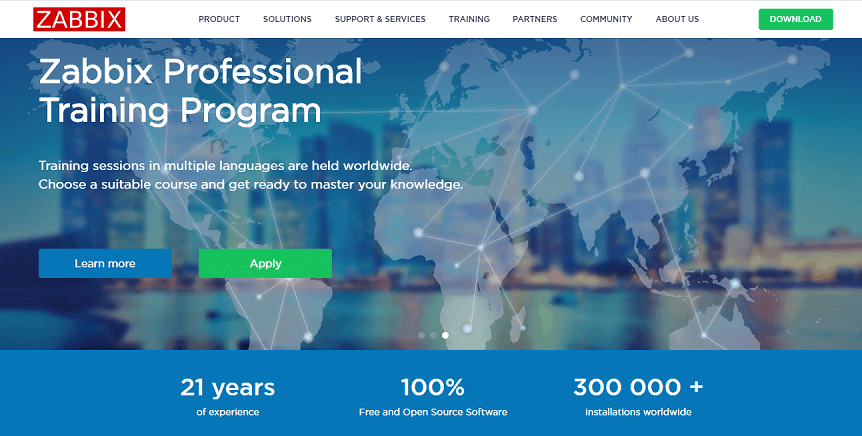
Zabbix — это универсальный инструмент мониторинга, способный отслеживать динамику работы серверов и сетевого оборудования, быстро реагировать на внештатные ситуации и предупреждать возможные проблемы с нагрузкой. Система мониторинга Zabbix может собирать статистику в указанной рабочей среде и действовать в определенных случаях заданным образом.

В этой обзорной статье расскажем об основных принципах и ключевых инструментах, на которых построена универсальная система мониторинга Zabbix.

Обзор

Систему создал Алексей Владышев на языке Perl. Впоследствии проект подвергся серьезным изменением, которые затронули и архитектуру. Zabbix переписали на C и PHP. Открытый исходный код появился в 2001 г., а уже через три года выпустили первую стабильную версию.

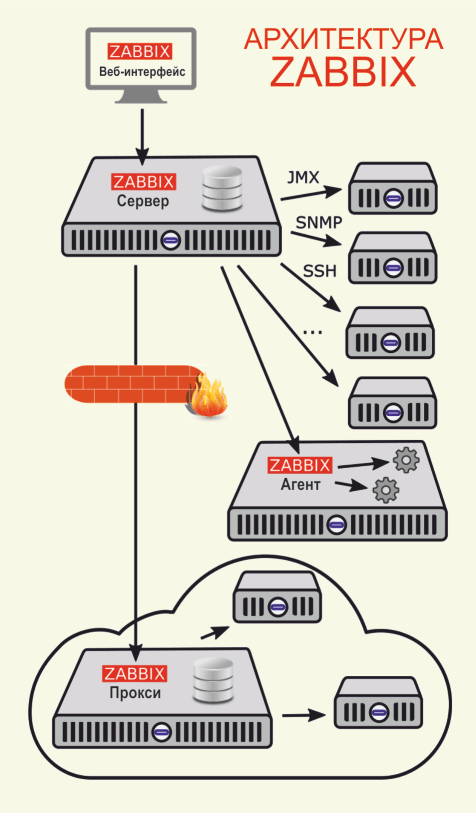
Веб-интерфейс Zabbix написан на PHP. Для хранения данных используются MySQL, Oracle, PostgreSQL, SQLite или IBM DB2.



На данный момент доступна система Zabbix 4.4. Скачать ее можно на [официальном сайте](https://www.zabbix.com/ru/download). Там же можно найти официальные курсы и вебинары для начинающих пользователей системы.

Далее рассмотрим, из чего состоит и как работает технология Zabbix в доступном формате «для чайников».

Архитектура Zabbix

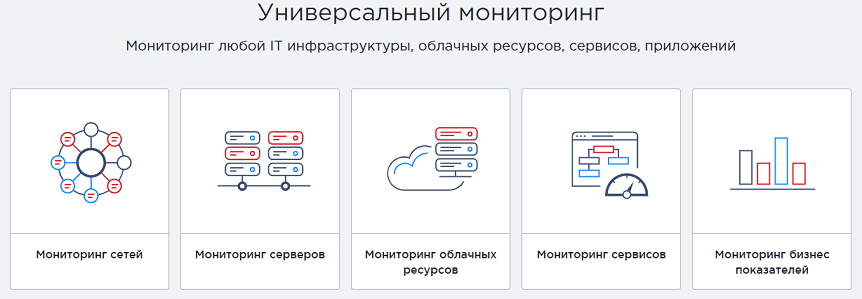


У Zabbix есть 4 основных инструмента, с помощью которых можно мониторить определенную рабочую среду и собирать о ней полный пакет данных для оптимизации работы.

1. **Сервер** — ядро, хранящее в себе все данные системы, включая статистические, оперативные и конфигурацию. Дистанционно управляет сетевыми сервисами, оповещает администратора о существующих проблемах с оборудованием, находящимся под наблюдением.
2. **Прокси** — сервис, собирающий данные о доступности и производительности устройств, который работает от имени сервера. Все собранные данные сохраняются в буфер и загружаются на сервер. Нужен для распределения нагрузки на сервер. Благодаря этому процессу можно уменьшить нагрузку на процессор и жесткий диск. Для работы прокси Zabbix отдельно нужна база данных.
3. **Агент** — программа (демон), которая активно мониторит и собирает статистику работы локальных ресурсов (накопители, оперативная память, процессор и др.) и приложений.
4. **Веб-интерфейс** — является частью сервера системы и требует для работы веб-сервер. Часто запускается на том же физическом узле, что и Zabbix.

Основные возможности

Функционал включает в себя общие проверки для наиболее распространенных сервисов, в том числе СУБД, SSH, Telnet, VMware, NTP, POP, SMTP, FTP и т.д. Если стандартных настроек системы недостаточно, их можно изменить самостоятельно или же пользоваться дополнением через API.



Стандартные функции системы

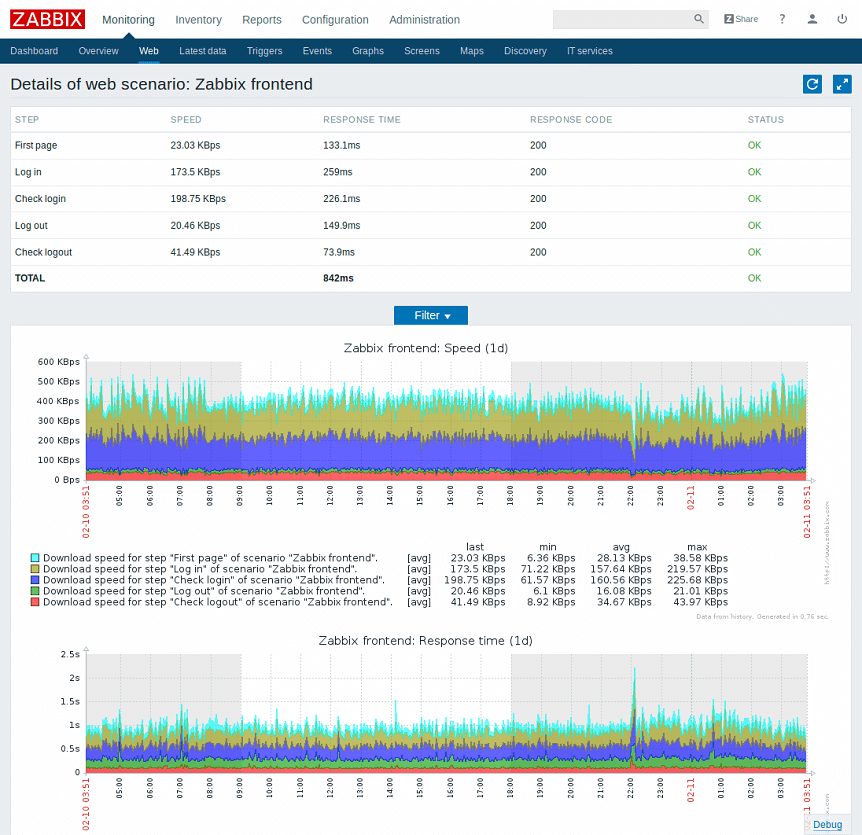
* Контроль нагрузки на процессор, касается и отдельных процессов.
* Сбор данных об объеме свободной оперативной и физической памяти.
* Мониторинг активности жесткого диска.
* Мониторинг сетевой активности.
* Пинг для проверки доступности узлов в сети.

Проверки

Для описания системы мониторинга Zabbix существует два ключевых понятия:

* **Узлы сети** — рабочие устройства и их группы (серверы, рабочие станции, коммутаторы), которые необходимо проверять. С создания и настойки узлов сети обычно начинается практическая работа с Zabbix.
* **Элементы данных** — набор самостоятельных метрик, по которым происходит сбор данных с узлов сети. Настройка элементов данных производится на вкладке «Элемент данных» или в автоматическом режиме — через подключение шаблона.

Сам Zabbix-агент способен отражать текущее состояние физического сервера, собирая совокупность данных. У него достаточно много метрик. С их помощью можно проверить загруженность ядра (Processor load), время ожидания ресурсов (CPU iowait time), объем системы подкачки (Total swap space) и многое другое.



В Zabbix существует целых 17 способов, дающих возможность собирать информацию. Указанные ниже, входят в число наиболее часто применяемых.

* **Zabbix agent** (Zabbix-агент) — сервер собирает информацию у агента самостоятельно, подключаясь по определенному интервалу.
* **Simple check** (Простые проверки) — простые операции, в том числе пинг.
* **Zabbix trapper** (Zabbix-траппер) — сбор информации с трапперов, представляющих собой мосты между используемыми сервисами и самой системой.
* **Zabbix aggregate** (Zabbix-комплекс) — процесс, предусматривающий сбор совокупной информации из базы данных.
* **SSH agent** (SSH-агент) — система подключается по SSH, использует указанные команды.
* **Calculate** (Вычисление) — проверки, которые система производит, сопоставляя имеющиеся данные, в том числе после предыдущих сборов.

У проверок есть заданные шаблоны (Templates), которые упрощают создание новых. Кроме обычных операций существует возможность регулярно проверять доступность веб-сервера с помощью имитации запросов браузера.

Проверка через пользовательский параметр

Чтобы выполнить проверку через агент, нужно прописать соответствующую команду в конфигурационный файл Zabbix-агента в качестве пользовательского параметра (UserParameter). Сделать это можно с помощью выражения следующего вида:

UserParameter=<ключ>,<команда>

Помимо самой **команды**, приведенный синтаксис содержит уникальный (в пределах узла сети) **ключ**элемента данных, который надо придумать самостоятельно и сохранить. В дальнейшем, ключ можно использовать для ссылки на команду, внесенную в пользовательский параметр, при создании элемента данных.

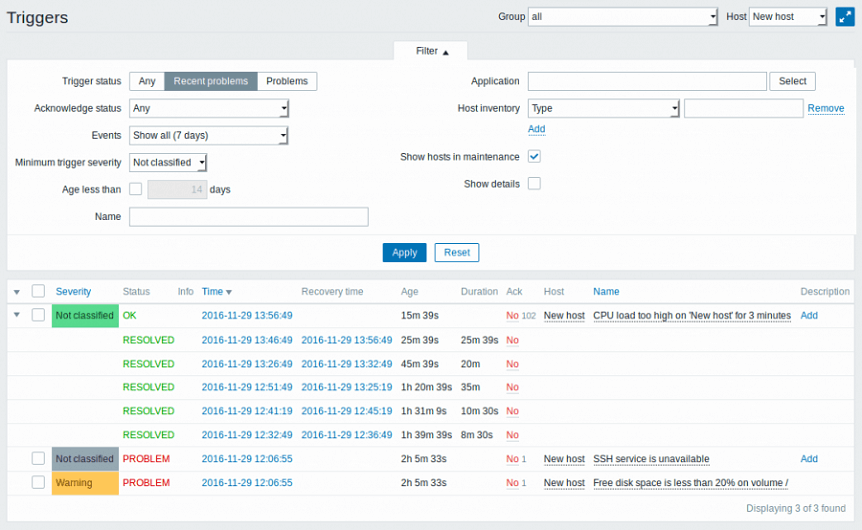
Пример

UserParameter=ping,**echo** 1

С помощью данной команды можно настроить агент на постоянное возвращение значения «1» для элемента данных с ключем «ping».

Триггеры

Это логические выражения со значениями FALSE, TRUE и UNKNOWN, которые используются для обработки данных. Их можно создать вручную. Перед использованием триггеры возможно протестировать на произвольных значениях.



У каждого триггера существует уровень серьезности угрозы, который маркируется цветом и передается звуковым оповещением в веб-интерфейсе.

* **Не классифицировано**(Not classified) — серый.
* **Информация**(Information) — светло-синий.
* **Предупреждение**(Warning) — жёлтый.
* **Средняя**(Average) — оранжевый.
* **Высокая**(High) — светло-красный.
* **Чрезвычайная**(Disaster) — красный.

Некоторые функции триггеров

* **abschange** — абсолютная разница между последним и предпоследним значением (0 — значения равны, 1 — не равны).
* **avg** — среднее значение за определенный интервал в секундах или количество отсчетов.
* **delta** — разность между максимумом и минимумом с определенным интервалом или количеством отсчетов.
* **change** — разница между последним и предпоследним значением.
* **count** — количество отсчетов, удовлетворяющих критерию.
* **date** — дата.
* **dayofweek** — день недели от 1 до 7.
* **diff** — у параметра есть значения, где 0 — последнее и предпоследнее значения равны, 1 — различаются.
* **last** — любое (с конца) значение элемента данных.
* **max**\**min** — максимум и минимум значений за указанные интервалы или отсчеты.
* **now** — время в формате UNIX.
* **prev** — предпоследнее значение.
* **sum** — сумма значений за указанный интервал или количество отсчетов.
* **time** — текущее время в формате HHMMSS.

Прогнозирование

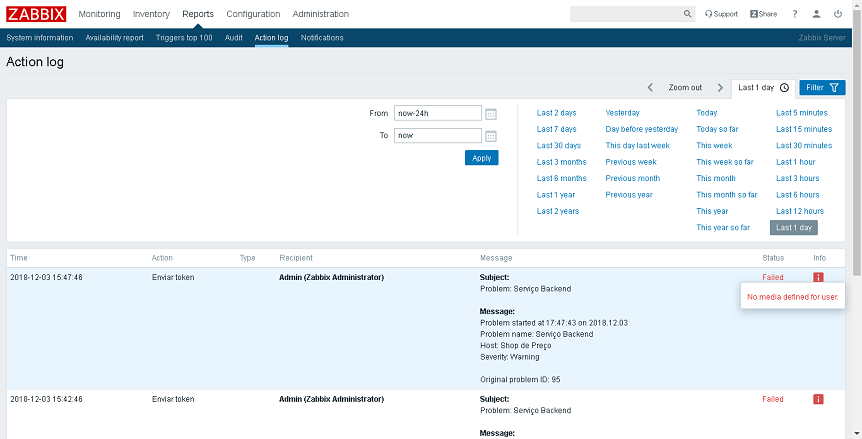
Триггеры обладают еще одной важной функцией для мониторинга — прогнозированием. Она предугадывает возможные значения и время их возникновения. Прогноз составляется на основе ранее собранных данных.

Анализируя их, триггер выявляет будущие проблемы, предупреждает администратора о возникшей вероятности. Это дает возможность предотвратить пики нагрузки на оборудование или заканчивающееся место на жестком диске.

Функционал прогнозирования добавили с обновлением системы 3.0, вышедшим в феврале 2016 года.

Действие

Действие (Action) представляет собой заданную реакцию на событие (Event). Действие может устанавливаться автоматически или вручную как для одного из событий, так и для целой группы.



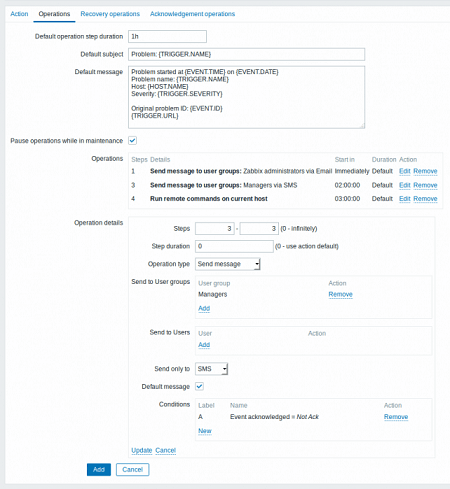
Параметры действий

* **Name** — имя действия.
* **Event source** — источник события. Источниками событий служат обнаружение (**Discovery** Events), авторегистрация (**Auto registration** Events) или заданный триггер (**Trigger** Events).
* **Enable escalations** — разрешение на эскалацию событий.
* **Period** — период времени для шага эскалации, указывается в секундах.
* **Default subject** — указывается, кто извещается по умолчанию.
* **Default message** — стандартный текст сообщения.
* **Recovery message** — текст уведомления после решения проблемы.
* **Recovery subject** — субъект, которого извещают после операции.
* **Status** — статус действия, может быть «активно» и «запрещено».

Для событий, вызванных триггером или обнаружением, есть свои типы условий. Например, «Application» с операторами «=», «like» и «not like» значит, что триггер является частью указанного приложения. Или «Service type» с операторами «=», «<»и «>» предусматривает, что обнаруженный сервис совпадает с указанным.

Операции

Пользователь может указать для событий операцию или группу операций.

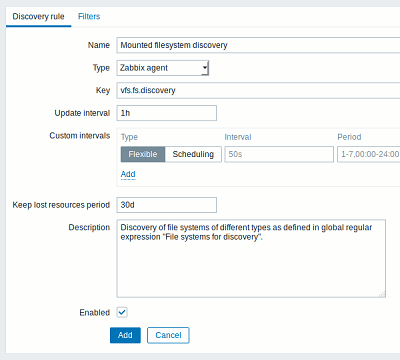


Параметры операций

* **Step** — при эскалации событий.
* **Operation type** — действия на определенном шаге, например, «Send message» или «Execute command».
* **Event Source** — источник событий.
* **Send message to** — отдельное сообщение (Single user) или групповое (User group).
* **Default message** — текст по умолчанию.
* **Subject** — кого оповещает система.
* **Message** — текст сообщения.
* **Remote command** — команда для удаленного управления.

Низкоуровневое обнаружение

Функция Низкоуровневого обнаружения (LLD) автоматически создает элементы и триггеры, которые позволяют отслеживать системы сервера, находящимся под наблюдением. Включение функции происходит через настройку атрибутов, которую можно сделать, пройдя по пути: «Настройка» → «Шаблоны» → «Обнаружение» (вкладка в строке с шаблоном) → вкладки «Правила обнаружения»/«Фильтры».



Что можно обнаружить

* Распространённые OID, используемые SNMP.
* Сетевые интерфейсы.
* Процессоры, их ядра.
* Файловые системы.
* Службы Windows.
* ODBC.

Дополнительные типы

Задать собственные типы низкоуровневого обнаружения возможно с применением формата JSON. Типы проверок, для которых можно указать список портов и интервал для них:

* SSH;
* LDAP;
* SMTP;
* FTP;
* HTTP;
* POP;
* NNTP;
* IMAP;
* TCP.

Если хост пропадает или обнаруживается, для события можно привязать любое действие — условия и операцию для них.

Прокси

Функция буферизации через прокси используется в том случае, когда существующая инфраструктура достаточно большая, а выделенный сервер не способен нести такую нагрузку. Прокси выступает промежуточным звеном, которое собирает информацию с агентов в буфер, а после отправляет данные на сервер.

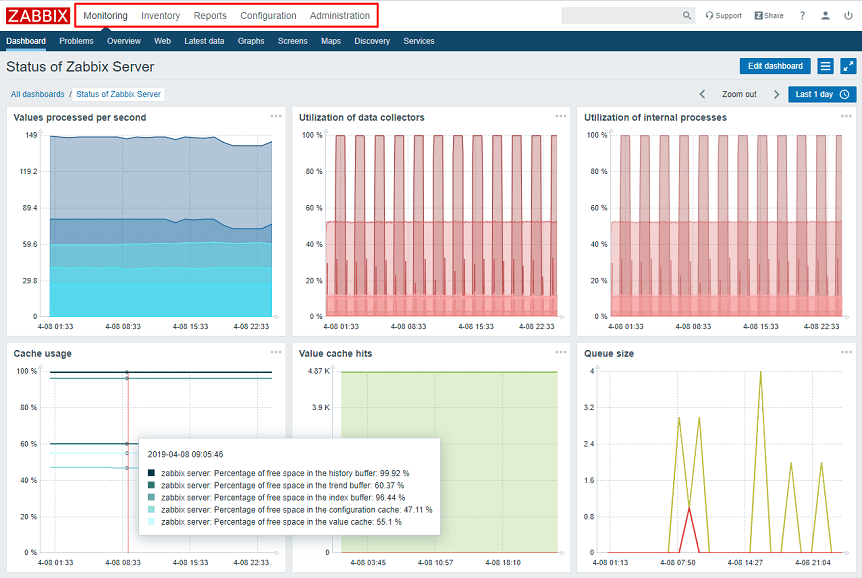
Прокси используется еще в нескольких случаях — если агенты находятся далеко друг от друга или ограничены локальной сетью. Это сказывается на доступности агентов и величине пингов.

Zabbix прокси функционирует как демон. Для его использования обязательно наличие отдельной базы данных.

Особенности веб-интерфейса

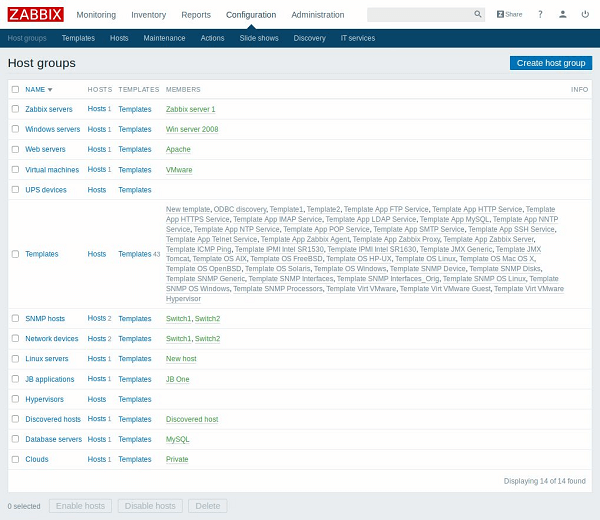
Система мониторинга Zabbix располагает удобным веб-интерфейсом, в котором сгруппированы элементы управления. Консоль предусматривает просмотр собранных данных, их настройку. Для безопасности входа и работы осуществляется автоматическое отсоединение через 30 минут пользовательского бездействия.

На главном экране всегда представлена информация о состоянии узлов сети и триггеров.

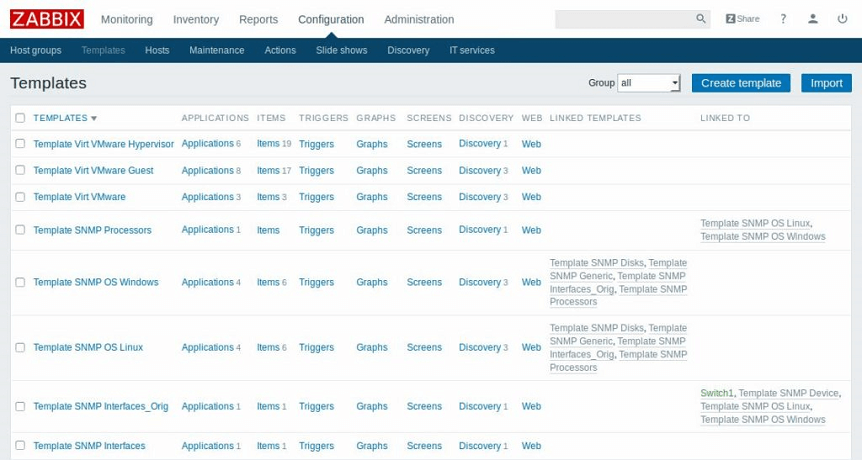


Пользователю доступны пять функциональных разделов, включая Monitoring («Мониторинг»), Inventory («Инвентарные данные»), Reports («Отчеты»), Configuration («Конфигурация») и Administration («Администрирование»).

В разделе «Конфигурации» можно найти группы хостов. По каждому элементу списка можно посмотреть более подробную информацию, например, последние события и графики данных.

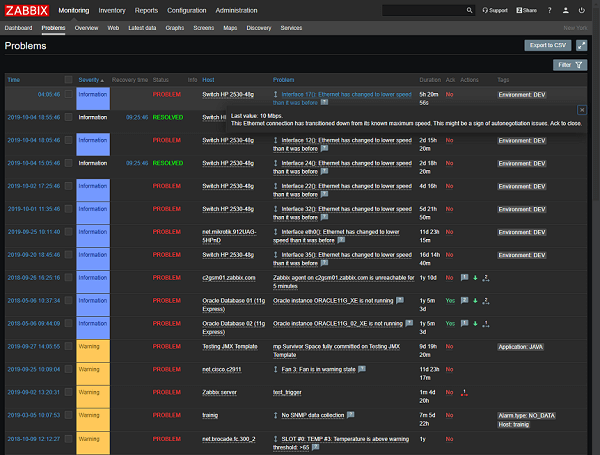


Управлять шаблонами, доступными администратору, можно в соответствующем подразделе — Templates («Шаблоны»).



Версия 4.4

Узнать версию [установленного Zabbix сервера](https://eternalhost.net/base/vps-vds/zabbix-ustanovka-nastroyka) можно во время запуске в файле-протоколе.



Основные нововведения в Zabbix 4.4

* Новый Zabbix Agent (zabbix\_agent2) создан на языке Go.
* Опции вывода графиков данных.
* Внешние уведомления, система отслеживания ошибок.
* Официальная поддержка TimescaleDB.
* База знаний для триггеров и элементов данных.
* Группировка данных и гистограммы.
* Официальная поддержка платформ, теперь Zabbix работает с SUSE Linux Enterprise Server 15, Debian 10, Raspbian 10, Mac OS/X, RHEL 8, MSI for Windows Agent и др.

Заключение

Zabbix по праву считается одним из самых продвинутых [инструментов для удалённого мониторинга](https://eternalhost.net/blog/sistemnoe-administrirovanie/sistemy-monitoringa-serverov) аппаратных и программных ресурсов. Система с успехом позволяет решать задачи по отслеживанию сетевой активности и работоспособности серверов, а также предупреждать о потенциально опасных ситуациях. Благодаря встроенным механизмам анализа и прогнозирования, Zabbix может стать основой для создания полноценной стратегии эффективного использования IT-инфрастуктуры в компаниях любого масштаба.