УДК 004.75

ИССЛЕДОВАНИЕ ВЛИЯНИЯ АВТОМАТИЗИРОВАННЫХ ОПЕРАТОРОВ И ИНСТРУМЕНТОВ ХАОС-ИНЖИНИРИНГА НА ОТКАЗОУСТОЙЧИВОСТЬ РАСПРЕДЕЛЕННЫХ, ОБУЧАЮЩИХ, ГЕЙМИФИКАЦИОННЫХ СИСТЕМ

Терехов Н.Г.
(«Университет ИТМО», г. Санкт-Петербург)

Научный руководитель – Болдырева Елена Александровна, кандидат технических наук, Университет ИТМО, доцент (квалификационная категория "ординарный доцент")
(«Университет ИТМО», г. Санкт-Петербург)

Введение. Современные распределенные геймификационные онлайн-системы легко масштабируются горизонтально, но сложны в конфигурации межсервисного взаимодействия. Для автоматизации обновления систем и обеспечения их отказоустойчивости при высоких нагрузках применяются SRE- и DevOps-практики. Однако традиционные механизмы Kubernetes, такие как LivenessProbe, VPA и HPA, обладают статической функциональностью и не всегда эффективно реагируют на перегрузки. В данной работе рассматривается возможность повышения отказоустойчивости таких систем за счет автоматизированных Kubernetes-операторов[1] и хаос-инжиниринга[3].

Основная часть. Цель исследования — повышение отказоустойчивости распределенных обучающих геймификационных систем путем внедрения Kubernetes-операторов и применения инструментов хаос-инжиниринга.

Для достижения этой цели были выполнены следующие задачи:

- Разработан Kubernetes-оператор для существующего веб-приложения а также выбран CRD для внедрения в существующую инфраструктуру.
- Проведено нагрузочное тестирование с симуляцией отказов.
- Оценено влияние оператора на устойчивость системы при высоких нагрузках.

Для проведения симуляции был использован инструмент хаос инжиниринга[2] с использованием как стандартных сценариев сбоя контейнеров[3] (потеря сетевого соединения отдельных подов) так и нагрузочных сценариев (принудительная загрузка процессора, захват оперативной памяти, заполнение диска)[3].

Предложенная архитектура оператора отслеживала и собирала метрики (утечка памяти, сообщения о переполнении очередей с передаваемыми данными и отрицательных ответов эндпоинтов) по основному процессу, рассматриваемого микросервиса, и предусматривала заранее перезапуск подов, переназначение конечных эндпоинтов брокеров сообщений.

Результаты показывают, что использование Kubernetes-операторов позволяет автоматизировать управление ресурсами, динамически адаптируя инфраструктуру к изменяющимся условиям нагрузки.

Однако применение такого рода интрузивного оператора и его влияние на инфраструктуру по косвенным показателям процесса приложения должно фиксироваться с возможностью версионировать состояние инфраструктуры, чтобы внедрение не приводило к несогласованности декларативному объявлению инфраструктуры.

Выводы. В ходе исследования подтверждена эффективность применения Kubernetes-операторов и хаос-инжиниринга для повышения отказоустойчивости геймификационных онлайн-систем. Разработанный оператор позволил повысить отказоустойчивость приложения, обеспечив динамическое управление ресурсами. Нагрузочное тестирование с инструментами хаос-инжиниринга показало, что автоматизированное управление масштабированием снижает риск сбоев, повышает стабильность системы и способствует быстрому восстановлению процессов контейнеров. Однако в долгосрочной перспективе такой подход может вести к потере согласованности декларативного объявления инфраструктуры с фактическим ее состоянием если не отслеживать необходимые изменения, вносимые в период инцидента. Полученные результаты подтверждают эффективность предложенного подхода для надежного функционирования распределенных геймификационных платформ.

Список использованных источников:

1. Трапезин А., Филянин И. Research and development of Kubernetes Operator for Machine Learning pipelines // IEEE 15th Annual Computing and Communication Workshop and Conference. 2025.
2. Akalanka B. M., Thivanka D. M. A Review of Resilience Testing in Microservices Architectures: Implementing Chaos Engineering for Fault Tolerance and System Reliability // IEEE 15th Annual Computing and Communication Workshop and Conference. 2025.
3. Yang T., Lee C., Shen J. An Adaptive Resilience Testing Framework for Microservice Systems // arXiv:2212.12850v1 [cs.SE]. 2022.