GeekBrains

Мониторинг распределенных приложений и сервисов

Система мониторинга на основе Zabbix

На этом уроке

Мы узнаем, как установить и сконфигурировать Zabbix.

Оглавление

Общие сведения о мониторинге

Мониторинговые системы в современном мире — неотъемлемая часть любой инфраструктуры. Они собирают информацию о наших серверах и сервисах, которая отражает их состояние. Собранные данные можно визуализировать для понимания текущей ситуации и для анализа поведения за определённый период. Помимо визуализации, на основе собираемых данных можно настроить предупреждения о возможных сбоях и подключить к ним оповещения. Настроив систему мониторинга должным образом, мы сможем:

1. Отслеживать состояние объектов и оперативно реагировать на сбои.

2. Анализируя данные, заметить слабые места и улучшить качество объекта.
3. Автоматизировать процессы на основе полученных данных.

Метрика (временной ряд) — единица собранной информации об объекте, для неё характерен некоторый набор данных — timestamp, набор тэгов/лейблов и само значение.

Метрики условно делятся на разные группы, например :

- обычные (CPU, Memory, Disk, Network, health check…) и бизнес-метрики (количество пользователей в день, количество транзакций в неделю, средняя цена покупок…);
- инфраструктурные метрики (CPU, Memory, Disk, Network, health check…) и метрики сервиса (количество обращений в период времени, время отклика, время обработки запроса…).

Хранить данные можно в как SQL, так и в NoSQL-базах данных, но для метрик рекомендуется использовать специально разработанные для них базы данных временных рядов — TSDB.

Alerting — неотъемлемая часть систем мониторинга, функция которой — сообщить о проблеме посредством различных каналов связи. Это может быть рассылка электронных писем, сообщение в мессенджер, SMS или даже звонок.

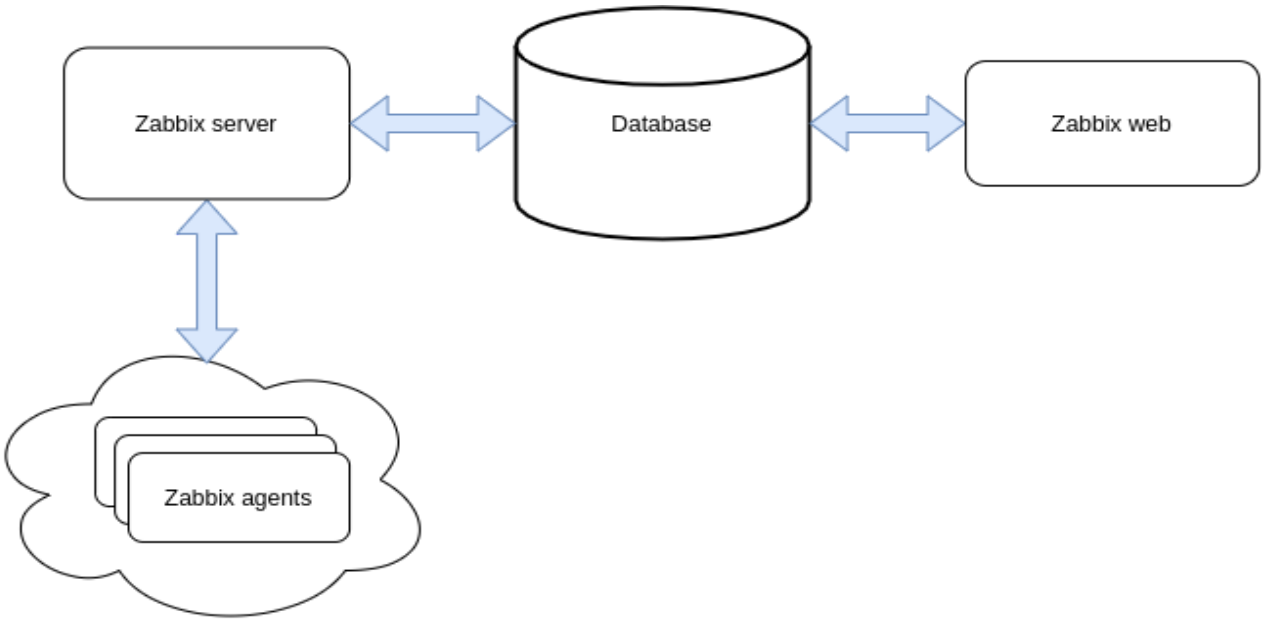
Zabbix

Теперь познакомимся с одной из систем мониторинга — Zabbix. Это программное обеспечение для мониторинга параметров сети, жизнеспособности и целостности серверов (определение взято с официального сайта). Zabbix имеет очень гибкие конфигурации для сбора данных и их анализа и множество настроек для оповещения. Весной 2020 года был выпущен новый релиз Zabbix 5.0, в котором был переработан веб-интерфейс, добавлено множество новых функций и интеграций с другими системами.

Архитектура Zabbix

Zabbix состоит из нескольких компонентов:

1. Zabbix server — основной компонент системы, выполняет обработку данных, вычисляет триггеры, отправляет оповещения.
2. Database (PostgreSQL, MySQL) — содержит конфигурацию и собранные данные.
3. Zabbix agent — разворачивается на наблюдаемых объектах. Локально собирает и отправляет данные Zabbix server.
4. Zabbix proxy (опционально) — собирает данные от имени Zabbix server.
5. Zabbix Java gateway — компонент для нативной поддержки мониторинга JMX-приложений
6. Zabbix web — веб-интерфейс.



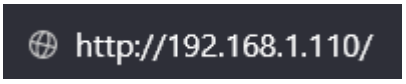
Установка

Запустить Zabbix можно, как и полагается, из пакетов, собрать из исходников и также в docker-контейнерах. Самый простой способ опробовать Zabbix — это, конечно, запустить его в Docker. В официальном репозитории приведены различные варианты compose-файлов. Воспользуемся одним из них.

Чтобы запустить Zabbix, выполним следующие команды:

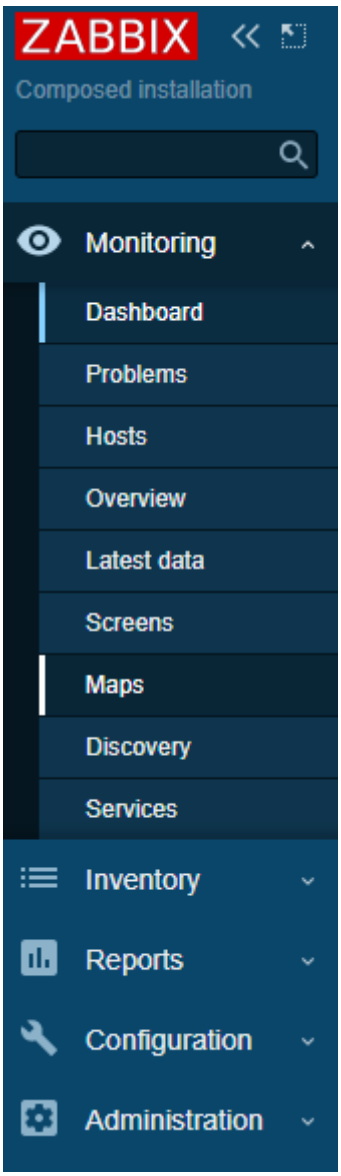
```
git clone https://github.com/zabbix/zabbix-docker.git
cd zabbix-docker
docker-compose -f docker-compose_v3_centos_pgsql_latest.yaml up -d
```

После этих нехитрых манипуляций мы можем открыть наш веб-интерфейс и полюбоваться на него, набрав в браузере ip-адрес zabbix-server Например у меня это:



Zabbix-web

Вкладка Monitoring



Перейдём к рассмотрению рабочего окружения.

Первое, что мы видим при подключении — окно для ввода логина и пароля (по умолчанию Admin:zabbix).

После того, как мы залогинились, мы видим основной дашборд. С правой стороны навигационная панель, разбитая по категориям. Секция Monitoring содержит всю информацию по собираемым данным. Здесь внимание следует уделить *Dashboard* (стартовый дашборд), скорее всего, именно сюда вы будете чаще всего смотреть. Также может быть полезна вкладка *Latest data*, где можно посмотреть последние собранные данные, и *Problems*, где мы можем видеть текущие проблемы, и проблемы, которые срабатывали в прошлом.

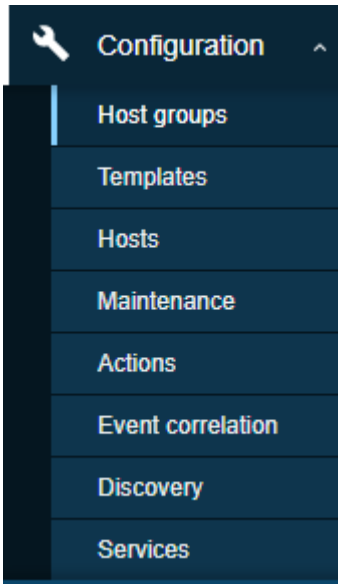
Внимание! *В Latest data для каждой метрики можно посмотреть графики изменений.*

Вкладка Reports

Как понятно из названия, здесь содержатся различные отчёты. Здесь ничего не настраивается, можно только посмотреть.

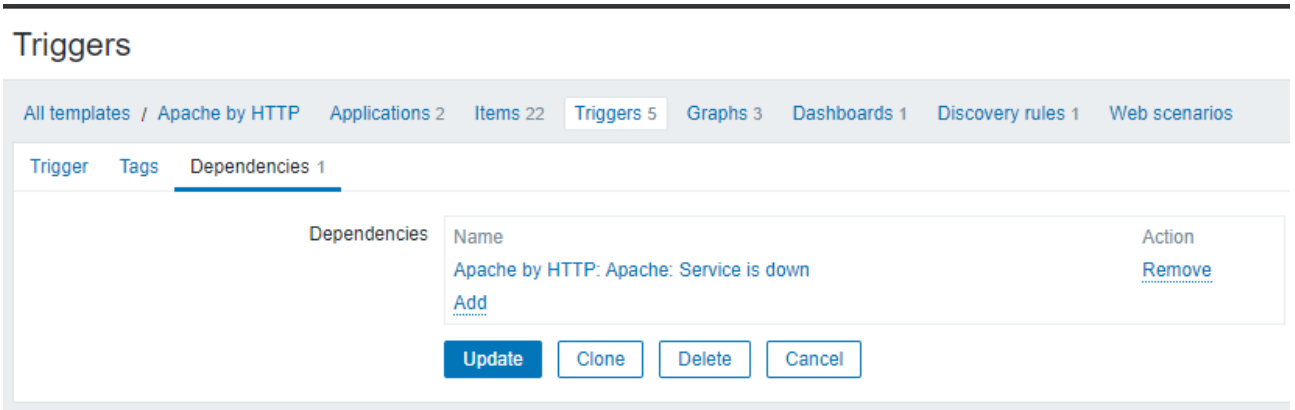
Вкладка Configuration

На этой вкладке мы остановимся более подробно. Здесь содержатся все элементы конфигурации для каждого узла, который мы хотим мониторить.

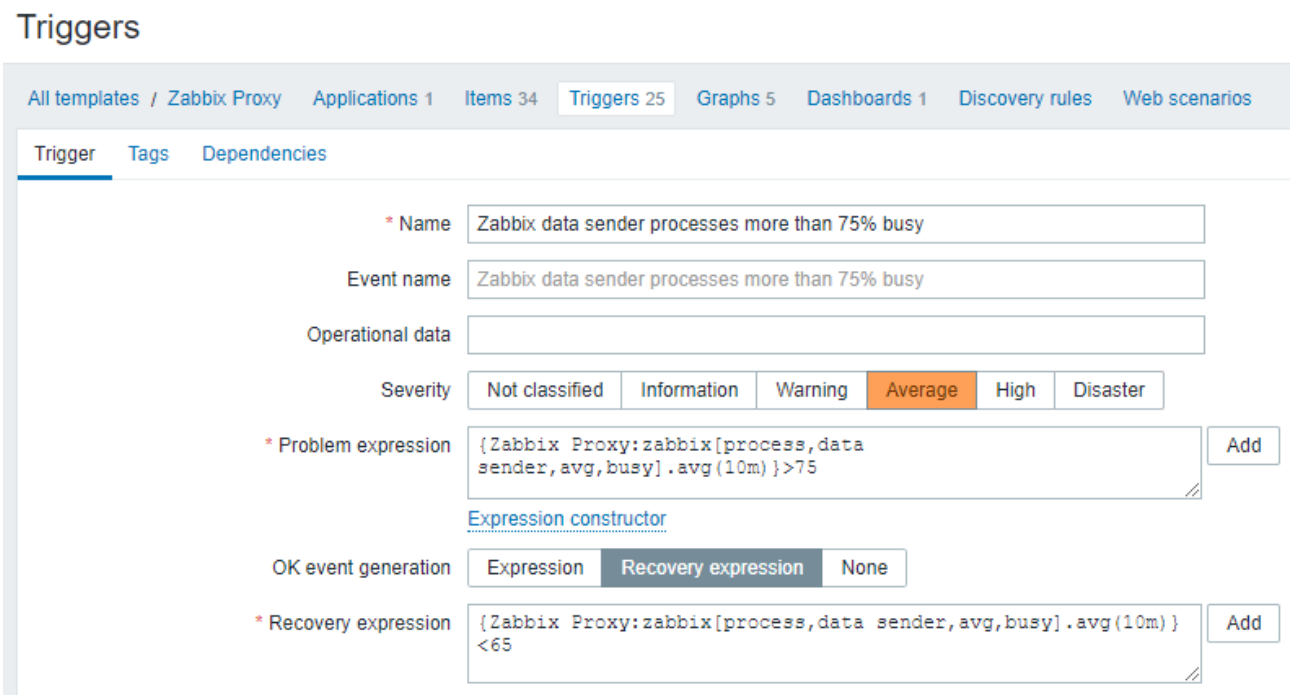


- **Host groups** — предназначены исключительно для группировки;
- **Hosts** — здесь мы создаём узлы, за которыми хотим наблюдать.У каждого узла должна быть *группа* и *уникальное имя*. Здесь же настраиваются элементы данных, которые собирают данные с узла, триггеры, которые будут срабатывать по определённым условиям, графики на основе собранных данных, можно также настроить discovery rules для автоматического добавления узлов сети и различные веб-проверки во вкладке web scenario;
- **Templates** — шаблоны, т. е., сборник элементов данных и триггеров, discovery rules, которые можно навесить как на отдельные хосты, так и на группу хостов;
- **Maintenance** — период обслуживания для узлов и групп.
- **Actions** — действия, которые могут выполняться при определённых условиях, например, срабатывание триггера, обнаружение нового хоста.
- **Event correlation** — позволяет сопоставить проблему и её решение;
- **Discovery** — настройка правил автообнаружения;
- **Services** — настройка мониторинга инфраструктуры более высокого уровня.

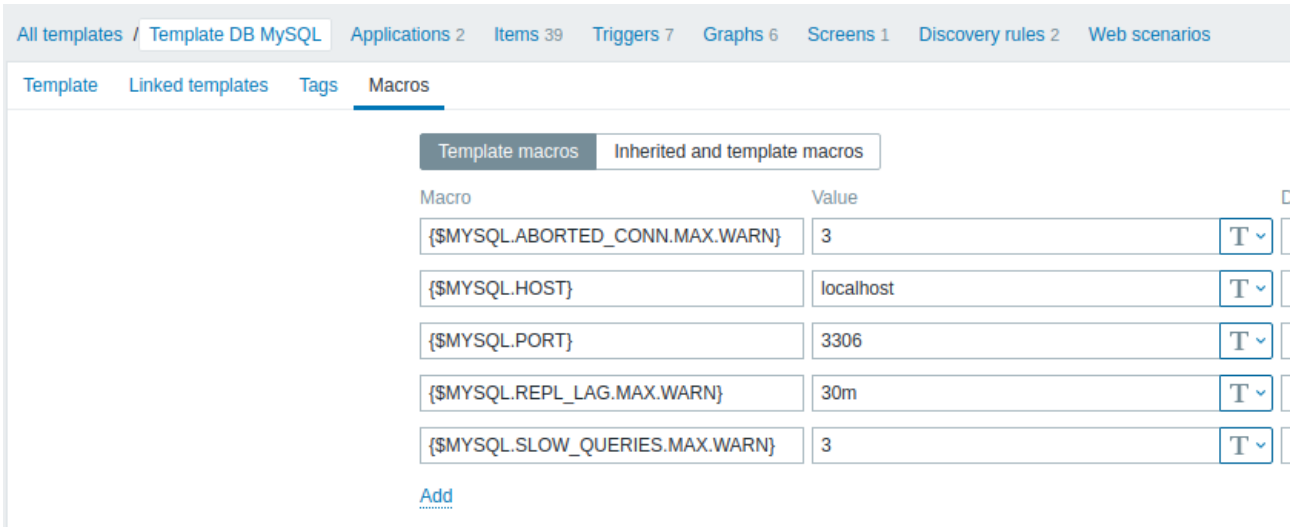
Внимание! *Если вы не хотите срабатывания множества ошибочных триггеров (например из-за сбоя в сети стали недоступны ваши узлы) рекомендуется устанавливать их иерархические зависимости. Также зависимости рекомендуется выставлять при ступенчатом срабатывании триггеров. Чтобы задать зависимость, откройте диалог настройки триггера. Далее нажмите на Dependencies и добавьте триггер, от которого он будет зависеть.*



Внимание! *Также можно добавить Recovery expression для некоторых триггеров, это будет полезно, когда собираемые данные постоянно находятся на границе трешхолдов.*



В различных ситуациях можно использовать макросы, что, несомненно, упрощает конфигурацию множества узлов. Например, мы можем заменить всего одно значение и при этом поменять трешхолд для нескольких триггеров, где он применяется. Чтобы добавить или изменить макросы для шаблона, выберите шаблон (например **Template DB MySQL**) и перейдите на вкладку **Macros.**



Ещё есть полезная функция *Mass update*, как понятно из названия, она позволяет менять значения для нескольких элементов сразу.

Итак, последовательность действий для мониторинга какого-либо узла примерно следующая:

1. Устанавливаем Zabbix-agent на хост.
   Внимание! *Существует два типа проверок агентов: активные и пассивные, по умолчанию используются пассивные проверки, это значит, что zabbix-server или zabbix-proxy запрашивает данные и агент отправляет результат обратно. Активные проверки подразумевают, что агент сам запрашивает список элементов данных для предварительной обработки, уменьшает нагрузку на сервер.*

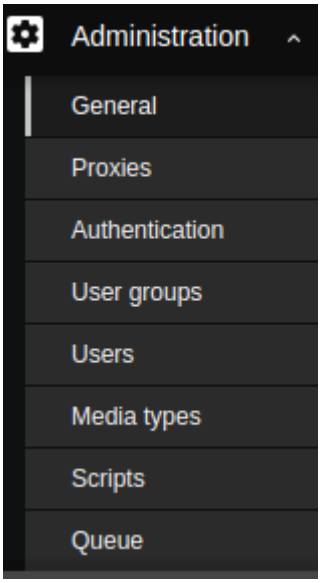
2. Создаём хост на Zabbix-server, назначаем имя, присваиваем группу, указываем агента.
3. Навешиваем на него темплейт или создаём самостоятельно **Items** и **Triggers**.

   Внимание! *При конфигурации Items можно для получаемых данных использовать **Preprocessing**, который позволяет их преобразовать. Важно иметь в виду, что в случае неправильной настройки этапов преобразования элемент данных становится неподдерживаемым.*

4. Настраиваем **Actions**, например, на рассылку уведомлений при возникновении определённых событий или на выполнение скриптов.
   Внимание! *Стоит обратить внимание на конфигурирование ступенчатой нотификации (эскалация) при возникновении проблем. Например если в течение 10 минут никто из ответственных не отреагировал на инцидент, стоит отправить следующее оповещение, на другой возможный источник связи или другой, более высокой группе ответственных*

Вкладка Administration

Большая часть элементов в этой вкладке настраиваются один раз и больше не используется. Здесь можно подтюнить внешний вид веб-интерфейса, настроить аутентификацию, установить права доступа или например задать какие-либо глобальные параметры.



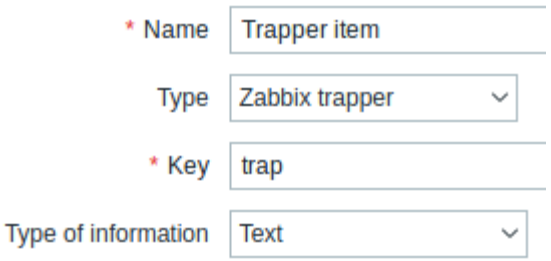
На что, на наш взгляд, стоит обратить внимание:

1. General->Housekeeping — здесь можно настроить период хранения различных данных.
   Внимание! *При большом количестве объектов мониторинга следует рассмотреть варианты оптимизации хранения данных, и либо перейти на PostgreSQL с поддержкой TSDB, либо использовать партиционирование в MySQL.*
2. General->Macros — здесь можно задать общесистемные макросы.
3. Proxies — Настройки для распределенного мониторинга.
4. Authentication — здесь можно задать глобальный метод аутентификации (внутренний, HTTP, LDAP).
5. User groups — здесь можно задать ограничения на доступ к веб-интерфейсу.
6. Users — настройка прав доступа, методов оповещения.

Zabbix-sender

Zabbix-sender — это утилита командной строки, которая позволяет отправлять данные на Zabbix server для последующей их обработки, обычно используется в кастомных скриптах.

```
/bin/zabbix_sender -z zabbix -s "Linux DB3" -k db.connections -o 43
# -z - имя или ip zabbix server
# -s - имя узла сети
# -k - ключ элемента данных
# -o - значения, которые вы хотите отправить
```



Внимание! *Тип элемента отправляемых данных должен быть траппер.*

Низкоуровневое обнаружение (Low Level Discovery, LLD)

LLD позволяет динамически создавать элементов данных, триггеры и графики для различных объектов. Например, если открыть шаблон Linux by Zabbix agent, то во вкладке Discovery rules мы можем увидеть три правила для автообнаружения — для файловых систем, блочных устройств и интерфейсов.

Общая схема работы для discovery rules такова:

1. Создаём правило, с ключом, который ищет объекты, например CPU.
2. Создаём шаблоны элементов данных, с триггерами и графиками.

Внимание! *Можно также обнаруживать не только отдельные элементы, но и сами узлы сети.*

Масштабирование

Для большой инфраструктуры, где мощностей одного Zabbix Server недостаточно, используются специальные сервисы — Zabbix Proxy. Zabbix proxy работает как отдельный сервер, собирает данные с подключённых к нему устройств и отправляет на Zabbix Server. Для Zabbix Proxy нужна отдельная база данных.

Zabbix API

Zabbix API — часть веб-интерфейса, позволяет посредством различных методов получать и изменять конфигурацию, получать данные истории. Широко используется для автоматизации рутинных процессов. Большинство методов поддерживают 4 основных действия: get, create, update, delete.

Чтобы воспользоваться всеми возможностями этого инструмента, необходимо получить ключ аутентификации, и все дальнейшие запросы выполнять с ним. Это можно сделать как обычным curl-запросом, так и воспользоваться одной из библиотек, предлагаемых сообществом. Команда из консоли будет выглядеть примерно так:

```
curl -s -H "Content-Type: application/json-rpc" -d '{"jsonrpc": "2.0", "method": "user.login", "params": {"user": "Admin", "password": "zabbix"}, "id": 1, "auth": null}' http://localhost/api_jsonrpc.php | jq '.result'
```

Соответственно, получив ключ, мы сможем выполнять различные действия. Например, попробуем получить все узлы сети, а затем добавим ещё один.

```
curl -s -H "Content-Type: application/json-rpc" -d '{"jsonrpc": "2.0","method":
"host.get","params":  {  "output":  ["hostid",  "host"],"selectInterfaces":
["interfaceid","ip"]},"id":  2,"auth":  "f68ea6d77a3952f896d7a77241895731"}'
http://localhost/api_jsonrpc.php  | jq '.result | .[].host'

curl -XPOST -s -H "Content-Type: application/json-rpc" -d  '{"jsonrpc": "2.0",
"method": "host.create","params": {"host": "TEST server","interfaces": [{"type":
1, "main": 1, "useip": 1, "ip": "192.168.3.1",  "dns": "",  "port": "10050" } ],
"groups": [ { "groupid": "2"}],"inventory_mode": 0,"inventory": {"macaddress_a":
"01234","macaddress_b":  "56768"}},"auth":  "f68ea6d77a3952f896d7a77241895731",
"id": 1}' http://localhost/api_jsonrpc.php | jq
```

После чего ещё раз выведем все хосты и убедимся, что теперь в списке присутствует новый.

Практическое задание

1. jj

Глоссарий

Узел сети (host) — это устройства, которые необходимо мониторить.

Элемент данных (Item) — сущности, кто собирает данные с узла сети.

Триггер (trigger) — это логические выражения, которые оценивают данные, собранные элементами данных, и отражают текущее состояние системы.

Дополнительные материалы

1. Система мониторинга Zabbix.
2. Библиотеки Zabbix (ссылка для скачивания)
3. HighLoad++, Андрей Гущин (Zabbix): высокая производительность и нативное партиционирование.
4. Активный и пассивный zabbix-агент.

5. Использование партиционирования в MySQL для Zabbix с большим количеством объектов мониторинга.
6. Высокая производительность и нативное партиционирование: Zabbix с поддержкой TimescaleDB.

Используемые источники

1. Руководство по Zabbix.
2. zabbix/zabbix-docker: Official Zabbix Dockerfiles.
3. Zabbix: низкоуровневое обнаружение.
4. Гайд по Zabbix: низкоуровневое обнаружение