# Лабораторная работа №8

# Мониторинг сети с помощью программ Zabbix и Wireshark

## Теоретическая часть

Система мониторинга сервера — специальное программное обеспечение, предназначенное для непрерывного контроля и анализа работы сервера. Она позволяет отслеживать основные параметры сервера: загрузка процессора, использование памяти, объем дискового пространства, сетевая активность и другие ключевые показатели.

В простейшем случае, при работе на одной системе, например на смартфоне, возможно использовать одну из множества утилит, которая снимет значения нагрузки процессора, занимаемого объема ОЗУ или же свободного места на носителе (CPU-Z и другие). Если мы работаем на компьютере с ОС Linux, то возможно использовать консольные утилиты htop, netstat и др. Каждая из утилит применяется для мониторинга отдельных параметров - такой подход не применим для централизованного сбора информации. Поэтому необходимо использовать специализированные системы мониторинга для анализа системы.

С необходимостью систем мониторинга люди столкнулись при анализе производительности LAN сетей в рамках одного офиса. В них использовался специальный протокол SNMP (Simple Network Management Protocol), который позволял настраивать двусторонний или односторонний доступ к различным сетевым устройствам: от роутеров и коммутаторов до принтеров. На его основе настраивались такие инструменты как Big Brother и nmon, которые предоставляли информацию о сетевых нагрузках и событиях в сети, используя для этого сетевое соединение.

С появлением веб-сайтов и интернет-сервисов стало необходимо модифицировать системы. В результате были разработаны веб-ориентированные легко масштабируемые инструменты, поддерживающие интернет-протоколы. Среди популярных open-source инструментов начала 2000-х — Zabbix, Nagios и Cacti. В дальнешем весь рост систем мониторинга происходил в сторону увеличения количества метрик или же разнообразия объектов для снятия этих метрик.

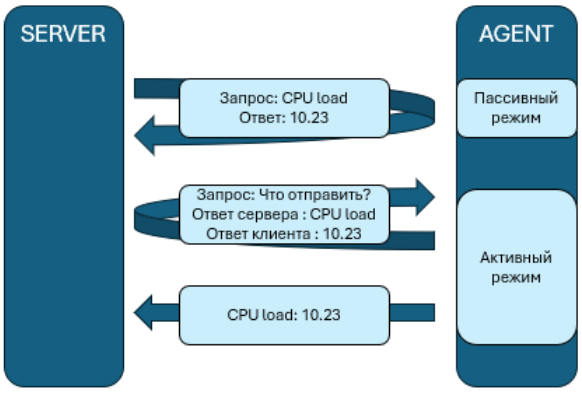
На сегодня основными системами мониторинга являются:

- **PingInfoView, Pingdom** – простые системы, использующие ping для проверки доступности узлов в сети. Позволяют с интервалом проверять состояние узлов и отображать доступность в режиме реального времени.
- **Zabbix** – мощная система с поддержкой агентов и безагентного сбора данных (SNMP, IPMI, HTTP и т.д.), средствами анализа данных, уведомлениями и веб-интерфейсом.

- **PRTG Network Monitor** – система с широким набором сенсоров для сбора данных (SNMP, WMI, ICMP и пр.), анализом данных, уведомлениями и удобным интерфейсом. В отличие от многих других конкурентов имеет ограничение в 100 метрик (сенсоров).
- **Graphite** — это бесплатный инструмент с открытым исходным кодом (FOSS), который строит графики числовых временных рядов, таких как производительность компьютерных систем. Graphite собирает, хранит и отображает данные временных рядов в реальном времени.
- **Prometheus** – система, в основе которой лежит база данных временных рядов (Time series database, TSDB). Поддерживает экспортеры и PushGateway (возможность отсылать метрики из кастомных скриптов и программ), уведомления и интеграцию с Grafana для визуализации.

Существуют две основные модели мониторинга:

- Push-модель – сервер мониторинга ожидает подключений от агентов для получения метрик. Модель обычно используется для мониторинга больших систем, где количество устройств может быть слишком большим для ручного сбора данных.
- Pull-модель – сервер мониторинга сам подключается к агентам мониторинга и забирает данные.




Современные системы, такие как Zabbix и Prometheus работают как Push и Pull модель. Обе системы являются очень популярными на рынке.

## Wireshark

Wireshark — программа-анализатор трафика для компьютерных сетей Ethernet и некоторых других. Имеет графический пользовательский интерфейс.
Для установки пакета необходимо воспользоваться программой:

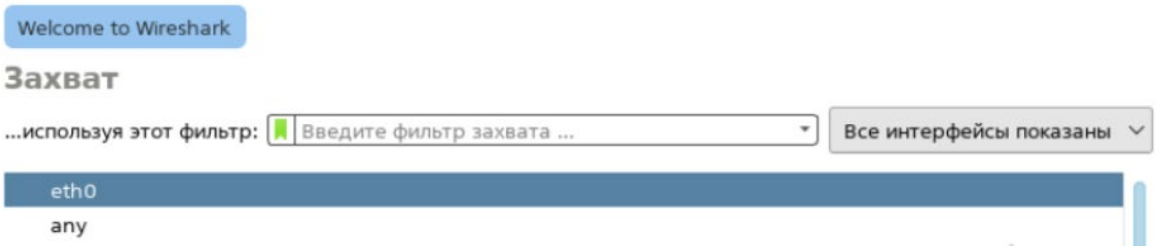
```
sudo apt install wireshark
```

При установке возможно указать, чтобы пользователи не обладающие правами суперпользователя могли производить захват трафика.

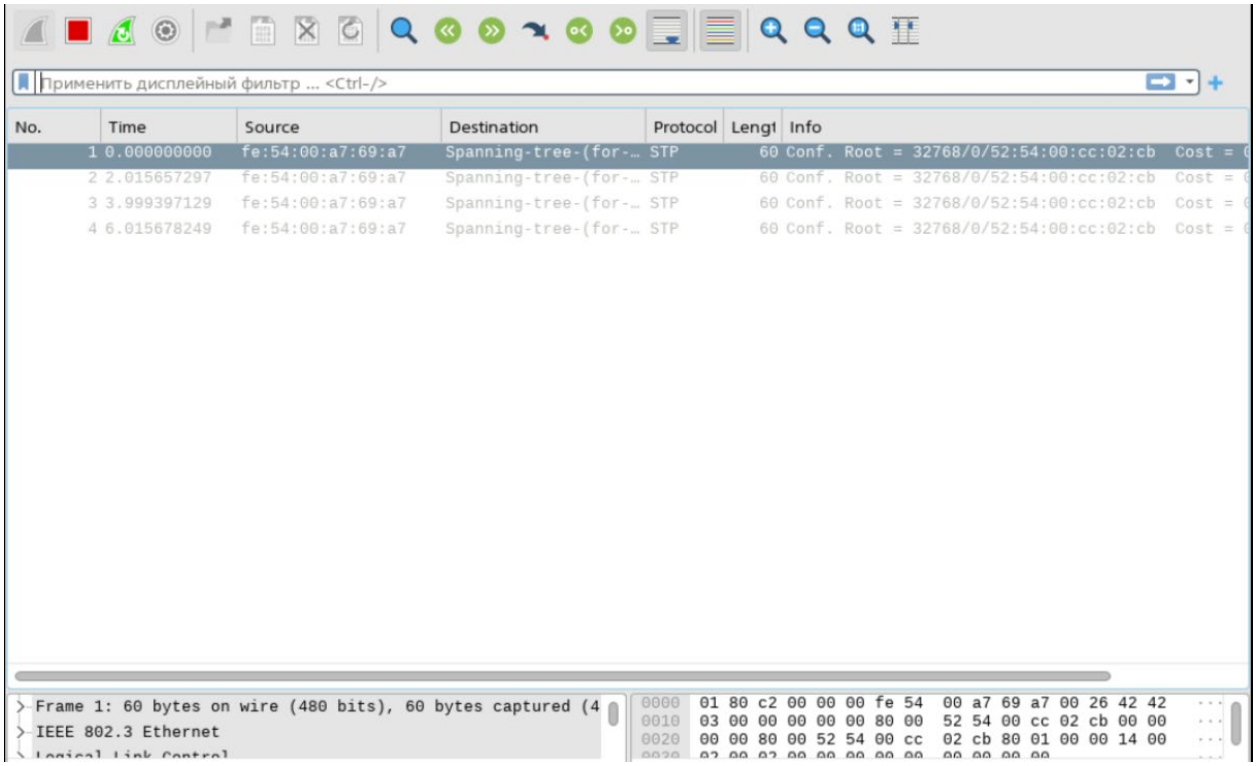
Для запуска программы воспользуйтесь командой:

```
sudo wireshark
```

После запуска программы выберите интерфейс, который вы будете прослушивать:




Далее открывается рабочее окно, в котором возможно наблюдать весь сетевой трафик, проходящий через указанный интерфейс.



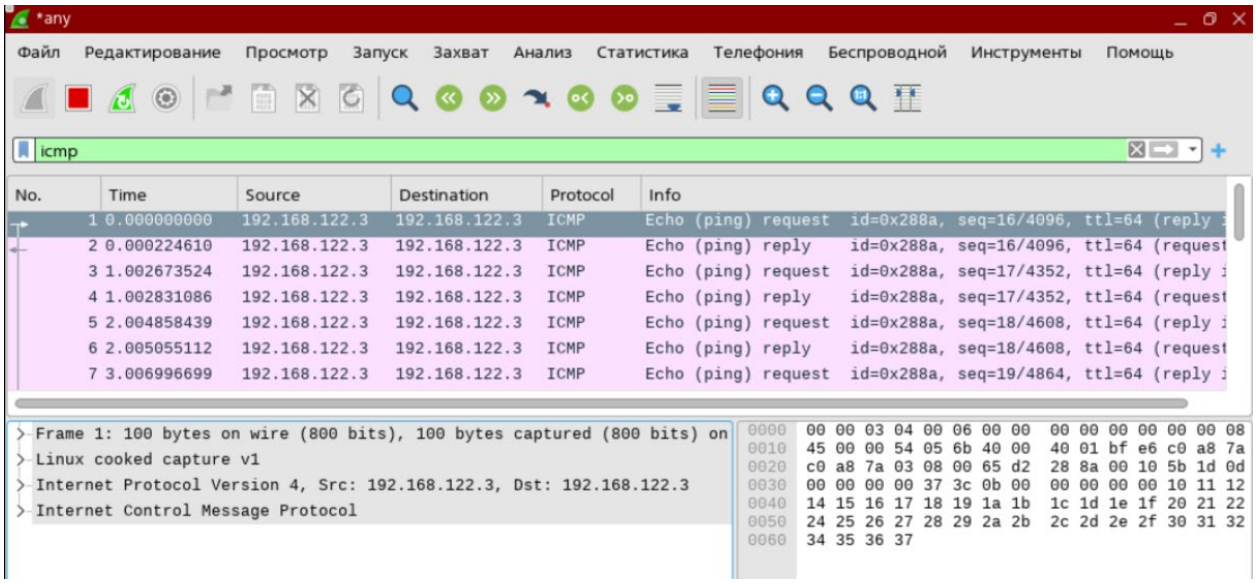

Для отбора необходимых строк возможно использовать фильтры.

| Название фильтра | Комментарий | Пример |
|---|---|---|
| <Название протокола> | Фильтрация пакетов с определенных, | icmp |
| ip.src==<ip адрес> | Фильтрация по адресу источника сообщений | ip.src==192.168.122.2 |
| ip.dst==<ip адрес> | Фильтрация по адресу назначения сообщений | ip.dst==192.168.122.3 |
| <Название протокола>.port == <номер порта> | Фильтрация по порту | tcp.port == 80 |

С помощью символов || фильтры возможно объединять

Например: icmp || tcp || udp

Для анализа пакета возможно воспользоваться окошками, расположенными ниже:

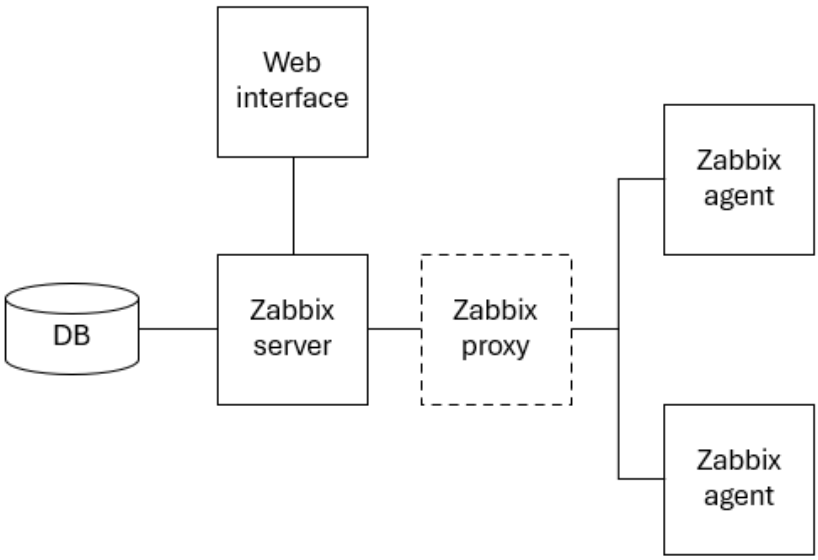



В правом окошке представлен пакет в «сыром» виде, как последовательность байт. В левом окне выводится его расшифровка.

Более подробно по программе wireshark возможно прочитать в источнике 1.

**Zabbix**

Zabbix — свободная система мониторинга статусов разнообразных сервисов компьютерной сети, серверов и сетевого оборудования, написанная Алексеем Владышевым. Для хранения данных используется MySQL, PostgreSQL, SQLite или Oracle Database, веб-интерфейс написан на PHP.




**Zabbix-сервер** — ядро системы, которое дистанционно контролирует сетевые сервисы и которое является хранилищем, содержащим все конфигурационные, статистические и оперативные данные. Он является тем

субъектом в программном обеспечении Zabbix, который оповещает администраторов о проблемах с контролируемым оборудованием.

**Zabbix-прокси** собирает данные о производительности и доступности от имени Zabbix-сервера. Все собранные данные заносятся в буфер на локальном уровне и передаются Zabbix-серверу, к которому принадлежит прокси-сервер. Zabbix-прокси является идеальным решением для дистанционного контроля филиалов и других точек, в том числе сетей, не имеющих местных администраторов. Он может быть также использован для распределения нагрузки одного Zabbix-сервера. В этом случае прокси только собирает данные, благодаря чему на сервер ложатся меньшие нагрузки на ЦПУ и устройства ввода/вывода.

**Zabbix-агент** — программа контроля локальных ресурсов и приложений (таких как накопители, оперативная память, статистика процессора и т. д.) на сетевых системах, эти системы должны работать с запущенным Zabbix-агентом.
Zabbix-агенты являются чрезвычайно эффективными из-за использования специфических системных вызовов для сбора информации и подготовки статистики.

**Веб-интерфейс** — часть Zabbix-сервера, и, как правило (но не обязательно), запускается на том же физическом узле, что и Zabbix-сервер. Работает на PHP, требует веб-сервер (например nginx, Apache httpd).

## Установка и настройка Zabbix

Для корректной работы сервера необходимо настроить веб интерфейс(apache2 + php), БД и доступ к ней, Locale для поддержки английского языка.

Первым шагом требуется установить необходимые пакеты:

```
sudo apt install zabbix-server-pgsql zabbix-frontend-php php-pgsql
```

Перед началом работы необходимо назначить метки безопасности служебным пользователям postgres и zabbix:

```
sudo pdpl-user -i 63 postgres
sudo pdpl-user -l 0:0 zabbix
sudo usermod -a -G shadow postgres
```

Предоставить служебному пользователю postgres право чтения базы данных меток безопасности локальных пользователей:

```
sudo setfacl -d -m u:postgres:r /etc/parsec/{macdb,capdb}
sudo setfacl -R -m u:postgres:r /etc/parsec/{macdb,capdb}
sudo setfacl -m     u:postgres:rx /etc/parsec/{macdb,capdb}
```

### Настройка apache2

В файле /etc/php/*/apache2/php.ini раскоментировать и дописать

```
[Date]
date.timezone = Europe/Moscow
```

В файле /etc/apache2/apache2.conf  установить “AstraMode off”
Этими действиями мы установили часовой пояс на веб сервере и отключили astra security mode

Необходимо проверить, что DNS работает корректно и производиться преобразование доменных имен в IP адреса.

**Настройка СУБД PostgreSQL**

Отредактировать конфигурационный файл /etc/postgresql/*/main/pg_hba.conf:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD
local   zabbix          zabbix                                  trust
# IPv4 local connections:
host    zabbix          zabbix          127.0.0.1/32            trust
```

Перезапустить кластер:

```
sudo systemctl restart postgresql
```

Создать пользователя и базу zabbix:

```
sudo -u postgres psql
```

```
CREATE DATABASE ZABBIX;
CREATE USER zabbix WITH ENCRYPTED PASSWORD '12345678';
GRANT ALL ON DATABASE zabbix to zabbix;
ALTER DATABASE zabbix OWNER TO zabbix;
```

Далее копируется шаблон базы данных

```
sudo zcat /usr/share/zabbix-server-pgsql/{schema,images,data}.sql.gz | psql -h localhost zabbix zabbix
sudo a2enconf zabbix-frontend-php
sudo systemctl reload apache2
```

**Настройка веб интерфейса**

Скопировать в файл /etc/zabbix/zabbix.conf.php

```
sudo cp /usr/share/zabbix/conf/zabbix.conf.php.example /etc/zabbix/zabbix.conf.php
```

Установить права доступа к созданному файлу:

```
sudo chown www-data:www-data /etc/zabbix/zabbix.conf.php
```

В файле /etc/zabbix/zabbix.conf.php задать значения переменных TYPE (тип используемой СУБД) и PASSWORD (пароль пользователя zabbix СУБД):

```
$DB['TYPE'] = 'POSTGRESQL';
$DB['PASSWORD'] = '<12345678>';
```

Перезапустить службу apache2:

```
sudo systemctl reload apache2
```

В конфигурационном файле /etc/zabbix/zabbix_server.conf раскомментировать строку, задающую пароль доступа к БД zabbix, и указать там пароль:

```
DBPassword=12345678
```

Разрешить автоматический запуск службы zabbix при перезагрузке ОС и запустить службу zabbix:

```
sudo systemctl enable zabbix-server
sudo systemctl start zabbix-server
```

**Настройка locale**

В файле /etc/locale.gen раскомментировать строку en_US.UTF-8. Выполнить locale-gen. Открыть WEB-страницу zabbix в WEB-браузере. Имя для входа: Admin (с заглавной буквы), пароль для входа: zabbix

```
firefox localhost/zabbix
```

**Настройка Zabbix-agent**

Устанавливаем на клиенте службу

```
sudo apt install zabbix-agent
```

В конфигурационном файле /etc/zabbix/zabbix_agentd.conf меняем ip сервера на актуальный(который меняли в /etc/hosts), имя хоста устанавливаем как имя клиента.

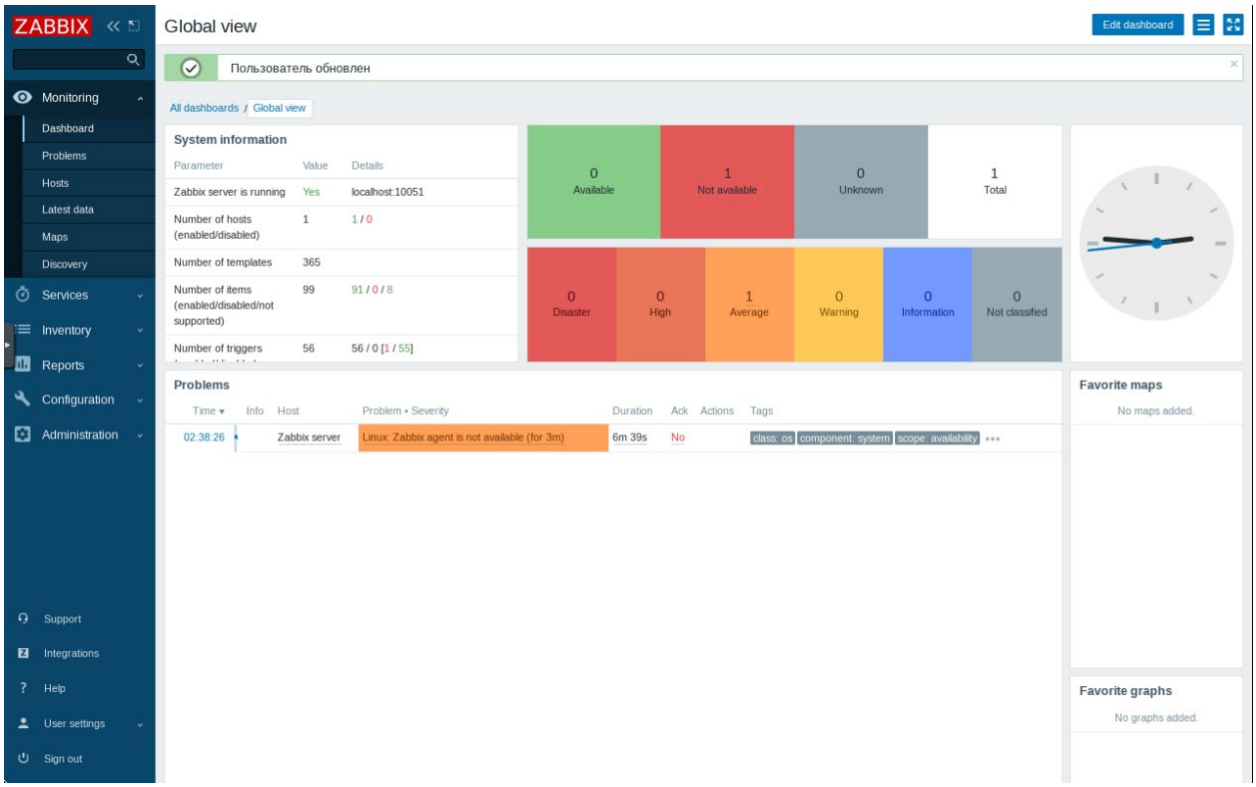
```
Server=192.168.122.2
```

Перезапустим агент

```
sudo systemctl restart zabbix-agent
```

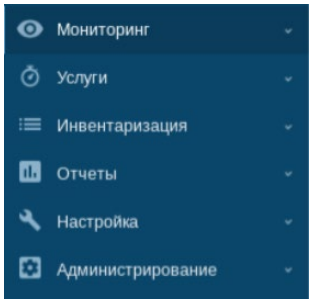
***Настройка***

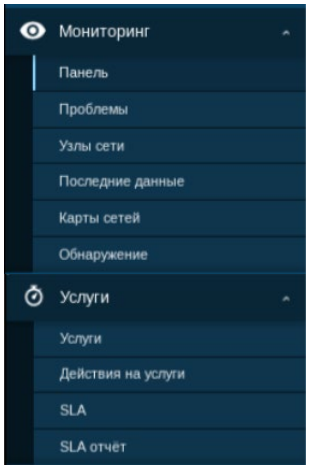
Web-страница Zabbix выглядит следующим образом:




Перед началом работы выберем язык приложения, для этого перейдите во вкладку User settings -> Profile. В секции Language выберите русский язык (Russian ru_RU). Нажмите на кнопку Update.

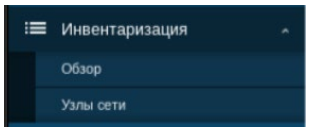
Рассмотрим боковую панель программы:



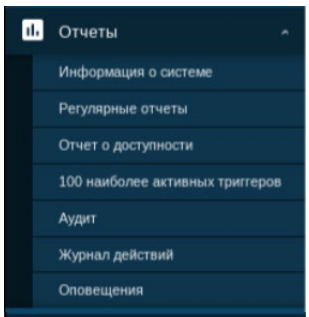



Вкладка мониторинг содержат подробную информацию о системе, хостах. Здесь можно найти ошибки и заглянуть в логи. Во вкладке hosts – все подключенные серверы, с подробными данными и графиками о состоянии их метрик

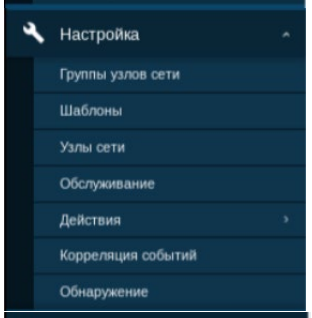
Услуги – возможность добавления услуг бизнес уровня для мониторинга



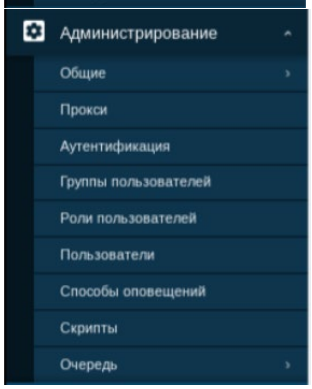

Меню Инвентаризация включает разделы обеспечивающие обзор инвентарных данных узлов сети по выбранному параметру.




Меню Отчеты включает в себя несколько разделов, которые содержат различные предустановленные и пользовательские отчеты, направленные на обзор таких параметров как информации о системе, триггеров и собранных данных



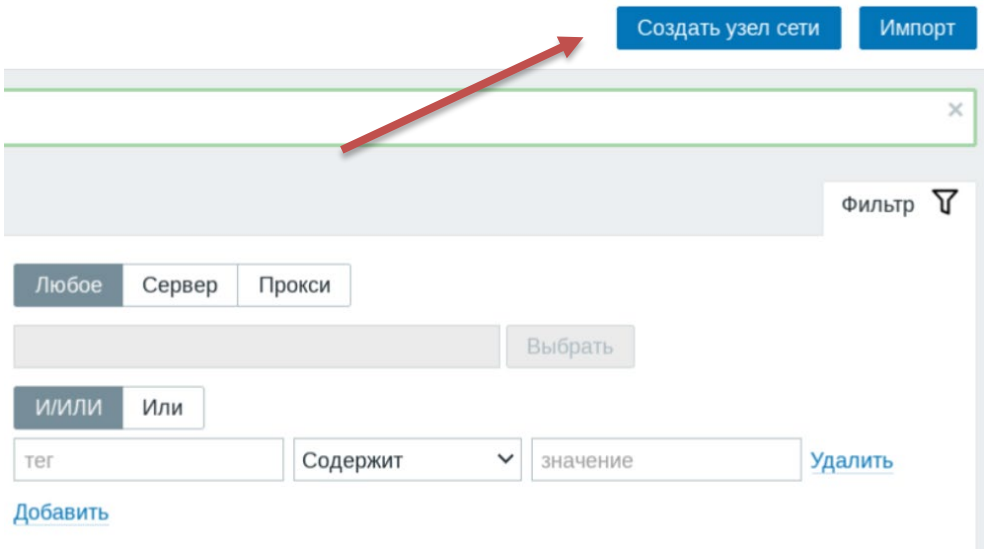

Меню настроек, здесь можно выбрать шаблоны, добавить хосты, настроить события и тд.




Меню Администрирование используется в Zabbix для административных функций. Это меню доступно только пользователям с типом Super Administrators.

**Добавление хоста и метрик**

Для добавления узлов перейдем во вкладку Настройка -> Узлы сети. В правом верхнем углу нажмем кнопку «Создать узел сети»

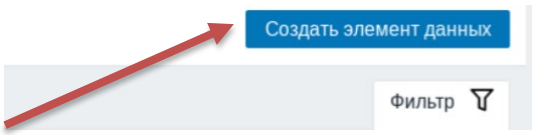



Заполним основные параметры узла.

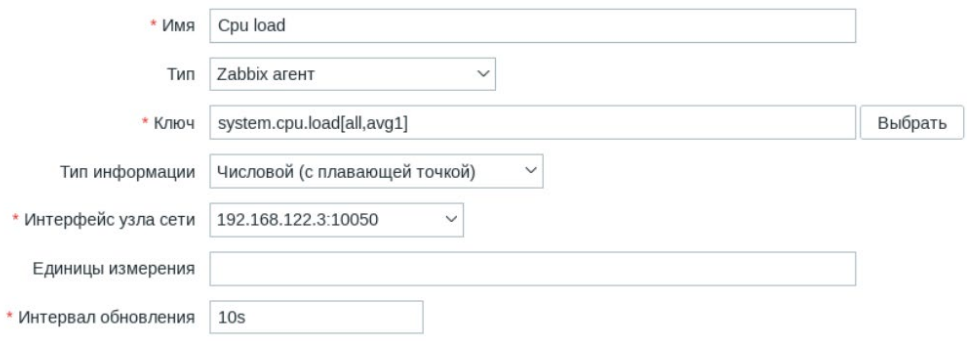
- Имя узла: node1
- Группы: Templates/Operating systems
- Интерфейсы: Агент, IP - <IP адрес клиента>


После добавления узла он будет отображен на нижней панели, однако параметр «Доступность» будет окрашен в серый -> ZBX.

Для того, чтобы хост был доступен необходимо добавить на клиент шаблон. Выберете столбец «Элементы данных» -> создать элемент данных.




В качестве примера установим элемент данных, получающий значение загрузки процессора. Для этого установите следующие параметры:



Обратите внимание, что в поле «Ключ» вы можете выбрать разнообразные метрики. В качестве периода хранения истории и динамики изменения укажите значение «7d». После добавления элемента данных через некоторое время поле доступность окрасится в зеленый -> ZBX.

Более подробно о метриках и их параметрах можно прочитать по ссылке: https://www.zabbix.com/documentation/5.2/ru/manual/config/items/itemtypes/zabbix_agent

Для удобства настройки сбора данных возможно использовать утилиту zabbix_get командной строки. Для этого укажите IP адрес клиента после параметра -s и команду с запросом метрики после ключа -k.

```
zabbix_get -s 192.168.122.4 -k system.cpu.load
```

Команда вернет значение нагрузки на процессор.

**Настройка панели мониторинга.**

Для отображения созданной метрики в виде графика перейдите во вкладку Мониторинг -> Панель.

В правой части экрана нажмите на кнопку «Изменить панель» (Если кнопка не видна – возможно установлен слишком большой масштаб экрана – уменьшите его, зажав ctrl и прокрутив колесико на мышке).


Далее перетаскивая и изменяя окна, возможно настроить панель в удобный для вас вид.

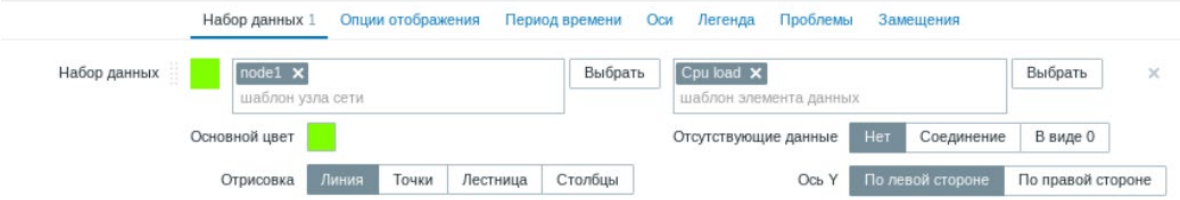
Нажав кнопкой мыши по пустому месту на экране панели вызовите диалоговое окно «Добавить виджет».

Установите следующие параметры:

Имя: CPU load

Интервал обновления: 10 секунд

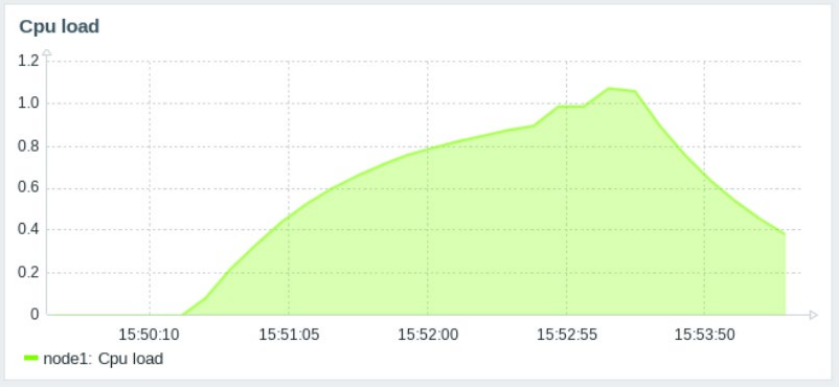
Набор данных: Шаблон узла сети – node1, шаблон элемента данных – CPU load




После добавления элемента график отобразится на панели. Сохраните изменения, нажав на кнопку «Сохранить изменения» в верхней части окна. Для того, чтобы убедиться в работоспособности системы запустите на клиенте скрипт, нагружающий CPU:

```
dd if=/dev/urandom | bzip2 -9 > /dev/null
```

Значение графика будет показывать нагрузку

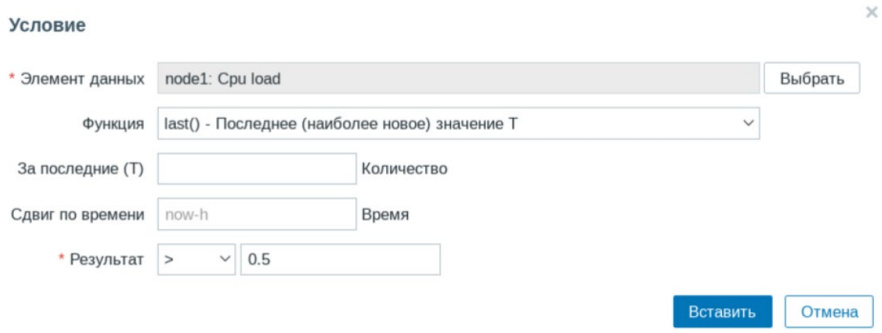



**Настройка триггеров.**

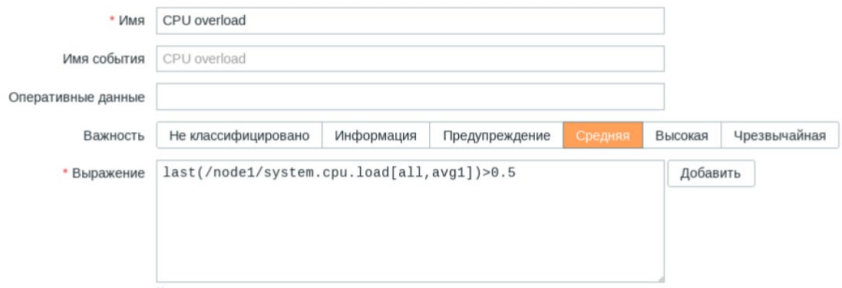
Для автоматического оповещения пользователей в случае нештатного события настраиваются триггеры.

Для этого перейдите во вкладку Настройки -> Узлы сети -> Триггеры. В правом верхнем углу нажмите на кнопку «Создать триггер». Укажите следующие параметры:

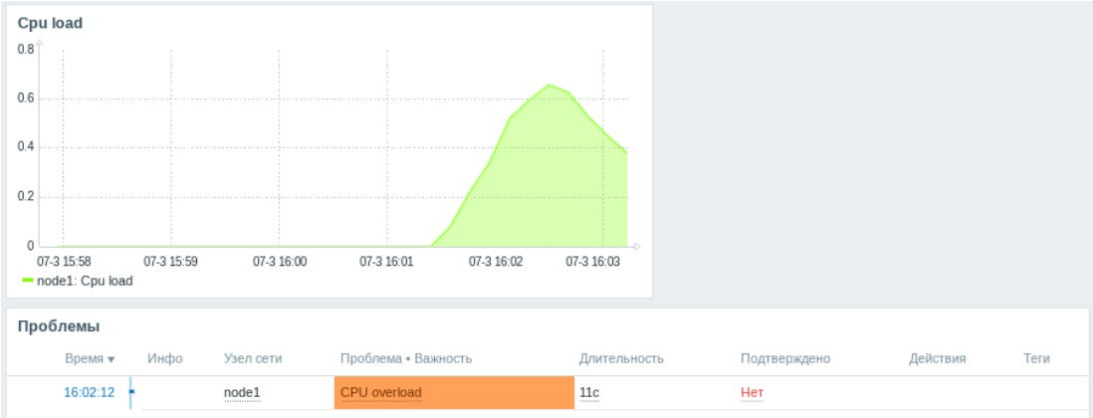
- Имя: CPU overload
- Важность: Средняя
- Выражение: -> Добавить
    - Элемент данных: node1, CPU load
    - Результат: > 0.5




Параметры будут выглядеть приблизительно следующим образом:



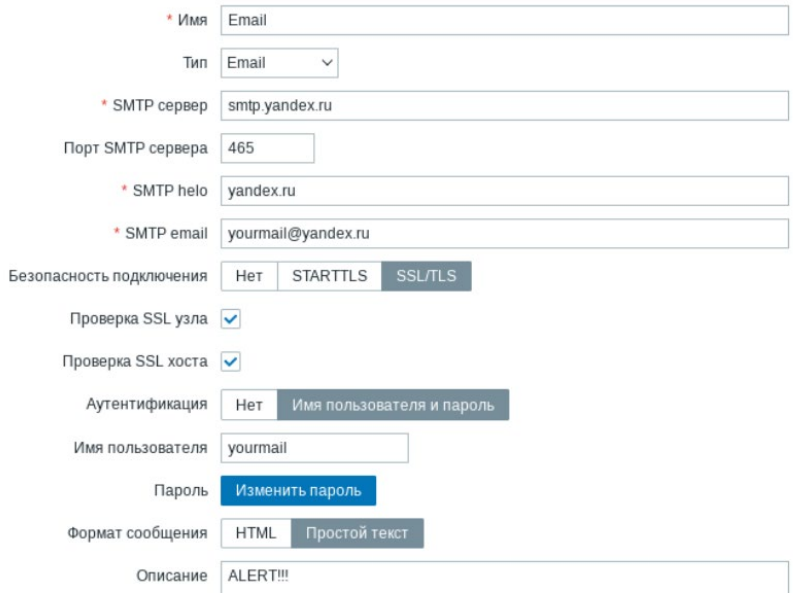

Добавьте триггер и вернитесь на вкладку с мониторингом. Снова нагрузите CPU клиента и убедитесь, что в разделе «Проблемы» появилось сообщение о перегрузке.



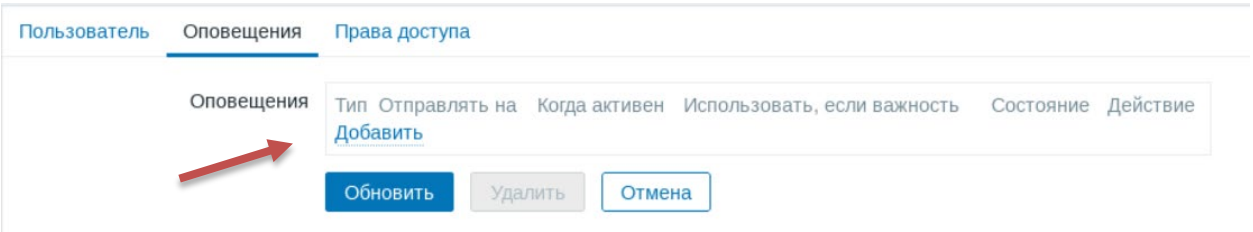

**Настройка передачи сообщений на электронную почту**

Для того, чтобы оповещения приходили на электронную почту, необходимо провести следующие настройки: Администрирование -> Способы оповещения

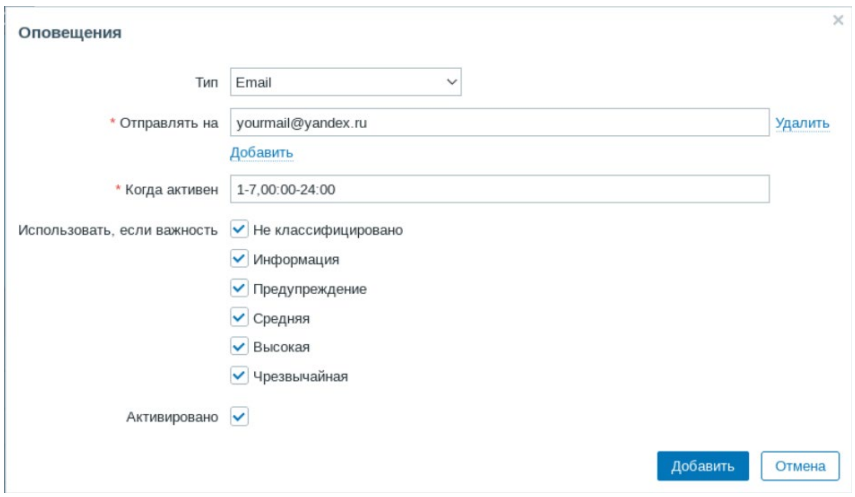
Выберите пункт «Email» и укажите следующие параметры:




Далее необходимо добавить почту пользователю Admin (Администрирование -> Пользователи). Для этого нажмите на имя пользователя, перейдите во вкладку «Оповещения» и добавьте его.



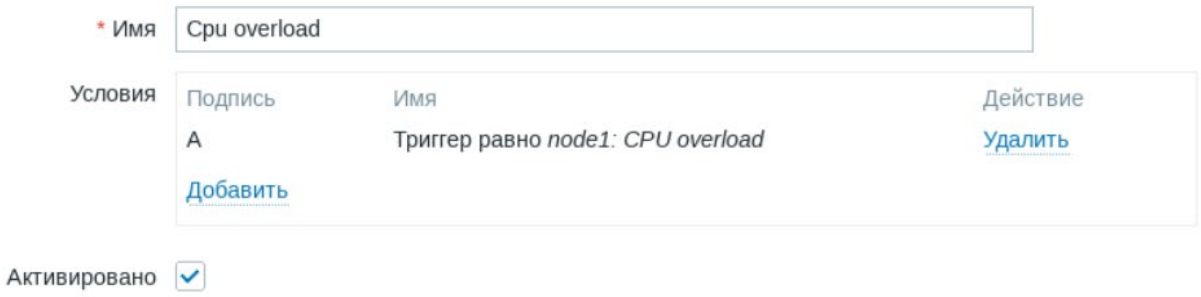

Далее установите следующие параметры:




После добавления нажмите на кнопку «Обновить».

Для прикрепления триггера к оповещению перейдите во вкладку Настройка -> Действия -> Действия триггеров. Создайте новое действие (кнопка в правом верхнем углу). Установите следующие параметры:

- Действия:
  - Имя: CPU overload
  - Условия: Добавить -> Триггер -> Выбрать -> CPU overload
- Операции:
  - Операции: Добавить -> Отправка пользователям (Добавить) -> Admin
  - Отправка только через Email




Нагрузите CPU на клиенте и убедитесь, что письмо с оповещением пришло на почту.

## Практическая работа

**Задание 1.**

*Работа с Wireshark. В рамках задания вам будет необходимо установить программу на виртуальной машине workstation. Далее, обращаясь к разным версиям снимков данных, которые вы делали после каждой из лабораторных работ проведите следующие эксперименты.*

1. Запустите программу wireshark
2. Установите фильтр, выбрав ip-адреса клиента, сервера и сетевого моста.
3. Выполните следующие команды и проанализируйте результат их работы в wireshark:
    - ping с сервера на клиент
    - ping на адрес, расположенный вне сети
    - ping с клиента на сервер по доменному имени сервера
    - ping с клиента на сервер по доменному имени сервера, не включенного в список DNS
    - открытие веб-страницы из предыдущей лабораторной работы на клиенте
    - обновление адреса DHCP на клиенте с помощью команд dhclient -r; dhclient
    - обмен TCP сообщениями между клиентом и сервером (запуск файлов tcpserver.py, tcpclient.py, исходные коды в приложении)
    - обмен UDP сообщениями между клиентом и сервером (запуск файлов udpserver.py, udpclient.py, исходные коды в приложении)

**Задание 2.**

*Основы работы с zabbix*

1. Установите и настройте систему мониторинга Zabbix на сервер. В качестве узлов для мониторинга выберете node1 и node2.
2. Добавьте мониторинг следующих параметров:
    - Утилизация CPU в процентах за последние 5 минут;
    - Количество пользователей, находящихся в системе;
    - Объем файла с логами (/var/log/syslog)
    - Любые два параметра на ваш выбор
3. Отобразите указанные значения с устройств node1, node2 в виде графиков на главной панели. Произведите действия, которые приведут к изменению этих значений.

**Задание 3.**

1. Добавьте триггер, срабатывающий в случае превышения объема файла с логами заданного значения (подберите его таким образом, чтобы можно было продемонстрировать действие преподавателю).
2. Настройте отправку уведомлений на вашу электронную почту
3. Опционально:
    - Настройте действие таким образом, чтобы после превышения размера файла с логами он автоматически очищался

- Настройте отправку уведомлений в телеграмм

**Список литературы:**

[1] https://wireshark.wiki

**Приложение:**

*tcpclient.py*

```
import socket

HOST = "127.0.0.1"
PORT = 65432
bufferSize = 1024

TCPClientSocket=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
TCPClientSocket.connect((HOST, PORT))
TCPClientSocket.sendall(b"Hello, world")
data = TCPClientSocket.recv(bufferSize)
print(f"Received {data!r}")
```

*tcpserver.py*

```
import socket

HOST = "127.0.0.1"
PORT = 65432
bufferSize  = 1024

TCPServerSocket=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
TCPServerSocket.bind((HOST, PORT))
TCPServerSocket.listen()

TCPAnswer=TCPServerSocket.accept()
connection = TCPAnswer[0]
address = TCPAnswer[1]
print(f"Connected by {address}")
data = connection.recv(bufferSize)
connection.sendall(data)
```

*udpclient.py*

```
import socket

msgFromServer       = "Hello world!"
bytesToSend         = str.encode(msgFromServer)

serverAddressPort   = ("127.0.0.1", 65432)
bufferSize          = 1024

UDPClientSocket=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
UDPClientSocket.sendto(bytesToSend, serverAddressPort)
data = UDPClientSocket.recv(bufferSize)
print(f"Received {data!r}")
```

*udpserver.py*

```
import socket

HOST = "127.0.0.1"
PORT = 65432
bufferSize  = 1024

UDPServerSocket=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
UDPServerSocket.bind((HOST, PORT))

UDPAnswer=UDPServerSocket.recvfrom(bufferSize)
connection = UDPAnswer[0]
address = UDPAnswer[1]
print(f"Connected by {address}")
UDPServerSocket.sendto(connection,address)
```