = \eta^{-1}$. Поскольку η^{-1} является первообразным корнем по модулю q, g служит общим первообразным корнем по модулям p и q порядка $(p-1)(q-1)/e$, где $e = \text{НОД}(p-1, q-1)$. Пусть x удовлетворяет условиям $x \equiv g \pmod p$ и $x \equiv 1 \pmod q$. Определим обобщённые циклотомические классы Уитмена:

$$C_i = \{g^{i+te} \bmod pq \mid t = 0,1,\ldots,(p-1)(q-1)/e-1\}, \quad i = 0,1,\ldots,e-1.$$

Отсюда получаем разбиение:

$$\mathbb{Z}_{pq} = \bigcup_{i=0}^{e-1} C_i \cup P \cup Q \cup \{0\}.$$

Для $j = 0,1,2,3$ определим $D_j = C_j \cup C_{j+4} \cup \cdots \cup C_{e+j-4}$. Согласно определению (1) получаем, что

$$s_i = \begin{cases} j, & \text{если } i \bmod pq \in D_j, \\ 0, & \text{если } i \bmod pq \in P \cup \{0\}, \\ 2, & \text{если } i \bmod pq \in Q. \end{cases} \tag{2}$$

Таким образом, эта последовательность может быть определена на основе обобщённых циклотомических классов. В случае $e = 4$ последовательность (s_i) представляет собой обобщённую циклотомическую последовательность Уитмена. Её свойства, в частности линейная сложность, были исследованы в [2, 3]. Далее рассмотрим общий случай, когда $e \geq 4$.

2. Линейная сложность последовательности над $\mathbb{Z}_4$

Линейной сложностью последовательности (h_n) с периодом T над $\mathbb{Z}_4$ называется наименьший порядок L линейного рекуррентного соотношения, которому удовлетворяют члены последовательности, то есть

$$h_{n+L} = c_{L-1}h_{n+L-1} + \ldots + c_1 h_{n+1} + c_0 h_n \quad \text{для } n \geq 0,$$

где $c_0 \neq 0$, $c_1, \ldots, c_{L-1} \in \mathbb{Z}_4$.

Обозначим через r порядок 2 по модулю pq и рассмотрим кольцо Галуа $\text{GR}(4, 4^r)$. Мультипликативная группа этого кольца содержит циклическую подгруппу порядка $2^r - 1$. Так как pq делит $2^r - 1$, то существует α порядка pq в $\text{GR}(4, 4^r)$. Тогда

$$L = pq - |\{n \mid S(\alpha^n) = 0, n = 0,1,\ldots, pq-1\}|, \tag{3}$$

где $S(x)$ — порождающий многочлен последовательности.

Теорема 1. *Пусть четвертичная последовательность (s_i) определена согласно (2). Тогда*

1. $L = pq - (p-1)(q-1)/4 - q$, если $p \equiv q \equiv 1 \pmod 8$ и $2 \in D_0$,
2. $L = pq - (p-1)(q-1)/4 - p$, если $p \equiv q \equiv 5 \pmod 8$ и $2 \in D_0$,
3. $L = pq - q$, если $p \equiv q \equiv 1 \pmod 8$ и $2 \notin D_0$,
4. $L = pq - p$, если $p \equiv q \equiv 5 \pmod 8$ и $2 \notin D_0$,
5. $L = pq - p - q + 1$, если $p \equiv 1 \pmod 8$ и $q \equiv 5 \pmod 8$,
6. $L = pq - 1$, если $p \equiv 5 \pmod 8$ и $q \equiv 1 \pmod 8$.

Доказательство. Определим $U(x) = \sum_{j=0}^{3} j \sum_{i \in D_j} x^i$. Тогда $S(\alpha^a) = U(\alpha^a) + 2$, если $a \in \mathbb{Z}_{pq}^*$.

Поскольку циклотомические числа Уитмена порядка 2 при $p \equiv q \equiv 5 \pmod 8$ и при $p \equiv q \equiv 1 \pmod 8$ совпадают, можно показать, аналогично [2], что $U(\alpha) \in \mathbb{Z}_4$ тогда и только тогда, когда

$2 \in D_0$. Кроме того, как и в [2], имеем $U(\alpha^a) = U(\alpha) - k$, если $a \in D_k$, $k = 0,1,2,3$. Отсюда следует

$$\left|\{a \in \mathbb{Z}^*_{pq} \mid S(\alpha^a) = 0\}\right| = \begin{cases} (p-1)(q-1)/4, & \text{если } p \equiv q \pmod 8 \text{ и } 2 \in D_0, \\ 0, & \text{в противном случае.} \end{cases}$$

Анализируя далее значения $S(\alpha^a)$, когда $a \in P \cup Q \cup \{0\}$, получаем утверждение теоремы 1 по формуле (3).

3. Линейная сложность последовательности над конечным полем

Данную последовательность также можно рассматривать над конечным полем четвертого порядка. Пусть $\mathbb{F}_4 = \{0,1,\mu,\mu+1\}$ — конечное поле порядка 4, где $\mu^2 = \mu + 1$. Определим отображение Грея:

$$\varphi(0) = [0,0], \quad \varphi(1) = [0,1], \quad \varphi(2) = [1,1], \quad \varphi(3) = [1,0].$$

Полагая $t_i = \varphi(s_i)$, получаем последовательность (t_i) над $\mathbb{F}_4$, которая определяется следующим образом:

$$t_i = \begin{cases} 0, & \text{если } i \bmod pq \in D_0 \cup P \cup 0, \\ 1, & \text{если } i \bmod pq \in D_1, \\ \mu+1, & \text{если } i \bmod pq \in D_2 \cup Q, \\ \mu, & \text{если } i \bmod pq \in D_3. \end{cases} \tag{4}$$

Теорема 2. *Пусть последовательность $\{t_i\}$ определена согласно (4). Тогда*

$$L = \begin{cases} pq - (p-1)(q-1)/4 - q, & \text{если } p \equiv q \pmod 8 \text{ и } 2 \in D_0, \\ pq - q, & \text{в противном случае.} \end{cases}$$

Доказательство. Пусть $T(x) = \sum_{i=0}^{pq-1} t(i)x^i$ и ξ — корень pq-й степени из 1 в расширении поля $\mathbb{F}_4$. Из [4] следует, что

$$\left|\{a \in \mathbb{Z}^*_{pq} \mid T(\xi^a) = 0\}\right| = \begin{cases} (p-1)(q-1)/4, & \text{если } p \equiv q \pmod 8 \text{ è } 2 \in D_0, \\ 0, & \text{в противном случае.} \end{cases}$$

Так как $\sum_{k \in D_j} \xi^{ak} = (p-1)/4$, то $\left|\{a \in P \mid T(\xi^a) = 0\}\right| = q - 1$.

Далее, $T(\xi^a) \neq 0$, когда $a \in Q$ и $T(1) = 0$. Данное утверждение завершает доказательство теоремы 2.

4. 4-адическая сложность последовательности

Наряду с линейной сложностью, 4-адическая сложность является другой важной характеристикой непредсказуемости последовательности. Она определяется как наименьшая длина регистра сдвига с обратной связью по переносу, синтезирующему данную последовательность, и обозначается через $\Phi_4(\mathbf{s})$. Согласно [6], она может быть вычислена по формуле

$$\Phi_4(\mathbf{s}) = \left\lceil \log_4 \left(\frac{4^p q - 1}{\text{НОД}(S(4), 4^{pq} - 1)} \right) \right\rceil,$$

где $S(X) = \sum_{i=0}^{pq-1} s_i X^i \in \mathbb{Z}[x]$ и $\lceil z \rceil$ — наименьшее целое число, большее или равное z.

Теорема 3. *Пусть четвертичная последовательность* (s_i) *определена согласно (2). Тогда*

$$\Phi_4(\mathbf{s}) \geq pq - \log_4(d_1 d_2 d_3),$$

где

$$d_1 = \text{НОД}(4^q - 1, (p-1)/2),$$
$$d_2 = \text{НОД}(4^p - 1, (3q+1)/2),$$
$$d_3 = \text{НОД}(4^{pq} - 1, (1+2pq)(1+6pq)).$$

Доказательство. Так как

$$4^{pq} - 1 = (4^q - 1) \cdot \frac{4^p - 1}{3} \cdot \frac{3(4^{pq} - 1)}{(4^p - 1)(4^q - 1)},$$

то рассмотрим три случая.

а) Пусть $d_1 = \text{НОД}(S(4), 4^q - 1)$. Определим $H_j(x) = \sum_{i \in D_j} x^i$, $j = 0,1,2,3$. Тогда

$$H_j(4) \equiv \frac{p-1}{4} \cdot \left(\frac{4^q - 1}{3} - 1 \right) \pmod{4^q - 1}.$$

Следовательно, $S(4) \equiv (p-1)/2 \pmod{4^q - 1}$ и $d_1 = \text{НОД}(4^q - 1, (p-1)/2)$..

б) Пусть $d_2 = \text{НОД}(S(4), 4^p - 1)$. Применив тот же подход, что и в случае а), получаем, что $d_2 = \text{НОД}(4^p - 1, (3q+1)/2)$.

в) Пусть $d_3 = \text{НОД}\left(S(4), \frac{3(4^{pq} - 1)}{(4^p - 1)(4^q - 1)} \right)$. В этом случае, используя метод из [5], получаем, что d_3 может быть равно только $1 + 2pq$ или $1 + 6pq$. Последнее замечание завершает доказательство теоремы.

Благодарности

Настоящая публикация подготовлена в ходе реализации НИР «Математическое моделирование природных процессов», выполняемой в рамках государственного задания в сфере научной деятельности.

Литература

1. *Green D. H., Green P. R.* Polyphase-related prime sequences // IEE Proceedings – Computers and Digital Techniques. – 2001. – Vol. 148, № 2. – P. 53–62.

2. *Edemskiy V.* The linear complexity and autocorrelation of quaternary Whiteman's sequences // International Journal of Applied Mathematics, Electronics and Computers. – 2013. – Vol. 1, № 4. – P. 7–11.

3. *Chen Z. X.* Linear complexity and trace representation of quaternary sequences over Z_4 based on generalized cyclotomic classes modulo pq // Cryptography and Communications. – 2017. – Vol. 9. – P. 445–458. – DOI: 10.1007/s12095-016-0185-6.

4. *Cesmelioglu A., Meidl W.* A general approach to construction and determination of the linear complexity of sequences based on cosets // Sequences and Their Applications. – 2010. – P. 125–138.

5. *Edemskiy V., Chen Z. X.* On the 4-adic complexity of the two-prime quaternary generator // Journal of Applied Mathematics and Computing. – 2022. – Vol. 68. – P. 3565–3585. – DOI: 10.1007/s12190-200-01740-z.

6. *Goresky M., Klapper A.* Algebraic Shift Register Sequences. – Cambridge : Cambridge University Press, 2012. – 498 p. – DOI: 10.1017/CBO9781139057448.

УДК 004.056

АНАЛИЗ СОВРЕМЕННЫХ ТРЕНДОВ БОРЬБЫ С ИНФОРМАЦИОННЫМ МОШЕННИЧЕСТВОМ НА ПРИМЕРЕ ВИШИНГА

А. В. Токарь, А. Е. Яковлева

МИРЭА – Российский технологический университет

Аннотация. В статье рассматривается актуальная проблема голосового фишинга в контексте растущих киберугроз, обусловленных развитием искусственного интеллекта и цифровых коммуникационных технологий. Проведен анализ механизмов реализации вишинговых атак, оценена масштабность проблемы на основе статистических данных и результатов собственного исследования. Предложены и обоснованы комплексные меры противодействия, включающие технологические, организационные и законодательные подходы. Выявлено, что основу надёжной защиты составляют внимательность пользователей, соблюдение базовых правил безопасности и повышение уровня осведомлённости.
Ключевые слова: телефонное мошенничество, вишинг, социальная инженерия, информационная безопасность, мошенничество, deep-fake технологии, кибербезопасность.

Введение

В условиях развития информационно-коммуникационных технологий и массового внедрения искусственного интеллекта голосовой фишинг (вишинг) становится одной из актуальных угроз информационной безопасности. Если традиционный фишинг осуществляется через электронную почту и веб-интерфейсы, то голосовой фишинг использует доверие, которое люди испокон веков питали к телефонной коммуникации. Расширение доступности VoIP-телефонии, простота подделки номеров вызывающего абонента и развитие технологий синтеза голоса создают условия для проведения персонализированных и масштабных атак. Применение искусственного интеллекта и deepfake-технологий позволяет злоумышленникам создавать убедительные имитации голоса конкретного человека, что существенно повышает эффективность манипуляции и вводит в заблуждение даже подготовленных пользователей. Целью данного исследования является анализ проблемы голосового фишинга: его эволюция, современные методы реализации атак, механизмы воздействия, результаты оригинального исследования осведомленности, а также разработка многоуровневых подходов к противодействию, включающих технологические, организационные и правовые решения.

1. Теоретическая часть

1.1. История вишинга

Вишинг (от англ. «voice phishing» — голосовой фишинг) — это форма телефонного мошенничества, которая появилась и стала активно развиваться в начале 2000-х годов на волне массового распространения интернет-технологий. Вишинг возник в связи с эволюцией классического фишинга — атак, направленных на выманивание конфиденциальной информации путем обмана. Если классический фишинг осуществлялся через электронную почту и веб-интерфейсы, то создатели вишинга решили использовать более древний, но по-прежнему доверительный канал коммуникации — телефонный звонок. Первые вишинговые атаки были зафиксированы на абонентах интернет-провайдеров и банках, когда злоумышленники выдавали себя за сотрудников финансовых учреждений для получения номеров кредитных карт и других платежных реквизитов. С развитием VoIP-технологий и появлением инструментов подделки

номеров (spoofing) вишинг из экзотической атаки превратился в массовый, систематический инструмент киберпреступности, адаптирующийся к каждому новому поколению технологий.

1.2. Методы реализации вишинговых атак

Вишинг реализуется как целенаправленная кампания, обычно включающая три стадии: сбор данных, разработка сценария взаимодействия и непосредственное проведение.

Стадия 1: Сбор данных. Злоумышленник собирает всевозможные сведения о потенциальной жертве: проводит анализ профилей в социальных сетях, изучает утечки баз данных или же незаконными путями приобретает закрытые базы данных и даже производит автоматизированный парсинг открытых источников. Это позволяет составить «портрет» адресата и выбрать наиболее правдоподобную маску.

Стадия 2: Разработка сценария. Мошенник моделирует ситуацию и использует наиболее «вероятные» проблемные моменты, связанные с жертвой. Например, понимая о наличии финансовых проблем, под предлогом предоставления денежных средств или их защиты, мошенник может получить данные о банковских счётах. На следующем этапе подготовленный сценарий прорабатывается с учётом возможных реакций и создаётся набор реплик и контрреплик. Наконец, сам звонок — момент применения психологических триггеров: срочность, страх, авторитет, обращение к эмпатии, которые направлены на снижение критичности суждений и ускорение принятия неверного решения.

Стадия 3: Непосредственное проведение. Сам звонок — это момент применения психологических триггеров: срочность, страх, авторитет, обращение к эмпатии, которые направлены на снижение критичности суждений и ускорение принятия неверного решения.

Популярные сценарии вишинга: «Сотрудник банка» — мошенник отмечает «подозрительную активность операций» и необходимость предотвращения хищения; «Техническая поддержка» — преступник притворяется лицом какой-либо популярной компании; «Социальная помощь» или «Медицинская помощь» — метод, применяемый с целью обмана пожилых людей; «Сотрудник налогового органа» — жертве воспроизводится автоматическое заранее заготовленное сообщение об наличии проблем с налоговой декларацией.

1.3. Современные тренды фишинга

Эволюция методов вишинга неразрывно связана с развитием информационных технологий. Современные атаки характеризуются комбинированием традиционных методов социальной инженерии с передовыми инструментами, которые значительно повышают их эффективность.

Подделка номеров (Caller ID Spoofing) представляет собой одну из базовых технологий, без которой вишинг неэффективен. Мошенники используют международную VoIP-маршрутизацию, анонимные шлюзы и специализированные технологии подмены номеров, позволяющие отправителю выдать свой номер за номер доверенной организации или конкретного человека. Эти технологии препятствуют трассировке источника вызова, что затрудняет расследование и превращает вишинговые кампании в практически безнаказанное преступление. Злоумышленник может звонить от имени банка, государственного органа или даже близкого человека жертвы, создав полную иллюзию законного и доверенного контакта.

Синтез голоса на основе искусственного интеллекта и технологии Deepfake — это новое поколение инструментов, кардинально изменивших вишинг. Использование систем машинного обучения и нейронных сетей позволяет злоумышленникам с высокой точностью воспроизводить индивидуальные характеристики голоса конкретного человека: интонации, тембр, речевые особенности, акценты и даже эмоциональный окрас. Искусственно сгенерированный голос может представить руководителя, коллегу, работника доверенной организации, или

даже близкого родственника жертвы. Это добавляет психологический элемент манипуляции, поскольку жертва слышит знакомый голос, что значительно снижает её сомнения и критичность восприятия информации. Технологии Deepfake-вишинга делают атаку максимально персонализированной и потому максимально опасной — жертва не может полагаться ни на признание голоса, ни на контекст разговора.

Автоматизированные IVR-системы (Interactive Voice Response) используются мошенниками для проведения массовых кампаний атак. Вместо того, чтобы каждый раз имитировать реального человека, киберпреступники развертывают системы, которые в автоматическом режиме проводят телефонные разговоры с сотнями тысяч потенциальных жертв. Эти системы программируются на реагирование на определённые ответы жертвы, могут переводить вызовы на «специалистов» (живых людей или ботов), и собирать конфиденциальную информацию в полностью автоматизированном режиме. IVR-системы позволяют масштабировать атаки, значительно снижая затраты на их проведение и повышая охват целевой аудитории

2. Исследовательская часть

Для комплексной оценки состояния проблемы вишинга и степени подготовленности к ней будущего поколения специалистов нами было проведено целевое исследование среди студентов МИРЭА – Российского технологического университета, обучающихся на 2-м курсе института кибербезопасности по специальности «Информационная безопасность». В исследовании принялb участие 30 студентов. Выбор этой аудитории определялся следующими соображениями: во-первых, студенты данного направления подготовки представляют собой категорию лиц, которые в ближайшем будущем будут нести ответственность за формирование и реализацию политик информационной безопасности в организациях различных уровней; во-вторых, анализ их компетенций позволяет оценить, насколько содержание образовательных программ соответствует реальным вызовам информационной безопасности, в частности угрозам вишинга; в-третьих, данный контингент служит индикатором для оценки уровня цифровой грамотности и осведомленности среди образованной части населения. Исследование проводилось в форме структурированного анкетирования, включавшего вопросы о практическом опыте столкновения с угрозами, владении современными знаниями о методах атак и оценке адекватности существующих правовых и технических механизмов защиты. Результаты проведённого анкетированиями показаны на рис.1. Подавляющее большинство опрошенных демонстрируют готовность эффективно противостоять вишингу: 93 % корректно реагируют на неожиданные звонки (например, кладут трубку, перезванивают в банк, проверяют информацию). Это указывает на сформированные базовые навыки защиты. Однако 83 % относятся к ним нейтрально, что свидетельствует о недостаточной систематизации знаний и автоматизации защитного поведения. При попытке социальной инженерии в виде звонка от «банка» 93 % студентов готовы отказать в информации или перепроверить звонок, что является позитивным результатом. Это говорит о критическом мышлении и осознании угрозы финансовых мошенничеств. Методы защиты применяются более чем половиной опрошенных: 50 % студентов используют различные способы защиты от вишинга, 37 % относятся к этому нейтрально, а 13 % демонстрируют отрицательное отношение или игнорируют угрозу. Понимание самого явления распределено относительно равномерно: 33 % хорошо знакомы с концепцией вишинга, 43 % впервые слышат об этом, 23 % имеют поверхностное представление. Это указывает на необходимость единого внедрения стандартизированных образовательных материалов по кибербезопасности.

Тревожным факто стало то, что 67 % опрошенных (или их близких) уже сталкивались с телефонным мошенничеством. При этом 33 % получают подозрительные звонки на регулярной или периодической основе, а 67 % воздерживаются от чёткого ответа, что указывает на

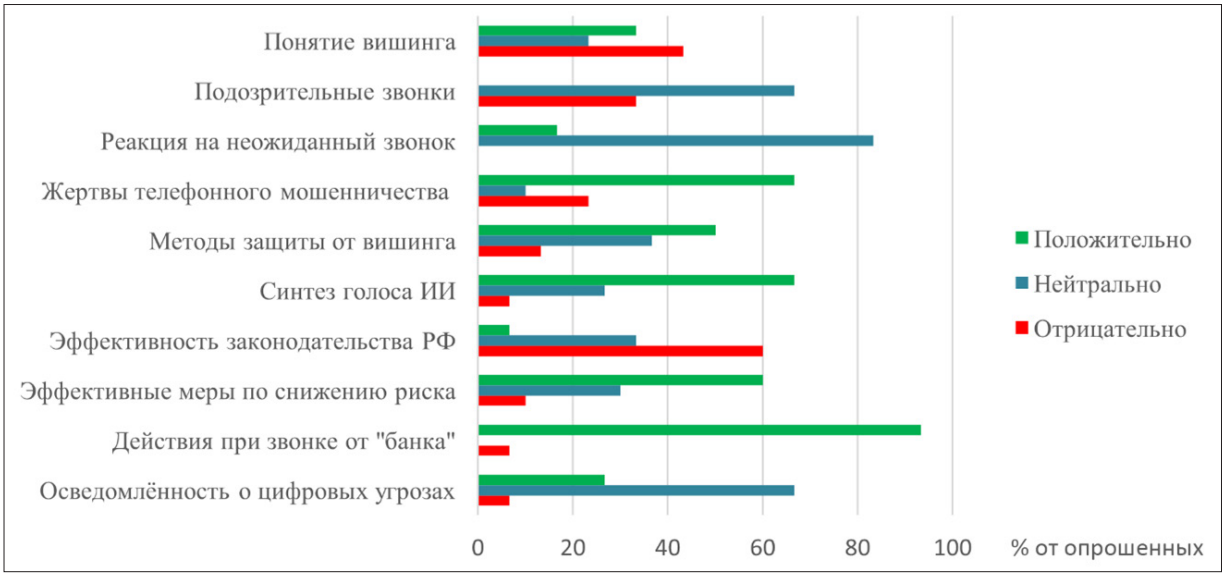

Рис. 1. Исследование степени подготовленности

привыкание к угрозе и её нормализацию среди молодёжи. 67 % студентов осведомлены о технологиях синтеза голоса на базе ИИ (deepfake), что значительно выше, чем базовое понимание вишинга. Это показывает интерес к теме, однако 27 % слышали об этом поверхностно, а 7 % не знакомы с растущей угрозой дипфейков. Только 7 % респондентов верят в эффективность законодательства РФ в борьбе с телефонным мошенничеством. 60 % считают его недостаточным, а 33 % затрудняются с оценкой. Это критический результат, отражающий недоверие молодого поколения к существующим механизмам защиты и необходимость совершенствования нормативно-правовой базы. При оценке потенциальных мер для снижения риска вишинга 60 % студентов видят перспективу в предложенных решениях, 30 % остаются нейтральны, а 10 % скептичны. Это указывает на готовность к инновационным подходам в борьбе с мошенничеством. Самоанализ студентов показывает скромность в оценке своих знаний: 27 % считают себя высоко осведомлёнными, 67% оценивают свой уровень как средний, 7 % признают недостаток знаний. Расхождение между практическим поведением (высокие показатели защиты) и субъективной оценкой указывает на необходимость повышения уверенности и дополнительного образования. Проведённое исследование подчёркивает важность совершенствования образовательных программ, укрепления законодательства и повышения осведомлённости для эффективного противостояния вишингу.

3. Противодействие вишингу

Современные стратегии борьбы с вишингом предполагают использование как технических, так и организационных мер, а их сочетание позволяет значительно повысить общий уровень защищённости. Рассматривать эти меры отдельно недостаточно: только интеграция современных технологий с систематическим повышением информированности пользователей способна существенно снизить риск успешных мошеннических атак. Основные технические и организационные меры подробно представлены в табл. 1 и табл. 2.

Таблица 1

Технические меры защиты

Мера	Описание
Фильтрация вызовов	Современные системы мониторинга используют алгоритмы машинного обучения для выявления подозрительных звонков, отличающихся по шаблонам поведения от легитимного трафика. Они могут блокировать звонки с подозрительных номеров или нетипичными маршрутам
Проверка идентификаторов (STIR/SHAKEN)	Проверка идентификационных данных вызова и стандарты STIR/SHAKEN способствуют снижению феномена Caller ID spoofing — подделки номера вызывающего абонента. Этот протокол предоставляет криптографическую верификацию источника звонка, помогая операторам телефонии фильтровать фальшивые вызовы. Его внедрение в ряде стран уже показало положительные результаты в снижении количества мошеннических звонков.
Блокировка мошеннических шлюзов	Позволяют систематически исключать из сети известные источники фейковых звонков. Для этого операторы и специализированные службы ведут и постоянно обновляют базы таких данных, обеспечивая своевременное реагирование.
Ограничение голосовых подтверждений	Введение двуканальной аутентификации — например, требование подтверждения через SMS или отдельное приложение — сокращает риски компрометации из-за вишинга.

Таблица 2

Организационные меры защиты

Мера	Описание
Минимизация привилегий сотрудников	Выдача только необходимых для выполнения рабочих функций прав снижает вероятность компрометации через инсайдерские открытия или в случае успешной социальной инженерии.
Сценарные учения и тренировки	Роль имитационных тренировок — моделировать реальные ситуации и отработать правильное поведение при подозрительных звонках.
Просветительская работа	Работа, направленная на разную аудиторию, от пожилых людей до специалистов финансового сектора, значительно повышает общую информационную осведомленность, что помогает избегать распространённых ловушек мошенников.
Фиксация подозрительных вызовов	Обеспечивает сбор доказательной базы, необходимой для последующего анализа и взаимодействия с правоохранительными органами.

Заключение

Таким образом, одним из ключевых направлений дальнейшей борьбы с вишингом является совершенствование механизмов обмена информацией между операторами связи, финансовыми организациями и правоохранительными органами, а также развитие международных стандартов и практик, что позволит ускорить блокировку и расследование инцидентов. Не-

смотря на сохраняющуюся угрозу, защита от вишинга достижима. Основу надёжной защиты составляют внимательность пользователей, соблюдение базовых правил безопасности и повышение уровня осведомлённости.

Литература

1. *Пекарева В. В.* Вишинг и смишинг как нескончаемые угрозы информационной безопасности / В. В. Пекарева // Скиф. Вопросы студенческой науки. – 2024. – № 1(89). – С. 80–83. – EDN LEZYXD.

2. *Шинкарук И. П.* Вишинг как одна из распространенных форм мошеничества в современном мире / И. П. Шинкарук // За нами будущее: взгляд молодых ученых на инновационное развитие общества: Сборник научных статей 3-й Всероссийской молодежной научной конференции, Курск, 03 июня 2022 года. Том 2. – Курск : Юго-Западный государственный университет, 2022. – С. 285–289. – EDN UKVEWT.

3. *Куренная В. О.* Фишинг и вишинг в информационной безопасности // Научно-образовательный журнал для студентов и преподавателей «StudNet». – 2022. – № 6. – С. 7214–7218.

4. *Ермакова А. Л., Чаплыгина В. Н.* Фишинг как распространенное киберпреступление современности // Закон и право. – 2022. – С. 149–151.

УДК 004.056

ГИБРИДНАЯ МОДЕЛЬ ОБНАРУЖЕНИЯ ЦЕЛЕВЫХ АТАК НА ОБЪЕКТЫ КРИТИЧЕСКОЙ ИНФОРМАЦИОННОЙ ИНФРАСТРУКТУРЫ

П. Е. Фатьянов[1], А. В. Душкин[1,2]

[1]Национальный исследовательский университет «Московский институт электронной техники»
[2]МИРЭА – Российский технологический университет

Аннотация. В статье рассматривается актуальная проблема обеспечения безопасности критической информационной инфраструктуры от целевых компьютерных атак. Проанализированы недостатки существующих сигнатурных и статистических методов обнаружения. Предложена гибридная модель, интегрирующая анализ сетевых аномалий на основе машинного обучения и поведенческого анализа процессов на уровне узлов. Модель направлена на выявление слабоинтенсивных и замаскированных атак, характерных для Advanced Persistent Threat. Приводится описание архитектуры модели. Эффективность модели подтверждена результатами эксперимента с использованием синтетического набора данных, имитирующего работу объекта критической информационной инфраструктуры. Показана возможность достижения снижения уровня ложных срабатываний при сохранении высокой точности обнаружения.
Ключевые слова: алгоритм классификации, гибридная модель, информационная безопасность, кибербезопасность, критическая информационная инфраструктура, машинное обучение, обнаружение вторжений, поведенческий анализ, сетевая аномалия, целевая атака.

Введение

Критическая информационная инфраструктура (КИИ) представляет собой совокупность информационных и телекоммуникационных систем, от которых зависит функционирование государства, экономики и социальной сферы. Защита КИИ от киберугроз является одной из приоритетных задач национальной безопасности большинства стран мира. Особую опасность представляют целевые компьютерные атаки, или Advanced Persistent Threat (APT), которые характеризуются длительным периодом разведки, тщательным планированием и использованием нулевых уязвимостей для достижения конкретных целей противника.

Традиционные подходы к обнаружению вторжений основаны на сигнатурном анализе, который позволяет выявить известные типы атак через сравнение сетевого трафика с базой сигнатур. Однако такой подход неэффективен при столкновении с новыми или видоизменёнными типами атак. Статистические методы позволяют обнаруживать аномалии на основе моделей нормального поведения системы, но часто страдают от высокого уровня ложных положительных срабатываний.

Целью настоящей работы является разработка гибридной модели, которая интегрирует преимущества обоих подходов, обеспечивая эффективное обнаружение целевых атак, включая слабоинтенсивные и замаскированные угрозы, при минимизации ложных положительных срабатываний.

1. Анализ существующих методов обнаружения

1.1. Сигнатурные методы

Сигнатурный анализ является наиболее распространённым методом в средствах защиты, таких как межсетевые экраны и системы обнаружения вторжений (IDS). Основной принцип

заключается в сравнении поступающих данных с предварительно определённой базой известных сигнатур атак.

Основные недостатки сигнатурного подхода: невозможность выявления новых, ранее неизвестных атак (zero-day exploits); необходимость постоянного обновления базы сигнатур; ограниченность при обнаружении слабоинтенсивных атак, маскирующихся под легитимный трафик; высокие вычислительные затраты при работе с большими объёмами данных.

1.2. Статистические и аномалийные методы

Методы обнаружения аномалий основаны на построении профиля нормального поведения системы и выявлении отклонений от этого профиля. Такие методы включают использование статистических распределений, машинного обучения (в частности, методов кластеризации и классификации) и нейронных сетей.

Преимущества: способность выявлять неизвестные типы атак; адаптивность к изменениям в поведении системы.

Недостатки: высокий уровень ложных положительных срабатываний (false positives); сложность интерпретации результатов; необходимость больших объёмов качественных тренировочных данных; уязвимость к атакам типа «отравление данных» (data poisoning).

1.3. Обзор гибридных подходов

В последние годы появляются работы, предлагающие комбинированные подходы к обнаружению вторжений, объединяющие несколько методов [1–6]. Однако большинство существующих решений либо полагаются на ad-hoc комбинацию методов без научного обоснования, либо требуют большого количества вычислительных ресурсов, что делает их непрактичными для развёртывания в реальных системах КИИ.

2. Предлагаемая гибридная модель

2.1. Архитектура модели

Предлагаемая гибридная модель состоит из четырёх основных компонентов:

1) *модуль сетевого анализа* (анализирует сетевой трафик для выявления аномалий на уровне потоков данных с использованием методов машинного обучения (алгоритмы Isolation Forest и Local Outlier Factor));

2) *модуль анализа поведения процессов* (отслеживает поведение процессов на уровне операционной системы (создание процессов, обращение к файлам, сетевые соединения) и выявляет отклонения от нормального поведения на основе поведенческих сигнатур);

3) *модуль корреляции событий* (собирает события от первых двух модулей и коррелирует их для выявления координированных атак, состоящих из нескольких этапов);

4) *система принятия решений* (на основе взвешенного анализа поступивших событий от трёх предыдущих модулей выносит окончательное решение о наличии атаки и её классификации).

2.2. Математическое описание модели

Пусть множество признаков сетевого трафика представлено вектором $x = (x_1, x_2, ..., x_n)$, где x_i — исходящий от объекта КИИ трафик в момент времени t_i.

Для выявления аномалий сетевого трафика используется оценка аномальности на основе Isolation Forest:

$$A_{net}(t) = P(x_t - \text{аномалия} \mid X_{train}),$$

где X_{train} — тренировочное множество трафика, собранное в период нормального функционирования системы.

Поведенческий анализ процессов описывается как:

$$A_{proc}(t) = \sum_{i=1}^{m} w_i \cdot f_i(p_t),$$

где p_t — множество активных процессов в момент времени t, f_i — функции оценки соответствия поведения i-му аспекту нормального поведения, w_i — веса признаков, определённые экспертным путём или оптимизированные через обучение.

Модуль корреляции событий вычисляет интегральный показатель угрозы:

$$R(t) = \alpha \cdot A_{net}(t) + \beta \cdot A_{proc}(t) + \gamma \cdot C(t, \tau),$$

где $C(t, \tau)$ — функция корреляции между событиями сетевого уровня и уровня процессов в временном окне τ, а α, β, γ — параметры, определяющие вклад каждого компонента в окончательное решение.

Решение о наличии атаки принимается путём сравнения $R(t)$ с адаптивным порогом:

$$\text{Атака обнаружена}, \quad \text{если } R(t) > \theta(t),$$

где $\theta(t)$ — адаптивный порог, который динамически обновляется на основе истории нормального поведения системы.

3. Результаты моделирования

Модель была протестирована на синтетическом наборе данных, имитирующем работу системы КИИ. Датасет включал:

1) 50 часов записей нормального функционирования системы;

2) 30 смоделированных сценариев целевых атак, включая APT-подобные инциденты с различной интенсивностью.

Результаты моделирования приведены на (рис. 1, 2) и в табл. 1.

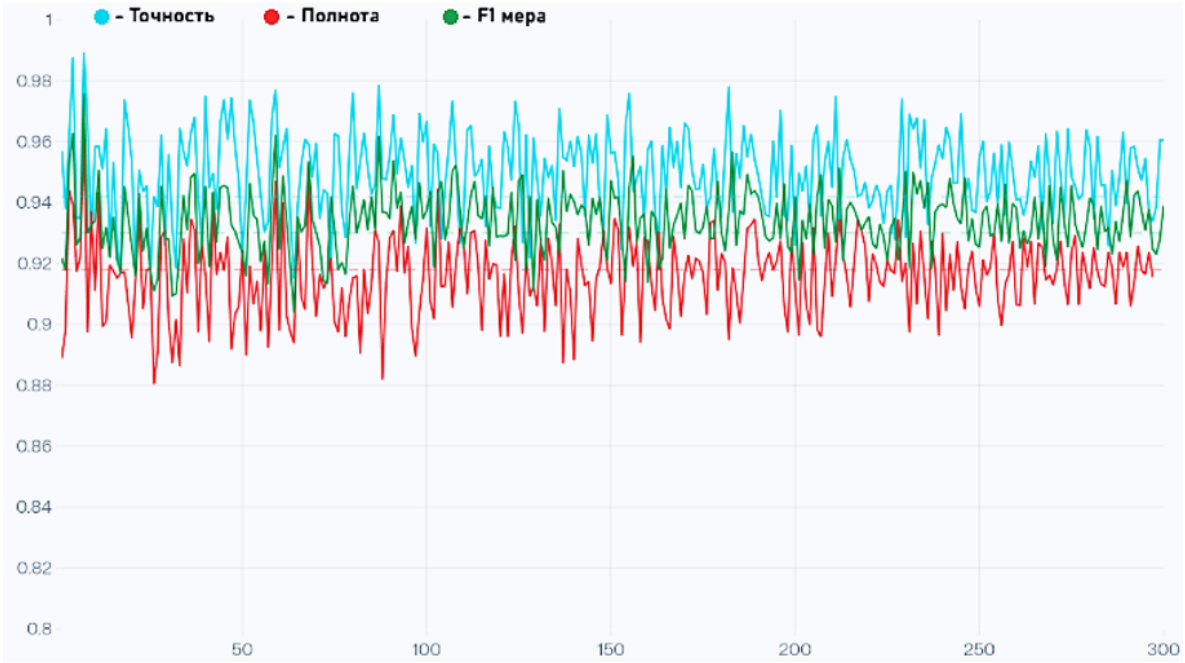

Рис. 1. Показатели точности, полноты и F1-меры в результате моделирования

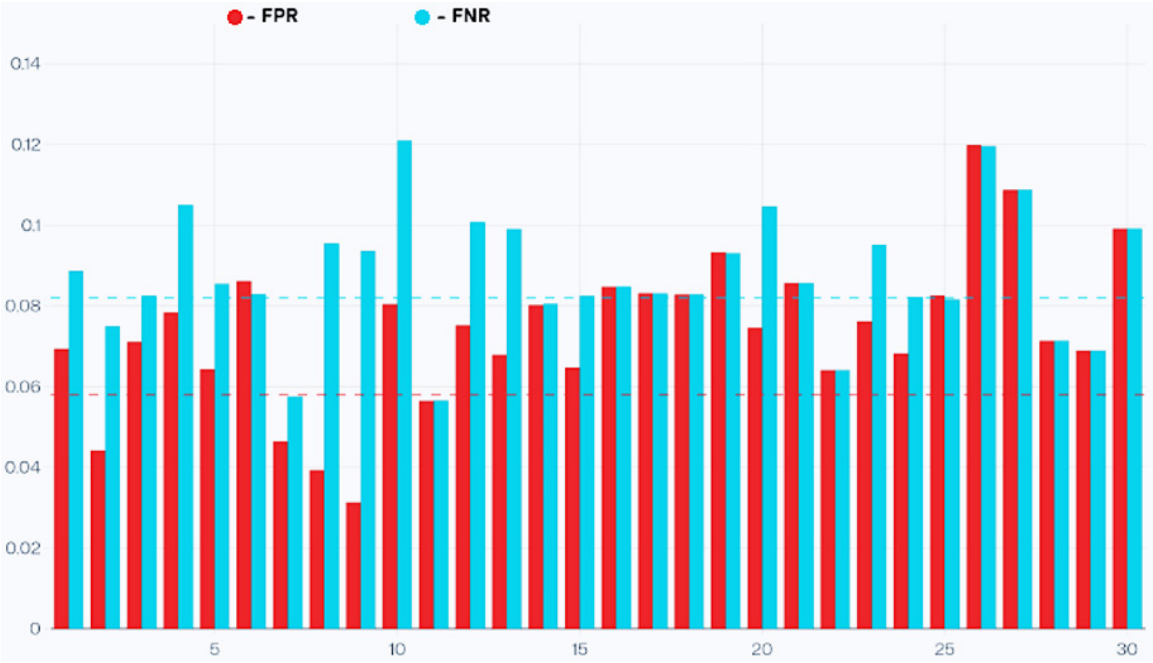

Рис. 2. Показатели FPR, FNR в результате моделирования

Таблица 1

Результаты моделирования

Метрика	Значение
Точность	94,2 %
Полнота	91,8 %
F1-мера	93,0 %
Частота ложноположительной ошибки (FPR): Система отправляет «ДА» (положительный результат), когда на самом деле «НЕТ» (отрицательный)	5,8 %
Частота ложноотрицательной ошибки (FNR): Система отправляет «НЕТ» (отрицательный результат), когда на самом деле «ДА» (положительный)	8,2 %

Сравнение с базовыми методами:

1) сигнатурный анализ: обнаружено 73 % атак (высокая специфичность, низкая чувствительность);

2) метод Isolation Forest: обнаружено 82 % атак (FPR = 15 %);

3) предложенная гибридная модель: обнаружено 91,8 % атак (FPR = 5,8 %).

Полученные результаты свидетельствуют о значительном улучшении показателей обнаружения при одновременном снижении уровня ложных положительных срабатываний.

Заключение

В работе предложена гибридная модель для обнаружения целевых атак на объекты критической информационной инфраструктуры. Модель интегрирует анализ сетевых аномалий на основе машинного обучения и поведенческого анализа процессов [7–12], позволяя выявлять слабоинтенсивные и замаскированные атаки, характерные для Advanced Persistent Threat. Предложенная гибридная модель демонстрирует улучшенные показатели по сравнению с традиционными подходами благодаря следующим факторам: 1) комплексному анализу (использование информации как с сетевого уровня, так и с уровня операционной системы позволяет выявить скоординированные атаки, которые на каждом отдельном уровне могут выглядеть

как нормальная деятельность); 2) снижению ложных срабатываний (модуль корреляции событий позволяет отфильтровать случайные аномалии, которые не являются результатом злонамеренной деятельности); 3) адаптивности (динамическое обновление порога срабатывания позволяет модели адаптироваться к изменениям в окружающей среде и сезонным колебаниям в работе КИИ). Однако, следует отметить некоторые ограничения текущей работы. Модель была протестирована только на синтетических данных, что может не полностью отражать сложность реальных сценариев атак. Кроме того, процедура оптимизации параметров α, β, γ требует дополнительного исследования для обеспечения оптимальной работы в различных конфигурациях КИИ. Результаты моделирования подтверждают эффективность предложенного подхода: гибридная модель обнаруживает 91,8 % целевых атак при уровне ложных положительных срабатываний 5,8 %, что является значительным улучшением по сравнению с существующими методами [1–4].

Дальнейшие направления исследований включают: 1) тестирование модели на реальных данных из операционных систем КИИ; 2) оптимизацию параметров модели для различных типов КИИ (энергетика, транспорт, связь и т.д.); 3) интеграцию методов глубокого обучения для выявления новых типов атак; 4) разработку распределённой архитектуры системы для масштабирования на крупные сети КИИ.

Литература

1. *Безбородов М. А.* Методология анализа информационной безопасности критической инфраструктуры // Информационная безопасность регионов. – 2023. – № 4. – С. 45–58.

2. *Кулешов А. А.* Модели и методы оценки защиты компьютерных сетей / А. А. Кулешов, В. В. Кульба // Автоматика и телемеханика. – 2022. – № 11. – С. 134–151.

3. *Лосев В. С.* Анализ целевых компьютерных атак на информационные системы государственных структур / В.С. Лосев, В.И. Шумов // Проблемы информационной безопасности. – 2023. – № 2. – С. 78–92.

4. *Кочедыков С. С.* Моделирование системы конфликтных взаимодействий в информационной системе критического применения / С. С. Кочедыков, А. В. Душкин, С. П. Соколовский // Вестник Воронежского института ФСИН России. – 2017. – № 4. – С. 74–84.

5. *Душкин А. В.* Способ распознавания вредоносных воздействий на информационную телекоммуникационную систему / А. В. Душкин, В. Н. Похващев, С. П. Соколовский // Телекоммуникации. – 2011. – № 10. – С. 25–28.

6. *Душкин А. В.* Способ повышения эффективности распознавания несанкционированных воздействий на информационную телекоммуникационную систему специального назначения в условиях ограничения временного и вычислительного ресурса / А. В. Душкин, С. П. Соколовский // Информация и безопасность. – 2010. – № 1. – С. 97–102.

7. *Душкин А. В.* Нейросетевая реализация модуля выявления несанкционированных воздействий на информационную телекоммуникационную систему специального назначения // Информация и безопасность. – 2010. – № 1. – С. 123–126.

8. *Щербаков М. В.* Применение методов машинного обучения для обнаружения аномалий в сетевом трафике / М. В. Щербаков, Д. А. Петросов // Наука и образование: электронное научно-техническое издание. – 2022. – Т. 28, № 6. – С. 234–256.

9. *Antonov A. S.* Hybrid Model for Detecting Advanced Persistent Threats in Critical Infrastructure Networks / A. S. Antonov, S. A. Petrov, N. N. Moiseev // Proceedings of International Conference on Information Security. – 2024. – P. 156–172.

10. *Dunn P. C.* Machine Learning Approaches for Network Anomaly Detection in SCADA Systems / P. C. Dunn, R. J. Williams, S. K. Clarke // IEEE Transactions on Industrial Informatics. – 2023. – V. 19, № 4. – P. 5234–5247.

11. *Garcia S.* A Meta-Learning Approach for Adaptive Behavior in Multi-Agent Systems / S. Garcia, A. Zunino, M. Campo // Knowledge-Based Systems. – 2022. – V. 234. – P. 107556.

12. *Dushkin A. V.* Algorithm for predicting the evolution of series of dynamics of complex systems in solving information problems / A. V. Dushkin, T. I. Kasatkina, V. A. Pavlov, R. R. Shatovkin // International Conference «Applied Mathematics, Computational Science and Mechanics: Current Problems» 18-20.12.2017, Voronezh, Russian Federation. IOP Conf. Series: Journal of Physics: Conf. Series 973 (2018) 012031. – DOI:10.1088/1742-6596/973/1/012035.