

# Кластерный мониторинг: отслеживаем работу Kubernetes



Павел СЕЛИВАНОВ,  
Developer Advocate, VK Cloud Solutions

## Встроенный мониторинг vs внешний

Оркестратор Kubernetes предназначен для автоматического управления контейнерами. Мониторинг K8s необходим, чтобы отслеживать состояние всего кластера и проверять работающие подсистемы.

- **Мониторинг кластера** показывает, как работают ноды (узлы сети), какова их производительность, сколько приложений запущено на каждой, как используются ресурсы кластера.
- **Мониторинг подсистем** подсказывает, достаточно ли доступных нод, и смогут ли они справиться с рабочей нагрузкой в случае отказа одной из них.
- **Мониторинг управляющих компонентов Kubernetes** необходим для оценки жизнеспособности кластера и отслеживания возможных проблем.

У Kubernetes есть встроенные функции мониторинга

Kubernetes (K8s) – это самый популярный сегодня контейнерный оркестратор. Объемы микросервисной разработки и масштабы запущенных в контейнерах приложений увеличиваются, и с ними возрастает важность эффективного мониторинга кластеров Kubernetes. О том, как работают системы мониторинга, какие решения имеются на рынке и почему встроенных функций часто оказывается недостаточно, расскажем в предлагаемой статье.

и логирования. С помощью базовых функций администраторы могут просматривать информацию по ресурсам CPU и памяти для каждого пода, а также просматривать логи, связанные с жизненным циклом самого приложения.

Однако система оповещения (алертинг) и более продвинутые возможности доступны только при использовании внешних систем мониторинга и логирования. Например, если инстанс приложения больше не существует из-за перезапуска или создания в другом месте, мониторинг становится недоступным. Такой подход неприемлем для управления масштабной инфраструктурой, связанной с контейнерами.

Если использовать только встроенные средства мониторинга Kubernetes, возникнет дефицит данных о событиях, происходящих с приложениями в контейнерах. Природа любых аварий или сбоев останется загадкой из-за отсутствия информации о причинах инцидента. Неважно, происходит сбой на уровне приложений, непосредственно с Kubernetes, либо проблема кроется в операционной системе.

Без глубокого мониторинга Kubernetes ИТ-системы теряют гибкость. Как только сложность проекта в кластере начнет расти, возникнет разрыв между темпами масштабирования решения и эффективностью мониторинга.

Без отслеживания метрик, алертинга и логирования событий

невозможно обеспечить прозрачность работы кластеров в рамках масштабных целостных инфраструктур. Вот поэтому, собственно, и необходимы внешние инструменты мониторинга.

## И пришел Prometheus

Kubernetes изначально ориентирован на поддержку мониторинга внешними решениями. Среда K8s устроена и работает по принципу Infrastructure-as-Code: в ней все постоянно меняется, перестраивается и перенастраивается, запускаются и перезапускаются новые кластеры и поды. На момент появления Kubernetes существующие системы (Zabbix, Nagios и пр.) не умели работать с подобными динамическими инфраструктурами.

Для мониторинга кластеров Kubernetes хорошо подходит Prometheus. Сейчас этот Open Source-инструмент считается стандартом в сфере мониторинга кластеров.

Prometheus решает две ключевые задачи:

- обеспечивает мониторинг – сбор метрик работы кластеров;
- оповещает об инцидентах, определяемых по установленным критериям для этих метрик (алертинг).

В основе системы лежат два базовых принципа: механизм Service Discovery (протоколы обнаружения сервисов) и конфигурирование из файлов config.

Это означает, что все необходимые нам параметры и настройки упакованы в файл, запускаемый и выполняемый применительно к любому кластеру Kubernetes. Итак, в едином config-файле описываются все принципы и определения, указывается, какие именно хосты, поды, приложения нужно мониторить и по каким параметрам (на основе механизма Service Discovery).

Другие инструменты мониторинга, как правило, подразумевают занесение хоста в систему с указанием его IP-адреса и порта. У Prometheus реализован подход через аннотации: они указывают, что и в каком элементе инфраструктуры нужно мониторить, – это позволяет передавать ответственность за мониторинг от администраторов пользователям кластеров, т. е. разработчикам.

Стандартно Prometheus состоит из трех компонентов:

- *Prometheus-сервер* занимается сбором метрик со всех объектов, которые указаны обязательными для мониторинга;
- *Alert Manager* отправляет оповещения об инцидентах;
- *Exporters* предоставляют метрики в нужном формате.

Это базовая архитектура инструмента – она обеспечивает возможность получать алерты о событиях, чтобы пользователи сервиса могли обратиться к нужному участку и решить проблему.

## Стек инструментов мониторинга

У Prometheus есть один врожденный дефект, который остается актуальным до сих пор, – сложности с хранением больших объемов данных. Система прекрасно использует метрики для анализа текущих проблем «здесь и сейчас», однако для исторической аналитики она не подходит. Решения, которые начали появляться после Prometheus, решали прежде всего именно эту проблему. Например, Victoria Metrics может полностью заменить Prometheus. Еще одно популярное решение – инструмент Thanos,

представляющий собой бэкенд-хранение исторических данных для Prometheus.

Кроме того, стоит отметить систему InfluxDB и весь технологический стек компании Influx, активно развивающийся как аналог Prometheus. В него входит инструмент Telegraf – агент-сборщик метрик, который поддерживает формат Prometheus и множество других форматов данных.

Самое простое решение – использовать Prometheus в комбинации с Grafana для вывода данных

тестируют перспективные разработки в контейнерах или запускают там свои приложения для внутреннего использования;

- *анализ исторических данных* – при необходимости посмотреть изменения количества пользователей или нагрузки ИТ-систем в течение месяцев и даже лет. Для этих целей понадобится расширенный стек мониторинга. Именно так работают крупные компании с большой клиентской базой. Например, когда ИТ-приложение для массового

## Другие инструменты мониторинга подразумевают занесение хоста в систему с указанием его IP-адреса и порта.

в формате дашбордов с отправкой алертов через Alert Manager. Однако если возникнет запрос на исторические данные – в перспективе нескольких месяцев или лет, – придется расширять функциональность Prometheus либо применять другие инструменты.

Если в ИТ-системе имеется много Legacy-компонентов, то не все можно мониторить с помощью Prometheus. В таком случае нужно рассматривать стек Influx, где всеядный (в плане формата данных) Telegraf решает эту проблему.

## Сценарии и структура мониторинга

Можно выделить два сценария мониторинга Kubernetes:

- *анализ метрик «здесь и сейчас»* – для разбора аварий, инцидентов, реализации автоматизирования кластера. В таком случае потребуются свежие метрики с циклом жизни в несколько дней, максимум недель, и можно легко обойтись базовой комплектацией Prometheus. Этот сценарий подходит командам, которые

рынка требует поддержки больших команд программистов и администраторов.

Мониторинг работает на четырех уровнях:

- *«железо»*. Нагрузка на CPU на физическом уровне вплоть до мониторинга температуры процессоров, различные параметры работы СХД, роутеров и другого оборудования;
- *операционная система*. Параметры работы базового системного софта: загрузка вычислительных мощностей CPU, память, занятый объем СХД, количество активных процессов, Load Average и пр.;
- *прикладной*. Приложения, которые работают в контейнерах. Это базы данных, очереди задач и количество сообщений в очереди, записи в БД, время ответа БД и т. д. Вплоть до уровня того софта, который мы разрабатываем и запускаем в контейнерах;
- *бизнес-мониторинг*. Отслеживание бизнес-показателей, связанных на работу ИТ-систем. Например, количество новых уникальных пользователей, купленных товаров и услуг, переходов, транзакций.

Любой мониторинг должен приносить конкретную, видимую пользу для эксплуатации ИТ-систем. При грамотно организованном мониторинге Kubernetes специалист сразу может понять, где именно в кластере есть неполадки, и получить исчерпывающие данные для их устранения. Эффективный алертинг тоже помогает: с ним ответственный специалист реагирует на ошибки оперативно и обоснованно.

## Переход к прозрачности

В последнее время задачи по мониторингу в ИТ трансформируются в общую концепцию Observability (наблюдаемости). Она предполагает полную прозрачность ИТ-систем и понимание, в каком состоянии находятся все элементы в любой момент времени.

Мониторинг – один из компонентов концепции Observability наряду с логированием, трассировкой и учетом метрик работы приложений APM (Application Performance Metrics).

Логирование – это сбор и анализ событий для построения графиков, дашбордов, сценариев для алертов.

Трассировка нужна для отслеживания обработки запроса пользователя в системе – это критически важно в микросервисных системах, где в единый комплекс связано множество компонентов. Например, трассировка даст ответ, как именно запрос пользователя к фронтенду сайта перемещается в рамках системы для получения конечного результата. Можно увидеть, где возникают проблемы или задержки, как системы и компоненты взаимодействуют между собой.

APM отслеживает параметры на уровне взаимодействия подсистем: количество входящих пакетов либо запросов на определенные Endpoint API, внутренние метрики приложения (количество внутренних процессов или тредов).

Концепция Observability значительно упрощает взаимодействие

с ИТ-системами. Чем более понятна система тому, кто ее эксплуатирует, тем она менее подвержена ИБ-инцидентам и различным сбоям.

## Микросервисная актуальность

Концепция наблюдаемости и качественный мониторинг как одна из ее составляющих особенно актуальны для развития микросервисных архитектур. Сложный комплексный продукт, созданный и развиваемый десятком команд разработчиков, представляет собой благодатную среду для проблем в коммуникациях. Они мешают прозрачности и единому пониманию основных терминов, целей и принципов взаимодействия. Без единого четкого мониторинга развивать многие современные ИТ-продукты очень сложно, а число таких проектов в будущем будет только расти.

**Кейс из практики.** Предположим, что руководство поставило задачу ИТ-отделу: понять, как взаимодействуют ИТ-системы между собой с точки зрения бизнес-процессов. Куда приходит запрос, кто его забирает, куда перенаправляет, как обрабатывает, какие действия запускают результаты обработки и т. д. – вплоть до получения результата.

Допустим, что целый месяц мы собирали данные: для этого обклеили в офисе огромную стену стикерами с описанием этапов движения запросов, взаимодействия подсистем и прочими подробностями. Однако успехом инициативы не увенчалась: к концу периода сбора данных схема взаимодействия устарела, поскольку ситуация в реальности меняется быстрее, чем человек успевает ее отследить.

Внедрять observability-решение для контейнеров необходимо сразу. Чтобы в реальном времени отслеживать все взаимодействия инфраструктурных компонентов и управлять их перестройкой на лету. Это помогает быстрее расследовать инциденты, повышает прозрачность затрат на ИТ

и усиливает отказоустойчивость всего решения. Даже самые сложные архитектуры перестают быть черным ящиком с непонятным принципом работы.

## Мониторинг Kubernetes как сервис

Чтобы развернуть Kubernetes, поддерживать его работу и организовать эффективный мониторинг кластеров, необходима целая команда специалистов, хорошо знакомых с технологией. Это отдельная трудоемкая задача.

Ее можно делегировать: Managed Kubernetes в облаке позволяет отдать облачному провайдеру задачи по обновлению и поддержанию работоспособности технологии. Kubernetes-as-a-service – один из наиболее популярных сегодня платформенных сервисов PaaS. К облачному решению K8s подключаются возможности мониторинга.

Например, в Kubernetes как сервисе на платформе VK Cloud Solutions для управления кластерами Kubernetes доступен стек мониторинга, основанный на Prometheus и Alert Manager. Функциональные возможности мониторинга можно добавить сразу, указав эту опцию в соответствующем поле, и он будет развернут одновременно с запуском кластера. Преднастроенные дашборды покажут состояние кластера Kubernetes, приложений в нем и всех параметров работы, которые нужно отслеживать. Эта функциональность не влияет на стоимость сервиса.

Лучшие результаты по наблюдаемости достигаются при создании сквозного решения по мониторингу в облаке – в этом случае нативные инструменты мониторинга собирают данные на уровне виртуальных машин и позволяют строить на основе полученной информации дашборды в едином интерфейсе с мониторингом кластеров Kubernetes, что особенно актуально при анализе ожидаемых пиковых периодов нагрузки на ИТ-системы и обработке непредсказуемых скачков в потреблении. ■