

УДК 519.688

Проектирование и разработка сервиса мониторинга для среды общих данных

Треус Иван Сергеевич

Кузьмина Инна Анатольевна (научный руководитель)

МГТУ им. Н.Э. Баумана, Москва, Россия

В условиях стремительного роста облачных технологий возрастает необходимость в эффективных решениях для мониторинга их состояния. Современные облачные системы — приложения, спроектированные для размещения на удаленных масштабируемых серверах, на которых обрабатывают и хранят информацию, включают множество компонентов, работающих в различных средах, что усложняет задачу своевременного обнаружения сбоев и анализа их причин. Чтобы повысить уровень надежности таких систем, требуется интеграция инструмента мониторинга.

Ключевые слова: мониторинг, среда общих данных, наблюдаемость, infrastructure as a code, метрики

Введение. Проектирование — сложный процесс, требующий совместной работы большого количества людей, в связи с чем возникает проблема организации взаимодействия между ними. Одно из возможных решений этой проблемы — использование среды общих данных (СОД) для проектных организаций. СОД представляет собой программно-технический комплекс для совместной работы всех участников проекта с информационными моделями на всех стадиях жизненного цикла проектирования [1].

Как и при работе любой сложной системы, при работе СОД не исключено возникновение внештатных ситуаций. Чтобы повысить уровень надежности системы важно иметь инструменты для их обнаружения и анализа. В качестве этих инструментов может выступать сервис мониторинга, который необходимо спроектировать, разработать и интегрировать в СОД. Сервис мониторинга позволяет оценить состояние системы, отражаемое логами приложений и инфраструктуры. Поэтому эти логи нужно собирать, хранить и анализировать, а результаты анализа необходимо представить пользователю в удобном формате: графики, отражающие работу системы, и уведомления о внештатных ситуациях.

Среда общих данных — масштабируемая система, предназначенная в том числе для размещения в облаке, поэтому при проектировании сервиса мониторинга использовался подход Infrastructure as a Code (IaC) [2]. Он позволяет с помощью кода определить способы взаимодействия модулей сервиса мониторинга друг с другом и средой общих данных, описать размещение модулей на серверах. Использование этого подхода также позволит разворачивать, настраивать и обновлять сервис мониторинга автоматически.

Анализ требований и обзор существующих решений. Стандарты в области разработки сервисов мониторинга продолжают развиваться. Все большее внимание уделяется новой концепции — наблюдаемости. Она необходима, чтобы лучше понимать состояние рабочих нагрузок [3]. Определение ключевых метрик и параметров мониторинга напрямую влияет на выбор технологий, выполняющих каждую из функций. Например, системы, ориентированные на высокочастотные метрики (такие как Prometheus), оптимальны для сценариев, требующих детального анализа производительности в реальном времени, тогда как решения на основе Elasticsearch или Loki лучше подходят для агрегации и поиска по большим объемам логов [4]. Таким образом, требования к наблюдаемости определяют не только набор собираемых данных, но и выбор технологий, обеспечивающих их эффективное хранение, анализ и интерпретацию с учетом масштабируемости.

Исходя из функциональных требований сервиса мониторинга, его можно разделить на три модуля: модуль сбора данных, модуль хранения и обработки данных, модуль визуализации данных. Для каждого из модулей необходимо подобрать технологии, которые будут соответствовать подходу IaC, чтобы их легко можно было интегрировать в процесс автоматического развертывания. Также нужно учитывать совместимость работы выбираемых технологий [5].

Среди множества доступных решений особое место занимает связка Prometheus и Grafana. Она образует полноценную систему мониторинга, охватывающую все ключевые аспекты: от сбора и хранения метрик до их анализа и визуализации. При этом система остается гибкой и масштабируемой, способной адаптироваться к изменениям в инфраструктуре. Важно отметить, что Prometheus не является универсальным решением для всех задач мониторинга.

Стек ELK (Elasticsearch, Logstash, Kibana) также занимает одно из ведущих мест. Этот комплекс инструментов предоставляет полноценное решение для сбора, хранения, обработки и визуализации логов и метрик, позволяя эффективно управлять распределенными системами. Основу стека ELK составляет Elasticsearch — мощная распределенная поисковая и аналитическая система, работающая с документами в формате JSON. Благодаря использованию Apache Lucene, Elasticsearch обеспечивает высокоэффективный полнотекстовый поиск, что делает его идеальным решением для работы с большими массивами неструктурированных данных. Система предлагает горизонтальную масштабируемость через механизмы репликации, а также включает встроенные алгоритмы сжатия данных. Однако при всех своих преимуществах Elasticsearch предъявляет значительные требования к ресурсам, особенно к оперативной памяти.

Для сбора и обработки данных в стеке ELK используется Logstash — универсальный конвейер, способный собирать информацию из самых разнообразных источников. Богатая экосистема плагинов позволяет работать с файлами, базами данных и очередями сообщений, а встроенные фильтры предоставляют возможности для сложных преобразований данных. Тем не менее, при всех своих функциональных возможностях, Logstash отличается

высоким потреблением ресурсов, особенно при обработке больших объемов информации, а его конфигурация для сложных сценариев может оказаться нетривиальной задачей.

Завершающим элементом стека является Kibana — веб-интерфейс для визуализации и анализа данных.

Традиционные решения, такие как Prometheus, несмотря на свою популярность, начинают демонстрировать существенные ограничения при работе с растущими объемами метрик. Эти ограничения становятся особенно заметными при масштабировании инфраструктуры, когда резко возрастает нагрузка на оперативную память и процессор. VictoriaMetrics позиционируется как высокопроизводительная СУБД временных рядов, предлагающая комплексное решение перечисленных проблем. Одним из ее ключевых преимуществ является значительная оптимизация использования ресурсов. По сравнению с Prometheus, эта система демонстрирует в 2–4 раза меньшее потребление оперативной памяти при аналогичных объемах метрик, а эффективные алгоритмы сжатия данных позволяют сократить занимаемое дисковое пространство в 5–10 раз. Оптимизированные методы обработки запросов обеспечивают существенно меньшую нагрузку на процессор, что особенно важно для крупных распределенных систем.

Обзор выбранных технологий. В качестве основы используемого стека была выбрана база данных VictoriaMetrics вместе с Grafana для визуализации. Также необходимо подобрать технологии по автоматизированному управлению инфраструктурой. Выбор пал на простые в настройке инструменты, которые помогают успешно добиться необходимых результатов. Развернуть сервис мониторинга предстоит на серверах трех типов: сервера среды общих данных, сервера баз данных и сервер мониторинга.

Kamal — инструмент для развертывания веб-приложений, разработанный компанией 37signals. Он обеспечивает деплой без простоя, управление вспомогательными сервисами и многое другое, необходимое для управления веб-приложением в продакшене с использованием Docker.

Itamae — инструмент для управления конфигурацией серверов, написанный на Ruby. Он позволяет описывать состояние серверов с помощью простых и понятных рецептов, обеспечивая идемпотентность и повторяемость настроек.

В качестве модуля сбора данных выступают cAdvisor и Grafana Agent, размещаемые на серверах среды общих данных и серверах баз данных. cAdvisor (Container Advisor) — инструмент для автоматического мониторинга контейнеров. Grafana Agent — инструмент для сбора и передачи метрик и логов.

Помимо VictoriaMetrics и Grafana на сервере мониторинга размещаются такие сервисы, как VictoriaLogs, который предназначен для высокопроизводительного хранения и обработки логов, и VMauth, предназначенный для проксирования запросов, поступающих в базу VictoriaMetrics, и Vmalert — для анализа данных, поступающих в базу, и отправки уведомлений в случае

сбоев. Уведомления о сбоях приходят администратору посредством сервиса alert manager, расположенного на том же сервере.

Архитектура программной реализации. Сервис мониторинга является одной из составных частей веб-приложения для администрирования среды общих данных. Приложение для администрирования в контексте использования на серверах заказчика предоставляет пользователю возможность для предварительной настройки развертывания как самой системы, так и ее сервиса мониторинга. Среди настраиваемых параметров сервиса мониторинга — директории хранения данных, есть возможность разместить мониторинг как локально, так и на специальном выделенном сервере. При использовании отдельного сервера потребуется установка TLS-сертификата для настройки доступа к Grafana с помощью протокола HTTPS.

Для доступа к инструментам мониторинга и другим возможностям веб-приложения для администрирования, пользователю необходимо пройти авторизацию. После авторизации при переходе на вкладку мониторинга, пользователю будет доступен просмотр данных в визуализаторе Grafana, встроенном в страницу с помощью HTML тега iframe, позволяющего встраивать другой HTML-документ в текущий.

Поскольку Grafana также требует авторизации, необходимо автоматически авторизовывать в ней пользователя, авторизованного в веб-приложении для администрирования. Иначе повторная авторизация будет портить пользовательский опыт и требовать дополнительных настроек доступа, создание учетных записей для Grafana. Авторизация осуществляется с помощью JSON Web Token (JWT). Токен генерируется сервисом авторизации приложения для администрирования, а Grafana принимает токен в качестве сессии. Токен подписывается и передается по HTTPS для защиты от подмены и изменений во время передачи в случае размещения мониторинга на отдельном сервере. При размещении сервиса мониторинга на локальном компьютере у пользователя, токен передается по внутренней Docker-сети.

Между всеми типами серверов автоматически настраиваются SSH-туннели для безопасной передачи данных в базу данных VictoriaMetrics из источников на серверах баз данных и среды общих данных.

Заключение. В ходе работы был проведен комплексный анализ, проектирование и разработка автоматизированного сервиса мониторинга среды общих данных. Рассмотрены современные технологии мониторинга, обоснован выбор технологического стека и выполнена его интеграция с подходом IaC. Это позволило создать сервис, который обладает высокой гибкостью, масштабируемостью и удобством управления.

Литература

- [1] Пронин В.И., Медведев Д.В. Ингипро. *Человек. Общество. Инклюзия*, 2023, т. 14, № 2.
- [2] Овчинников М.А., Овчинникова Е.В. Инфраструктура, управляемая кодом: методология и инструменты. *Вестник науки и образования Северо-Запада России*, 2015, т. 1, № 3, с. 63–67.

- [3] CNCF Observability Whitepaper. Cloud Native Computing Foundation. URL: <https://github.com/cncf/tag-observability/blob/main/whitepaper.md> (accessed 06.05.2025).
- [4] Альфара А.А., Королев Д.В., Зайцев К.С., Дунаев М.Е. Разработка системы мониторинга для серверного приложения. *International Journal of Open Information Technologies*, 2023, т. 11, № 8, с. 24–31.
- [5] Максимов В.Ю. Преодоление трудностей наблюдаемости в микросервисной архитектуре. *Инновационная наука*, 2024, № 2–1, с. 30–35.

Design and development of a monitoring service for a Common Data Environment

Treus Ivan Sergeevich

Kuzmina Inna Anatolievna (*scientific advisor*)

BMSTU, Moscow, Russia

In the context of the rapid growth of cloud technologies, there is an increasing need for effective solutions to monitor their state. Modern cloud systems — applications designed to run on remote, scalable servers that process and store information — include numerous components operating in different environments, which complicates the timely detection of failures and analysis of their causes. To improve the reliability of such systems, the integration of a monitoring tool is required.

Keywords: monitoring, common data environment, observability, infrastructure as a code, metrics