Дмитриев Дмитрий Валерьевич,
к.т.н., доцент кафедры “Информатика и Системы Управления”
Нижегородский государственный технический
университет им. Р.Е. Алексеева

Мельников Роман Васильевич, магистрант,
Нижегородский государственный технический
университет им. Р.Е. Алексеева

Вайнбаум Денис Алексеевич, магистрант,
Нижегородский государственный технический
университет им. Р.Е. Алексеева

Исаев Максим Александрович, магистрант,
Нижегородский государственный технический
университет им. Р.Е. Алексеева

АРХИТЕКТУРНАЯ МОДЕЛЬ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ МИКРОСЕРВИСНОЙ SELF-SERVICE ПЛАТФОРМЫ УПРАВЛЕНИЯ ИНСТРУМЕНТАМИ АВТОМАТИЗАЦИИ

Аннотация. В работе рассматривается проблема обеспечения информационной безопасности микросервисных self-service платформ, предназначенных для управления инструментами автоматизации в корпоративной среде. Показано, что традиционные модели разграничения доступа и подходы к безопасности микросервисных систем не в полной мере учитывают специфику динамически расширяемых платформ, в которых пользователь одновременно выступает потребителем и разработчиком функциональности. Предложена архитектурная модель обеспечения информационной безопасности, основанная на детерминированном разграничении доступа и принципах Security as Code. Впервые инструмент автоматизации формализуется как самостоятельный объект безопасности с декларативно описываемыми политиками доступа, применяемыми централизованно на уровне архитектуры системы. Практическая применимость модели подтверждена реализацией прототипа микросервисной платформы.

Ключевые слова: Микросервисная архитектура, информационная безопасность, Security as Code, self-service платформа, разграничение доступа, автоматизация.

Введение

В современных корпоративных и телекоммуникационных информационных системах наблюдается устойчивый рост количества инструментов автоматизации, используемых инженерными и эксплуатационными подразделениями. Такие инструменты применяются для управления сетевыми узлами, администрирования сервисов, анализа состояния инфраструктуры и выполнения регламентных операций. Как правило, данные средства разрабатываются различными командами, используют неоднородные технологические стеки и со временем формируют разрозненный и слабо управляемый набор решений.

Отсутствие централизованного подхода к управлению инструментами автоматизации приводит к усложнению сопровождения, затрудняет контроль доступа и повышает риск несанкционированного выполнения операций. Особенно остро данные проблемы проявляются в условиях self-service эксплуатации, когда инженеры различных подразделений

получают возможность самостоятельно разрабатывать, настраивать и использовать инструменты автоматизации без постоянного участия администраторов.

Существующие системы управления автоматизацией, как правило, реализованы в виде монолитных платформ, ориентированных на фиксированный набор сценариев использования. Такие решения отличаются высокой сложностью, избыточностью функциональности и ограниченной адаптируемостью. При этом используемые в них механизмы информационной безопасности часто носят статический характер и плохо масштабируются в условиях динамически изменяющейся инфраструктуры.

Переход к микросервисной архитектуре позволяет повысить гибкость, модульность и отказоустойчивость системы управления инструментами автоматизации. Однако данное архитектурное решение приводит к усложнению модели доверия, увеличению поверхности атаки и росту требований к централизованному контролю безопасности. В этих условиях классические подходы к разграничению доступа оказываются недостаточными.

Объектом исследования является микросервисная self-service платформа управления инструментами автоматизации в корпоративной среде. Предметом исследования являются архитектурные методы обеспечения информационной безопасности таких платформ при распределённом и динамическом характере доступа пользователей. Целью работы является разработка архитектурной модели обеспечения информационной безопасности микросервисной платформы управления инструментами автоматизации на основе детерминированного разграничения доступа и принципов Security as Code.

Анализ существующих подходов к обеспечению безопасности

В традиционных корпоративных информационных системах для обеспечения безопасности широко применяются модели списков контроля доступа (ACL) и ролевая модель управления доступом (RBAC). Аутентификация пользователей, как правило, осуществляется с использованием токенов доступа, в том числе на основе стандарта JWT, что позволяет проверять подлинность запросов и передавать информацию о правах пользователя между компонентами системы.

Несмотря на эффективность данных подходов в монолитных приложениях, их применение в микросервисных архитектурах сопровождается рядом ограничений. Политики доступа, как правило, являются статическими и тесно связаны с бизнес-логикой отдельных сервисов, что затрудняет их изменение и сопровождение. Управление правами доступа в распределённой среде требует согласованности между большим количеством компонентов, что повышает вероятность ошибок конфигурации.

Для частичного решения данных проблем в микросервисных системах применяется архитектурный паттерн API Gateway, обеспечивающий централизованную аутентификацию и базовую авторизацию. Однако в большинстве случаев API Gateway используется исключительно как технический компонент маршрутизации и не рассматривается как элемент формализованной модели информационной безопасности.

Анализ существующих подходов показывает, что они не учитывают специфику self-service платформ управления инструментами автоматизации, для которых характерны динамическое расширение функциональности, высокая изменяемость компонентов и необходимость строгой предсказуемости принимаемых решений. Это обуславливает необходимость разработки архитектурной модели, в которой безопасность является не вспомогательной функцией, а неотъемлемой частью архитектуры системы.

Архитектурная модель микросервисной платформы

Предлагаемая платформа относится к классу микросервисных self-service систем управления автоматизацией, отличающихся отсутствием жёстко фиксированного функционального состава и возможностью динамического подключения новых инструментов.

В рамках данного класса систем пользователь выступает не только потребителем, но и разработчиком функциональности, что принципиально изменяет требования к архитектуре и модели информационной безопасности.

Архитектура платформы построена на основе микросервисного подхода и включает в себя сервис аутентификации и авторизации, сервис управления инструментами автоматизации, сервис оркестрации и развертывания, а также API Gateway, выполняющий роль единой точки входа. Каждый микросервис развертывается в изолированной среде и взаимодействует с другими компонентами посредством стандартизированных интерфейсов.

API Gateway является ключевым элементом архитектуры с точки зрения безопасности. Через него осуществляется проверка подлинности пользователей, применение политик разграничения доступа, валидация входных параметров и регистрация событий безопасности. Такой подход позволяет централизовать контроль доступа и минимизировать дублирование логики безопасности в отдельных сервисах.

Клиентская часть платформы реализована с использованием подхода микро-фронтенда, при котором интерфейсы отдельных инструментов разрабатываются и развертываются независимо. Это повышает гибкость системы и ускоряет внедрение новых инструментов, однако требует учета дополнительных угроз, связанных с изоляцией пользовательских компонентов и контролем их взаимодействия.

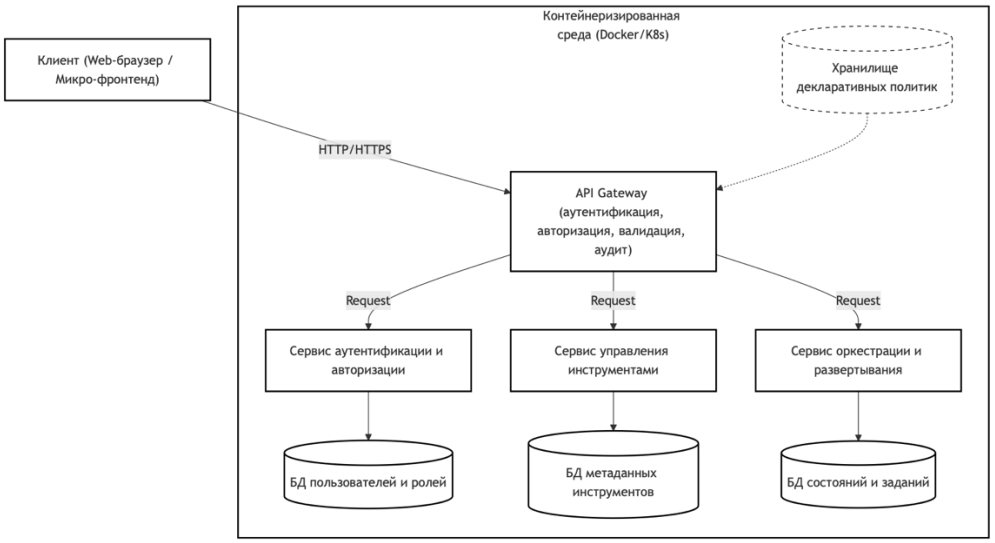

Рисунок 1. Архитектурная схема микросервисной платформы управления инструментами автоматизации.

Детерминированная модель обеспечения информационной безопасности

Под детерминированной моделью обеспечения информационной безопасности в рамках данного исследования понимается формализованная совокупность архитектурных и декларативных правил, при которых результат принятия решения о доступе к ресурсу однозначно определяется заданной конфигурацией политики безопасности и не зависит от статистических, эвристических или поведенческих факторов.

Данная модель ориентирована на обеспечение предсказуемости и воспроизводимости решений, что является критически важным для систем управления инженерной автоматизацией.

Основой предлагаемой модели является концепция Security as Code, предполагающая декларативное описание политик безопасности в конфигурации инструмента автоматизации. Каждому инструменту сопоставляется набор формализованных правил, определяющих допустимые роли пользователей, ограничения на параметры запросов и условия выполнения операций. Политики интерпретируются на уровне API Gateway до передачи запроса во внутренние сервисы системы.

Научная новизна предложенной модели заключается в интеграции детерминированного разграничения доступа с архитектурой микросервисной self-service платформы, в которой политики безопасности ассоциируются не с пользователем или сервисом, а с управляемым инструментом автоматизации. Таким образом, инструмент автоматизации впервые рассматривается как атомарный объект безопасности, обладающий собственным декларативно описываемым набором ограничений доступа.

```
# Пример политики для инструмента "Управление сетевыми ACL"
apiVersion: security.platform.io/v1
kind: ToolPolicy
metadata:
name: network-acl-manager
version: 1.2.0
spec:
description: "Инструмент для добавления/удаления правил в сетевых ACL"
# Кто имеет доступ
allowedRoles:
– "network-admin"
– "security-engineer"
# Какие операции (эндпоинты) доступны
operations:
– path: /api/v1/acl/rules
method: POST
description: "Добавить правило"
# Ограничения на параметры запроса
parameterConstraints:
sourceIp:
required: true
pattern: "^ ((25 [0-5]|2 [0-4] [0-9]| [01]? [0-9] [0-9]?)\\.){3} (25 [0-5]|2 [0-4] [0-9]| [01]? [0-9] [0-9]?)$"
port:
required: true
range: [1, 65535].
– path: /api/v1/acl/rules/{id}
method: DELETE
description: "Удалить правило"
# Только для роли admin
allowedRoles: ["network-admin"].
# Условия выполнения (например, время)
conditions:
timeRestriction:
```

```
enabled: true
allowedDays: ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday"].
timeRange: "08:00-20:00"
```

Листинг 1. Пример декларативной политики безопасности инструмента автоматизации.

Централизованное применение политик безопасности дополняется механизмами аудита, фиксирующими все операции пользователей и изменения конфигураций инструментов. Это обеспечивает трассируемость действий и формирует основу для анализа инцидентов информационной безопасности. Предложенная модель позволяет снизить операционные риски и повысить управляемость системы без потери гибкости.

Обсуждение результатов и перспективы развития

Разработанная архитектурная модель была апробирована в виде прототипа микросервисной платформы, развернутой в контейнеризированной среде. Использование контейнеризации позволило обеспечить изоляцию компонентов и воспроизводимость окружения, а также протестировать различные сценарии взаимодействия сервисов и применения политик безопасности.

Следует отметить, что предложенная детерминированная модель ориентирована на системы с заранее формализуемыми правилами доступа и не предназначена для выявления аномального поведения пользователей. Данное ограничение является осознанным архитектурным выбором, обусловленным требованиями к предсказуемости и управляемости процессов в инженерных системах.

Перспективными направлениями дальнейших исследований являются расширение набора декларативных политик безопасности, интеграция платформы с оркестраторами контейнеров и системами мониторинга, а также автоматизация анализа журналов событий безопасности.

Заключение

В работе предложена архитектурная модель обеспечения информационной безопасности микросервисной self-service платформы управления инструментами автоматизации. Модель основана на детерминированном разграничении доступа и принципах Security as Code, что позволяет рассматривать инструмент автоматизации как самостоятельный объект безопасности с формализованными политиками доступа. Практическая значимость работы заключается в возможности применения предложенного подхода в корпоративных системах, требующих безопасного и управляемого использования инструментов автоматизации.

Список литературы:

1. Богомолов, Р. Д. Современные подходы к построению архитектуры масштабируемых веб-сервисов / Р. Д. Богомолов // Тенденции развития науки и образования. – 2023. – № 97-12. – С. 34-38.

2. Атчисон Л. Масштабирование приложений. Выращивание сложных систем. – СПб.: Питер, 2018. – 256 с.

3. Ким, П. Е. Сравнение инструментов для реализации микрофронтендов (микросервисов) с интеграцией во время сборки SPA веб – приложений / П. Е. Ким, К. Г. Кашапов, Е. А. Голикова // Научно-технические инновации и веб-технологии. – 2023. – № 2. – С. 33-38.

4. Rasheedh, J. A. Reactive Microservices Architecture Using a Framework of Fault Tolerance Mechanisms / J. A. Rasheedh, S. Saradha // Proceedings of the 2nd International Conference on Electronics and Sustainable Communication Systems, ICESC 2021: 2, Coimbatore, 04–06 августа 2021 года. – Coimbatore, 2021. – P. 146-150.