Мотивация

Мониторинг является критически важным компонентом любой современной IT-системы, особенно для крупных компаний, работающих с большими объемами данных и сложных бизнес-процессов. Он позволяет следить за состоянием системы в реальном времени, выявлять проблемы до того, как они приведут к сбоям, а также оптимизировать производительность и ресурсы. Вот несколько ключевых причин, почему мониторинг необходим:

1. Предупреждение сбоев

Мониторинг помогает выявить потенциальные проблемы еще до их возникновения. Например, он может отслеживать такие параметры, как загрузка процессора, использование памяти, сетевой трафик и другие метрики. Если какой-то показатель начинает приближаться к критическому значению, система мониторинга может предупредить команду о необходимости вмешательства. Это предотвращает серьезные аварии и простои, которые могут привести к финансовым потерям и недовольству клиентов.

2. Оптимизация производительности

С помощью мониторинга можно анализировать текущую нагрузку на систему и ее компоненты. Это дает возможность понять, где происходят узкие места, и своевременно принять меры для улучшения производительности. Например, можно перераспределить ресурсы между серверами, обновить оборудование или изменить архитектуру приложения. В результате компания сможет работать быстрее и эффективнее, обеспечивая лучший пользовательский опыт.

3. Улучшение безопасности

Современные системы мониторинга часто включают функции анализа логов и обнаружения аномалий. Это позволяет быстро обнаруживать подозрительные активности, такие как попытки взлома или утечки данных. Чем раньше будет замечена угроза, тем меньше ущерб она нанесет. Кроме того, мониторинг помогает соблюдать требования регуляторов, таких как GDPR или PCI DSS, что важно для многих компаний.

4. Повышение прозрачности и управляемости

Благодаря мониторингу руководство компании получает полную картину состояния своих IT-ресурсов. Это облегчает принятие решений относительно инвестиций в инфраструктуру, оптимизации процессов и распределения ресурсов. Менеджеры могут видеть, какие части системы работают эффективно, а какие требуют улучшений, что способствует более рациональному управлению бизнесом.

5. Быстрое восстановление после аварий

Даже при наличии мониторинга иногда случаются непредвиденные инциденты. Однако благодаря собранным данным можно быстрее определить причину сбоя и восстановить работу системы. Лог-файлы и метрики позволяют провести детальный анализ произошедшего и устранить проблему, минимизируя время простоя.

Что это даст компании?

Снижение рисков: Мониторинг уменьшает вероятность серьезных инцидентов, связанных с отказом оборудования или программных ошибок.

Экономия средств: Предсказуемое управление ресурсами и предотвращение аварий помогают избежать затрат на устранение последствий сбоев.

Повышение удовлетворенности пользователей: Быстрое реагирование на проблемы и стабильная работа системы улучшают восприятие бренда клиентами и партнерами.

Соответствие требованиям регуляторов: Компании, работающие в регулируемых отраслях, обязаны обеспечивать безопасность и надежность своих систем. Мониторинг позволяет соответствовать этим требованиям.

Таким образом, внедрение мониторинга в систему приносит множество преимуществ, начиная от повышения надежности и заканчивая экономией ресурсов.

Выбор подхода к мониторингу

Я бы выбрал метод RED. Т.к. метрики RED измеряют показатели, которые важны для конечных пользователей сервисов.

какие метрики и в каких частях системы отслеживать

* CPU % for shop API
* CPU % for CRM API
* CPU % for MES API

Загрузка процессора показывает, насколько интенсивно используется вычислительная мощность сервера или устройства. Высокий уровень загрузки может указывать на перегрузку системы, что приводит к замедлению работы приложений и ухудшению пользовательского опыта. Отслеживание этого параметра поможет вовремя заметить проблемные участки кода или процессы, потребляющие слишком много ресурсов.

Ярлыки:

instance: Идентификатор конкретного экземпляра сервера или виртуальной машины, чтобы понимать, какая машина испытывает высокую нагрузку.

service: Название сервиса или приложения, которое работает на данном сервере, чтобы локализовать источник нагрузки.

environment: Окружение (production, staging, development), чтобы различать нагрузки в разных средах.

* Memory Utilisation for shop API
* Memory Utilisation for CRM API
* Memory Utilisation for MES API
* Memory Utilisation for shop db instance
* Memory Utilisation for MES db instance

Эта метрика отслеживает количество используемой оперативной памяти. Высокие значения могут свидетельствовать о проблемах с управлением памятью в приложениях (например, утечках памяти), что приведет к снижению производительности и возможным сбоям. Важно контролировать этот параметр, чтобы предотвратить исчерпание доступной памяти.

Ярлыки:

host: Имя хоста или IP-адрес, чтобы знать, на каком устройстве наблюдается проблема.

application: Название приложения, использующего память, для точного определения источника потребления.

memory\_type: Тип памяти (physical, swap), чтобы отличать физическую память от подкачки.

* Number of requests (RPS) for internet shop API
* Number of requests (RPS) for CRM API
* Number of requests (RPS) for MES API
* Number of requests (RPS) per user for internet shop API
* Number of requests (RPS) per user for CRM API
* Number of requests (RPS) per user for MES API

Этот показатель отражает количество запросов, обрабатываемых системой каждую секунду. Высокое значение RPS может сигнализировать о росте трафика или увеличении числа пользователей, что требует масштабирования инфраструктуры. Низкое значение может говорить об уменьшении интереса к сервису или технических проблемах.

Ярлыки:

endpoint: URL конечной точки, чтобы отслеживать популярность конкретных маршрутов.

method: Метод HTTP-запроса (GET, POST, PUT, и т.д.), чтобы дифференцировать типы операций.

response\_code: Код ответа HTTP, чтобы оценить успешность обработки запросов.

* Response time (latency) for shop API
* Response time (latency) for CRM API
* Response time (latency) for MES API

Время отклика измеряет задержку между отправкой запроса клиентом и получением ответа от сервера. Увеличенное время отклика указывает на возможные задержки в обработке запросов, что негативно сказывается на пользовательском опыте. Регулярное отслеживание этого показателя помогает поддерживать высокий уровень обслуживания.

Ярлыки:

route: Маршрут или путь API, который был вызван, чтобы точно узнать, где происходит задержка.

status\_code: Статус-код ответа, чтобы связывать медленные ответы с конкретными результатами.

client\_ip: IP клиента, чтобы исследовать географические особенности запросов.

* Number of HTTP 500 for shop API
* Number of HTTP 500 for CRM API
* Number of HTTP 500 for MES API
* Number of HTTP 500 for shop API

Ошибки 500 являются индикатором внутренних проблем на стороне сервера. Они могут возникать из-за различных факторов, включая ошибки в коде, неправильную конфигурацию или перегрузку системы. Отслеживая частоту появления таких ошибок, можно быстро обнаружить и исправить возникающие проблемы.

Ярлыки:

error\_message: Сообщение об ошибке, которое поможет лучше понять природу проблемы.

request\_path: Путь запроса, вызвавшего ошибку, чтобы локализовать проблемный участок кода.

user\_agent: Информация о браузере или клиенте, который отправил запрос, чтобы учесть специфику среды пользователя.

* Size of S3 storage
* Size of shop db instance
* Size of MES db instance

Объем свободного дискового пространства важен для обеспечения нормальной работы системы. Недостаток места на диске может привести к остановке сервисов, потере данных и другим серьезным проблемам. Постоянный контроль за использованием диска позволит заранее подготовиться к увеличению объема данных.

Ярлыки:

disk\_partition: Раздел диска, чтобы отслеживать заполненность отдельных томов.

filesystem: Файловая система (ext4, ntfs), чтобы учитывать особенности разных файловых систем.

mount\_point: Точка монтирования, чтобы понимать, какое хранилище используется.

* Number of dead-letter-exchange letters in RabbitMQ

Метрика отслеживает количество сообщений, которые были перенаправлены в обменник мертвых писем (Dead Letter Exchange, DLX). Это происходит, когда сообщение не может быть доставлено или обработано, например, из-за отказа очереди или превышения времени жизни сообщения. Высокая частота таких сообщений может указывать на проблемы с производительностью или конфигурацией системы обмена сообщениями.

Ярлыки:

queue: Имя очереди, из которой было удалено сообщение, чтобы локализовать источник проблемы.

reason: Причина удаления сообщения (например, expired, rejected, maxlen), чтобы понять, почему сообщение попало в DLX.

routing\_key: Ключ маршрутизации, который использовался для доставки сообщения, чтобы анализировать маршруты сообщений.

* Number of message in flight in RabbitMQ

Эта метрика отслеживает количество сообщений, находящихся в процессе передачи между брокером и потребителем, но ещё не подтверждённых потребителем. Большое количество сообщений в полёте может указывать на перегрузку потребителей или проблемы с сетевыми соединениями.

Ярлыки:

channel: Канал, через который передаются сообщения, для идентификации конкретных каналов с высокой нагрузкой.

consumer\_tag: Тег потребителя, чтобы различать нагрузку на разных потребителей.

exchange: Обменник, через который проходят сообщения, чтобы отслеживать активность по различным обменникам.

* Number of connections for shop db instance
* Number of connections for MES db instance

Метрика отслеживает общее количество текущих соединений с вашим сервисом или системой. Она полезна для оценки текущей нагрузки на серверы, выявления пиковых периодов активности и прогнозирования будущих потребностей в ресурсах. Также она помогает обнаружить аномалии, такие как внезапные увеличения или уменьшения числа соединений, что может указывать на проблемы с безопасностью или изменениями в поведении пользователей.

Ярлыки:

service: Название сервиса или компонента системы, к которому устанавливаются соединения, чтобы различать нагрузку на разные части системы.

type\_of\_connection: Тип соединения (например, HTTP, SSH, DB), чтобы анализировать различные виды соединений.

origin: Источник соединения (IP-адрес или географическое расположение), чтобы отслеживать запросы из разных регионов или стран.

status: Статус соединения (active, idle, closed), чтобы различать активные и неактивные соединения.

* Number of simultanious sessions for shop API
* Number of simultanious sessions for CRM API
* Number of simultanious sessions for MES API

Метрика отслеживает количество активных сессий, происходящих одновременно в системе. Это важный показатель для оценки текущей нагрузки на серверы, выявления пиковых периодов активности и прогнозирования будущих потребностей в ресурсах. Она также помогает в обнаружении аномального поведения, например, резкого роста числа сессий, что может указывать на атаку или неожиданный рост популярности сервиса.

Ярлыки:

session\_type: Тип сессии (например, web, API, mobile), чтобы различать активность по разным платформам.

user\_id: Идентификатор пользователя, чтобы отслеживать активность конкретных пользователей.

duration: Продолжительность сессии, чтобы анализировать среднюю продолжительность взаимодействия.

geolocation: Географическое местоположение пользователя, для анализа региональной активности.

device\_type: Тип устройства (например, desktop, smartphone, tablet), чтобы понимать, откуда поступают сессии.

План действий

Установка и настройка Prometheus

Prometheus настраивается для использования собственной временной базы данных (TSDB).

Добавляем экспортеры (Spring Actuator, RabbitMQ Exporter, PostgreSQL Exporter, …).

Проверяется, что Prometheus успешно собирает метрики.

Установка и настройка Grafana

Установка Grafana

Prometheus как источник данных в Grafana.

Создание дашбордов:

Создаются дашборды для визуализации метрик: нагрузка на CPU, использование памяти, время отклика сервисов, количество ошибок и т.д.

Настройка алертинга

Настройка Alertmanager:

Ставится и настраивается Alertmanager для обработки алертов из Prometheus.

Определяются правила для генерации алертов на основе пороговых значений метрик.

Интеграция с мессенджерами:

Alertmanager интегрируется с корпоративными мессенджерами (например, Slack, Telegram) для отправки уведомлений.