

Агентный мониторинг ОС Windows

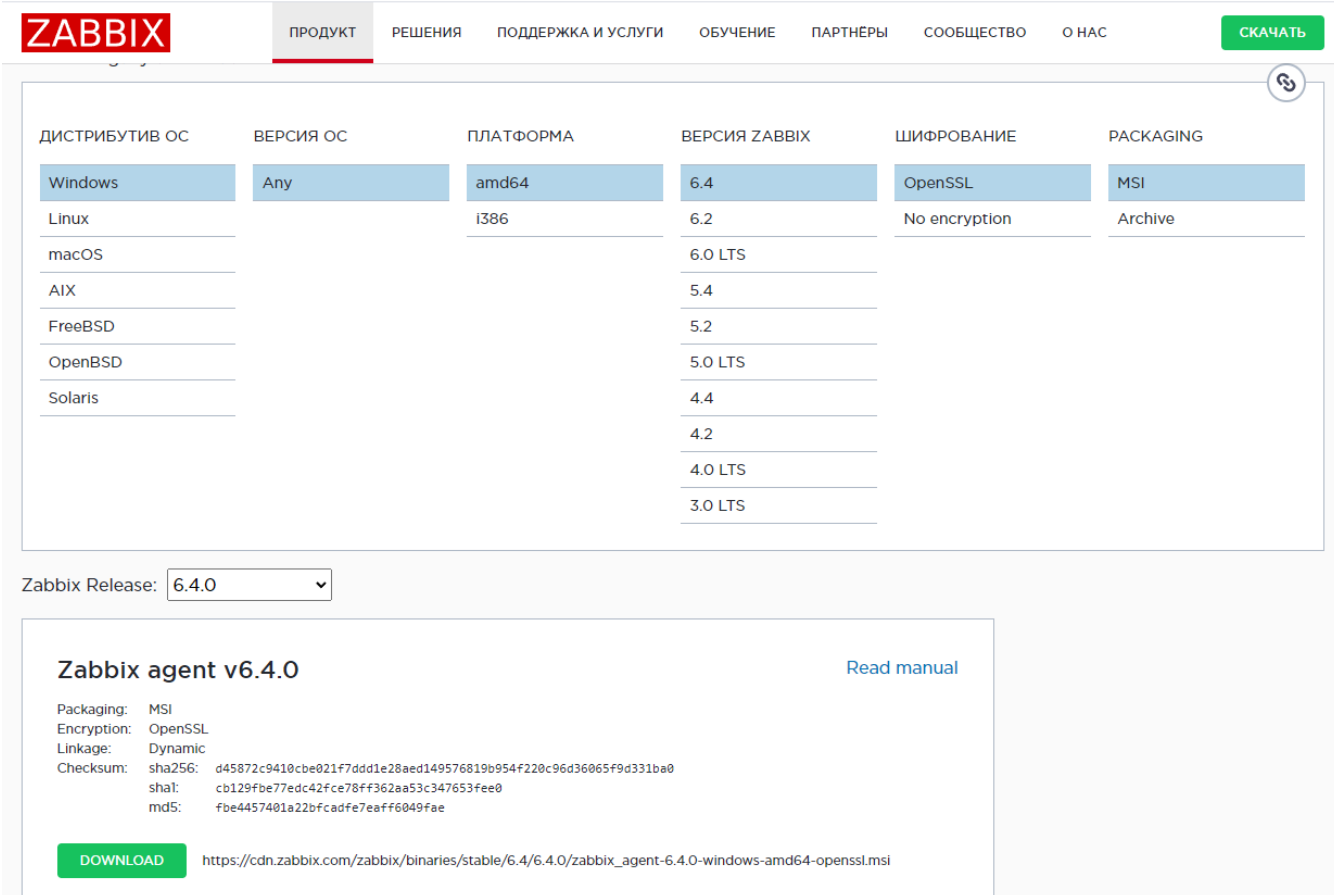
В качестве объекта мониторинга будет использован ОС Windows установленный на виртуальной машине и введенный в домен

Шаг 1. Настройка Zabbix агента на Windows

Как установить Агент на Windows системах можно прочитать в документации на оф. Сайте zabbix [Zabbix Агент](#)

Сначала необходимо скачать исходный код агента (с официального сайта проекта – Рис. 1):

Ссылка для исходного кода агента: [Zabbix Cloud Images and Appliances](#)



The screenshot shows the Zabbix website's download page. The top navigation bar includes links for ПРОДУКТ, РЕШЕНИЯ, ПОДДЕРЖКА И УСЛУГИ, ОБУЧЕНИЕ, ПАРТНЕРЫ, СООБЩЕСТВО, and О НАС, along with a green button labeled СКАЧАТЬ. The main content area features a table with columns for ДИСТРИБУТИВ ОС, ВЕРСИЯ ОС, ПЛАТФОРМА, ВЕРСИЯ ZABBIX, ШИФРОВАНИЕ, and PACKAGING. The 'Windows' distribution is selected, showing version 6.4 on the amd64 platform. Below the table, a dropdown menu shows 'Zabbix Release: 6.4.0'. The bottom section displays 'Zabbix agent v6.4.0' with a 'Read manual' link, packaging details (MSI, OpenSSL, Dynamic linkage), checksums (sha256, sha1, md5), and a green 'DOWNLOAD' button with the URL: https://cdn.zabbix.com/zabbix/binaries/stable/6.4/6.4.0/zabbix_agent-6.4.0-windows-amd64-openssl.msi.

ДИСТРИБУТИВ ОС	ВЕРСИЯ ОС	ПЛАТФОРМА	ВЕРСИЯ ZABBIX	ШИФРОВАНИЕ	PACKAGING
Windows	Any	amd64	6.4	OpenSSL	MSI
Linux		i386	6.2	No encryption	Archive
macOS			6.0 LTS		
AIX			5.4		
FreeBSD			5.2		
OpenBSD			5.0 LTS		
Solaris			4.4		
			4.2		
			4.0 LTS		
			3.0 LTS		

Zabbix Release: 6.4.0

Zabbix agent v6.4.0 [Read manual](#)

Packaging: MSI
Encryption: OpenSSL
Linkage: Dynamic
Checksum: sha256: d45872c9410cbe021f7ddd1e28aed149576819b954f220c96d36065f9d331ba0
 sha1: cb129fbe77edc42fce78ff362aa53c347653fee0
 md5: fbe4457401a22bfcadfe7eaff6049fae

DOWNLOAD https://cdn.zabbix.com/zabbix/binaries/stable/6.4/6.4.0/zabbix_agent-6.4.0-windows-amd64-openssl.msi


```
ListenPort=10050    # Name of log file – имя лог-файла

LogFile=c:\program files\zabbix\zabbix_agentd.log

# Maximum size of log file in MB. Set to 0 to disable automatic log rotation.

LogFileSize=10
```

```
Server=192.168.88.208
Hostname=host.windows.home
ListenPort=10050
LogFile=c:\program files\zabbix\zabbix_agentd.log
LogFileSize=10
```

Сам файл `zabbix_agentd.win.conf` нужно переименовать в `zabbix_agentd.conf`, файл `zabbix_agentd.log` создать вручную (самое простое – с помощью текстового редактора Notepad++). Результат должен соответствовать Рисунку 7.2:

Имя	Дата изменения	Тип
conf	10.09.2018 17:06	Папка с файлами
dev	10.09.2018 17:02	Папка с файлами
zabbix_agentd.exe	23.01.2018 17:47	Приложение
zabbix_agentd.log	10.09.2018 17:08	Файл "LOG"
zabbix_get.exe	23.01.2018 17:47	Приложение
zabbix_sender.exe	23.01.2018 17:47	Приложение

Рисунок 7.2 – Вид каталога Zabbix-агента

Запустите командную строку `cmd` от имени **Администратора** и выполните следующие команды (их можно скопировать и вставить, **проверьте соответствие имен файлов и путей к ним**):

Установка агента:

```
"C:\Program Files\zabbix\zabbix_agentd.exe" --config
```

```
"C:\Program Files\zabbix\conf\zabbix_agentd.conf" --install
```

Запуск службы:

```
"C:\Program Files\zabbix\zabbix_agentd.exe" --config
```

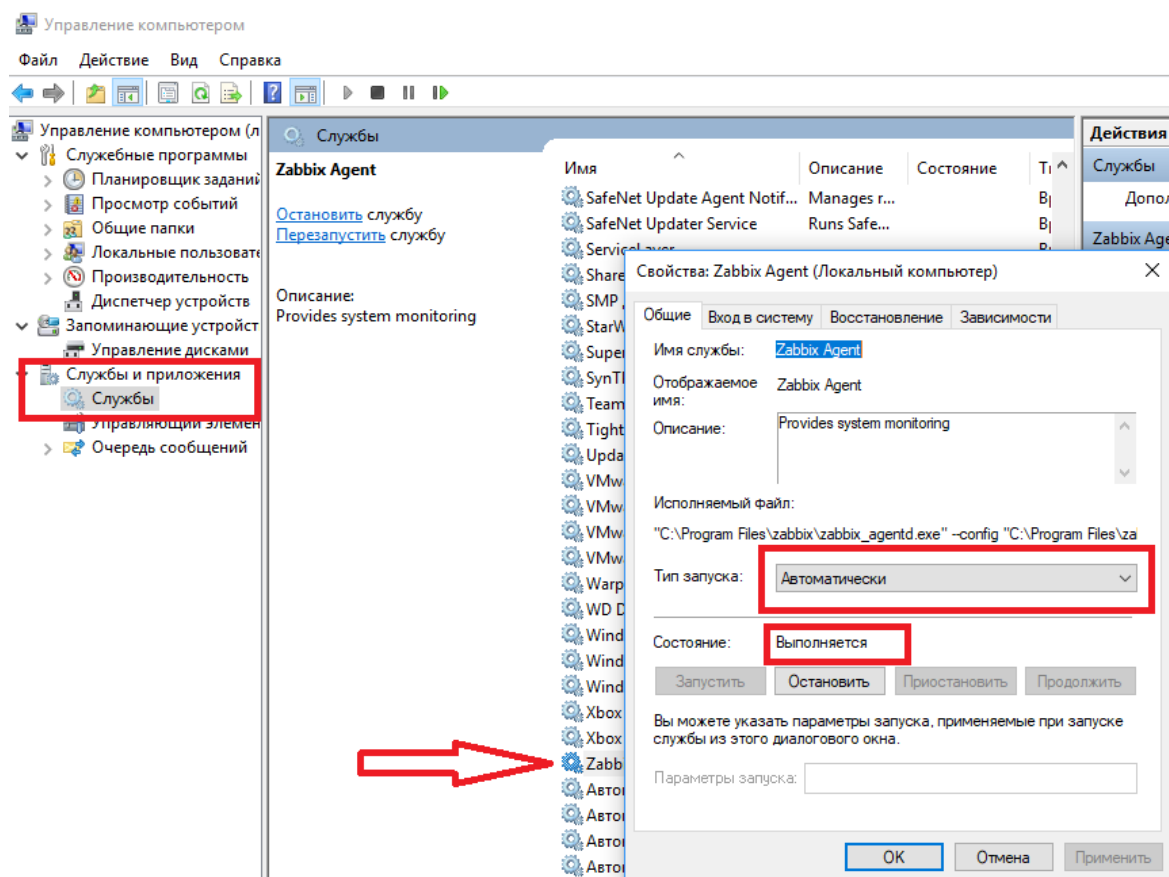
```
"C:\Program Files\zabbix\conf\zabbix_agentd.conf" --start
```

Просмотр доступных для мониторинга компонентов:

```
"c:\program files\zabbix\zabbix_agentd.exe" --config
```

```
"c:\program files\zabbix\conf\zabbix_agentd.conf" --print
```

После установки убедитесь, что служба будет автоматически запускаться (Рис. 3):



```
zabbix@ZABBIX:~$ zabbix_get -s 192.168.88.18 -k agent.ping_
```

Рисунок 3 – проверка автозапуска Zabbix-агента в ОС Windows

На этом настройка Zabbix-агента в ОС Windows завершена.

Важно! Не забудьте отключить брандмауэр (или сделать на нем исключение для взаимодействия по порту 10500), и удостовериться, что Ваш антивирус не

блокирует порт 10050. Вообще, порт 10050 – стандартный порт системы Zabbix для работы агентов и Zabbix-прокси. При осуществлении комплексного мониторинга корпоративных сетей межсетевые экраны, брандмауэры и антивирусы должны разрешать сетевые взаимодействия для протоколов транспортного уровня (модели OSI) TCP / UDP по портам 10050 (агент) и 10051 (траппер – при необходимости).

Шаг 2. Теперь перейдем к системе мониторинга.

В Zabbix создайте группу узлов сети Agent Monitoring Servers, и в ней – узел сети host.windows.home, с IP-адресом Вашего ПК, на который Вы установили Zabbix-агент.

Важно!!! Название Узла сети (который будет опрашиваться как Zabbix-агент) в системе мониторинга должно совпадать с параметром Hostname в конфигурационном файле Zabbix-агента (zabbix_agentd.conf). Иначе связка Сервер – Агент не будет работать.

При создании сетевого узла в меню «Инвентарные данные узла сети» установите параметр «Автоматически» - благодаря этой настройке инвентарные данные (определенные элементы данных), собранные агентом с ОС, будут автоматически вноситься в базу данных Zabbix как инвентарные данные, и станут доступны для просмотра в меню Инвентаризация (Рис. 7.4):

The screenshot shows the Zabbix web interface for configuring a host. The main heading is 'Узлы сети' (Network Hosts). Below it, there's a breadcrumb 'Все узлы сети / host.windows.home' and a status 'Активировано'. There are tabs for 'ЗBX', 'SNMP', 'JMX', and 'IPMI'. A sub-menu is open for 'Группы элементов данных' (Data element groups), showing options: 'Узел сети', 'Шаблоны', 'IPMI', 'Макросы', 'Инвентарные данные узла сети' (selected), and 'Шифрование'. Under the selected tab, there are three buttons: 'Деактивировано', 'Вручную', and 'Автоматически' (selected). Below these are four input fields: 'Тип', 'Тип (полная детализация)', 'Имя', and 'Псевдоним'.

Рисунок 7.4 – Включение инвентаризации в рамках Узла сети

Чтобы исключить еще одно рутинное действие по включению автоматического сбора параметров инвентаризации для каждого создаваемого Узла сети, эту функцию можно глобально включить один раз для всех создаваемых узлов сети. В предыдущих версиях Zabbix этот функционал присутствовал, но не работал (Рис. 7.5):

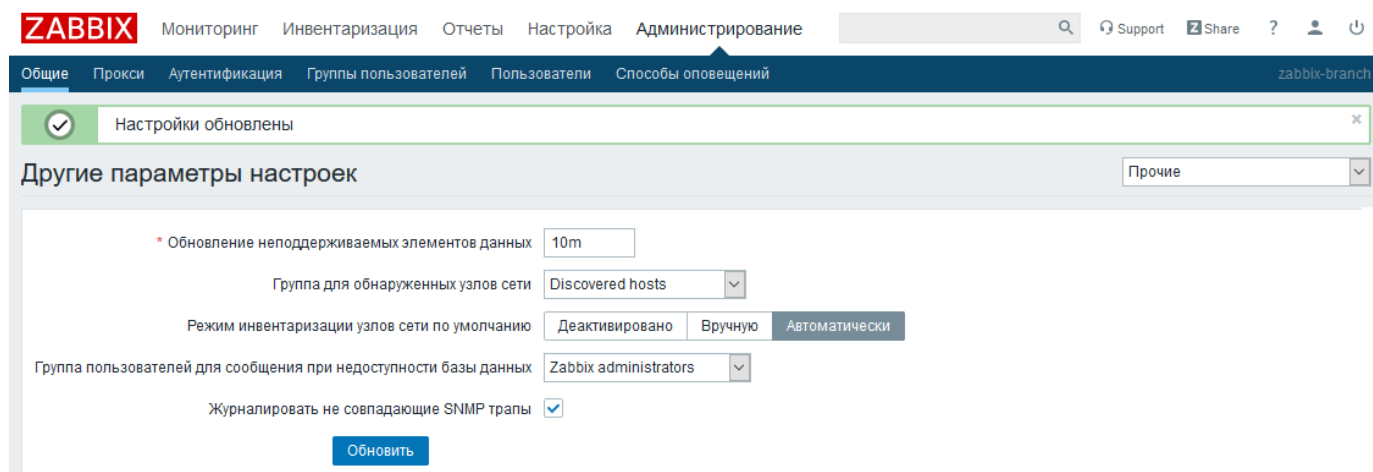


Рисунок 7.5 – Включение глобальной настройки инвентаризации по умолчанию

Создайте Шаблон опроса Agent Windows Monitor (разумеется, в группе User Templates). Шаблон опроса Agent Windows Monitor Вы будете заполнять необходимыми элементами данных согласно документации разработчиков системы Zabbix, [4], описывающей возможности Zabbix-агента для ОС Windows - обязательно ознакомьтесь с ней. Для наглядности воспользуйтесь таблицей 7.1, в которую внесены элементы данных, необходимые для полноценного мониторинга ОС Windows с помощью Zabbix-агента:

Таблица 1 - элементы данных для мониторинга ОС Windows.

Номер	Описание	Возвращаемое значение	Параметры	Примеры ключей
1	Доступность сетевого узла	1, 0	См. главу 4	icmpping[,3,,,]
2	Доступность агента	1, 0	-	agent.ping
3	Идентификация системы (инвентаризация)	Сведения о системе, текст	-	system.uname
4	Имя хоста (инвентаризация)	Имя хоста, текст	netbios / host – в зависимости от способа извлечения параметра агентом	system.hostname[host]
5	Время работы (инвентаризация)	Целое число (количество секунд)	-	system.uptime Единица измерения - uptime
6	Количество процессов	Целое число	Отображает количество процессов	proc.num[]
7	Загрузка CPU system.cpu.load[<cpu>,<режим>]	Число с плавающей точкой	cpu – возможные значения: all (по умолчанию), percpu (общая загрузка делится на количество CPU онлайн) режим – возможные значения: avg1 (усреднение за одну минуту, по умолчанию), avg5, avg15	system.cpu.load[,avg1] system.cpu.load[,avg5] system.cpu.load[,avg15]
8	Размер памяти в байтах или в процентах от общего количества.	Число с плавающей точкой	Имеет массу параметров для решения различных задач	vm.memory.size[pavailable] – отображает долю свободной оперативной памяти (в процентах)
9	Утилизация процессора	Число с плавающей точкой	Имеет массу параметров для решения различных задач	system.cpu.util[,system,avg1] - отображает загруженность, или утилизацию процессора (в процентах)

Самый первый, базовый элемент данных, присутствующий почти во всех шаблонах опроса – проверка доступности узла с помощью echo-запроса (по протоколу ICMP). Создайте его с соответствующими триггером и графиком.

На заметку! Проверка доступности сетевого узла по протоколу ICMP, а так же триггер и график для этой проверки – должны присутствовать почти в любом шаблоне опроса. Это повышает удобство использования системы и ее прозрачность (иначе к узлу сети приходится применять два разных шаблона – один для проверки доступности и второй для всего остального), что подходит не для всех ситуаций.

Второй элемент данных, отражающий работоспособность самого Zabbix-агента (agent.ping), можно визуализировать в виде графика. Однако, с триггером есть проблема. Она заключается в том, что о работоспособности Zabbix-агента предоставить информацию может только сам Zabbix-агент. Соответственно, если он по какой-то причине не отвечает (не функционирует, недоступен весь узел сети) –

сообщить об этом системе мониторинга он не может. Zabbix-сервер при попытке опроса этого элемента данных, не будет получать никаких значений. Для того, чтобы правильно оповещать администратора сети о недоступности Zabbix-агента, необходимо использовать сложный комплексный триггер, который будет срабатывать на условие, при котором сам Узел сети доступен по ICMP, а вот Zabbix-агент не отдает никаких данных. Такая конструкция выглядит следующим образом:

`{ Agent Windows Monitor:icmping[,3,,].last() }=1 and { Agent Windows Monitor:agent.ping.nodata(120) }=1`

Важно! Комплексные триггеры строятся на основе комбинации нескольких условий, налагаемых на получаемые значения от разных элементов данных (разумеется, в рамках одного шаблона).

Третий, четвертый, и пятый элементы данных (см. таблицу 1) – извлекают сведения об информационной системе. По сути, в базу данных Zabbix заносятся строковые значения, в текстовом формате. Для таких элементов данных невозможно создать триггер или график, но они служат для сбора так называемых инвентарных данных, упорядочивания их в Zabbix, и удобного вывода администратору (см. рис. 6):

Элементы данных

Все шаблоны / Agent Windows Monitor

Группы элементов данных

Элементы данных 2

Триггеры 2

Элемент данных

Предобработка

* Имя

Тип

* Ключ

Тип информации

* Интервал обновления

Пользовательские интервалы

Тип	Интервал	Период	Дей
Добавить			

* Период хранения истории

Новая группа элементов данных

Группы элементов данных

Заполнение поля инвентаря узла сети

Рисунок 6 – Элемент данных для получения сведения о системе

<input type="checkbox"/>	Мастер	Имя ▲	Триггеры	Ключ	Интервал	История	Динами
<input type="checkbox"/>	...	Время работы (инвентаризация)		system.uptime	15m	0	0
<input type="checkbox"/>	...	Доступность Zabbix-агента	Триггеры 1	agent.ping	1m	60d	60d
<input type="checkbox"/>	...	Доступность узла	Триггеры 2	icmpping[.3,...]	1m	60d	60d
<input type="checkbox"/>	...	Идентификация системы (инвентаризация)		system.uname	24h	60d	
<input type="checkbox"/>	...	Имя хоста (инвентаризация)		system.hostname[host]	24h	60d	

После тестирования – установите периодичность опроса 1 день (ед. измерения 1d), кроме времени работы (там используйте 15m-20m)

host.windows.home - other - (5 элементов данных)			
<input type="checkbox"/>	Время работы (инвентаризация)	17.09.2018 13:48:49	2 дня, 04:14:42
<input type="checkbox"/>	Доступность Zabbix-агента	17.09.2018 13:48:45	1
<input type="checkbox"/>	Доступность узла	17.09.2018 13:48:26	1
<input type="checkbox"/>	Идентификация системы (инвентаризация)	17.09.2018 13:48:48	Windows BEE-PC 10.0.17134 M...
<input type="checkbox"/>	Имя хоста (инвентаризация)	17.09.2018 13:48:47	BEE-PC

Рисунок 7 – Последние данные после сбора инвентаризационной информации

ZABBIX					
Мониторинг Инвентаризация Отчеты Настройка Администрирование					
Обзор Узлы сети					
Инвентарные данные узла сети					
Узел сети ▲	Группа	Имя	Тип	ОС	Этикетка
host.windows.home	Agent Monitoring Servers	BEE-PC	Windows BEE-PC 10.0.17134 Майкрософт Windows 10 Pro x64		2 days, 04:20:42

Рисунок 8 – Просмотр инвентарных данных узла сети

На заметку! Сбор инвентарных данных – важная часть мониторинга сети. С серверов можно собирать сведения об ОС, с сетевого оборудования – точные названия моделей (так называемые Product ID (PID) и серийные номера шасси). Все это помогает упорядочить наблюдаемую сеть и упростить задачи по ее администрированию.

Обратите внимание, используемый Тип данных – Zabbix-агент. В данном разделе во всех элементах данных будет использоваться только он.

Шестой элемент данных, proc.num (см. таблицу 7.1), запрашивает количество выполняемых на текущий момент процессов – в ОС Windows это можно наблюдать в диспетчере задач. Создайте элемент данных и график к нему.

Седьмой элемент данных, system.cpu.load (см. таблицу 7.1), запрашивает средние значения загрузки процессора за 1, 5 или 15 минут. Вообще, данный параметр (в компьютерной технике называемый Load Average), замечателен с точки зрения

наглядности работы процессоров. Load Average - это среднее значение загрузки системы за некоторый период времени (1, 5 и 15 минут), вычисляемое по сложной формуле, зависящей от так называемой «длины очереди выполнения». Длина очереди выполнения - количество всех процессов, выполняемых в данный момент времени, плюс количество процессов, ожидающих в очереди. Подробнее понятие Load Average и способ его вычисления рассматриваются в источниках [5-7]. В данном примере Load Average переведено как «Нагруженность процессора», однако полноценного краткого аналога данного термина в русском языке нет.

Создайте 3 элемента данных system.cpu.load для периодов времени 1, 5 и 15 минут, график со всеми тремя элементами данных, и триггеры для каждого элемента данных (порог срабатывания триггера установите из расчета 0.7 для каждого ядра процессора – данный пример был подготовлен на двухъядерном процессоре i5-3230M, и порог срабатывания составил 1.4). Для пояснения рассмотрим рисунок 7.9 (аномальная загрузка процессора была вызвана с помощью запуска нескольких процессов архивации видеофайлов):

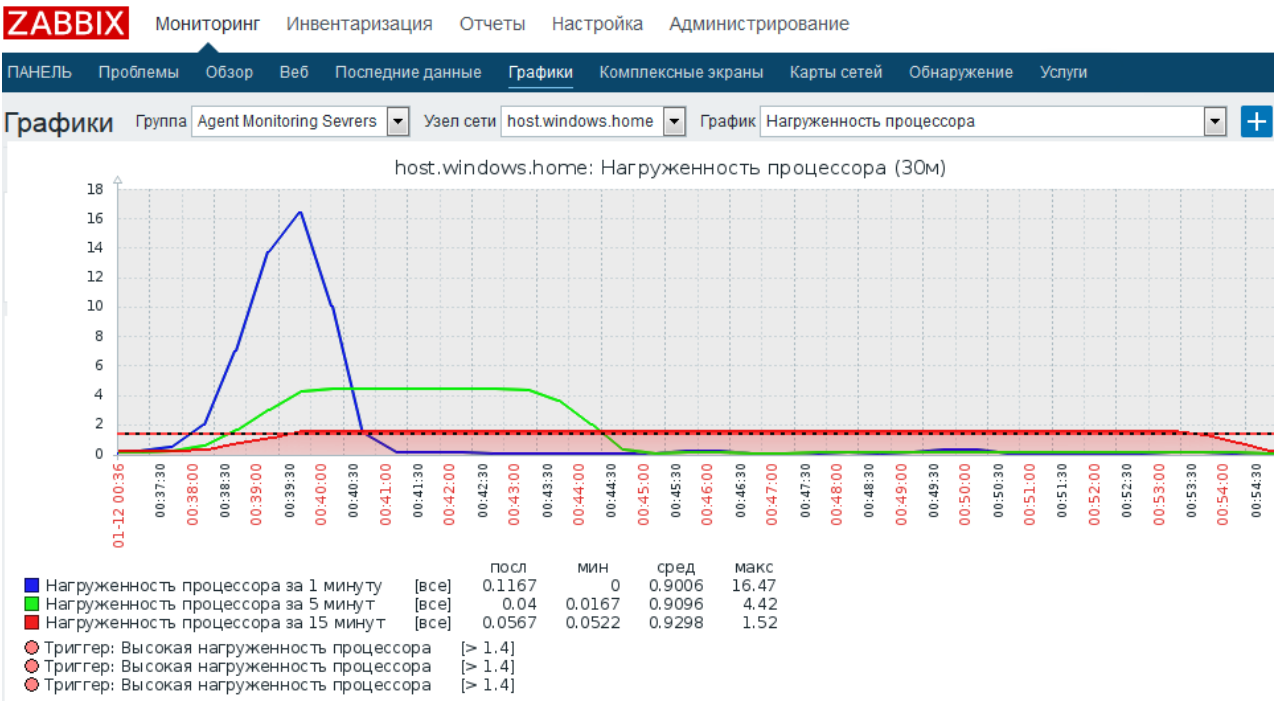


Рисунок 9 – график для визуализации собранных значений Load Average (нагруженность процессора)

Восьмой элемент данных - `vm.memory.size[pavailable]` отображает долю свободной (незанятой) оперативной памяти (в процентах). У элемента данных `vm.memory.size` масса параметров для отображения общего количества памяти, занятости различных типов памяти – все это используется в зависимости от поставленных задач. Для простого мониторинга свободной оперативной памяти создайте элемент данных, триггер (с оповещением о том, что доступной оперативной памяти менее 20%), и график.

Девятый элемент данных (см. таблицу 1) - `system.cpu.util` показывает «утилизацию», или загруженность процессора. По сути, загруженность процессора - процент времени, в течение которого процессор загружен работой (не простаивает). `system.cpu.util` запрашивает усреднённую утилизацию процессора за интервал времени. На каждом отрезке, на котором не выполняется Idle Thread (этот термин обозначает простой), процессор считается занятым какой-то реальной нагрузкой. Чем больше полученное значение (в нашем случае, речь идет о процентах), тем больше загрузка процессора и меньше простоев. Создайте элемент данных, триггер (с порогом 80%), и график.

Важно! При создании шаблонов опроса, элементов данных, триггеров и графиков, возьмите за правило называть их понятными именами, и при необходимости добавлять описания и пояснения.

Создание триггеров:

Проверка загруженности процессора:

```
{Agent Windows Monitor: system.cpu.load[,avg1].last()}>0.7 or {Agent Windows Monitor: system.cpu.load[,avg5].last()}>3 or {Agent Windows Monitor: system.cpu.load[,avg15].last()}>7
```

Проверка работоспособности агента (agent.ping) по icmp

```
{Agent Windows Monitor: agent.ping.nodata(1m)=1} and {Agent Windows Monitor: icmping[,3,,].last()}=1
```

Проверка доступности свободной памяти (не более 80%):

{Agent Windows Monitor:system.cpu.util[,system,avg1].last()}>80

{Agent Windows Monitor:vm.memory.size[pavailable].last()}>80

Доступность в сети:

{Agent Windows Monitor:icmpping[,3,,].last()}=0

{Agent Windows Monitor:system.cpu.load[,avg1].last()}>0.7 or {Agent Windows Monitor:system.cpu.load[,avg5].last()}>3 or {Agent Windows Monitor:system.cpu.load[,avg15].last()}>7

{Agent Windows Monitor:agent.ping.nodata(1m)}=1 and {Agent Windows Monitor:icmpping[,4,,].last()}=1

{Agent Windows Monitor:system.cpu.util[,system,avg1].last()}>80

{Agent Windows Monitor:vm.memory.size[pavailable].last()}>80

{Agent Windows Monitor:icmpping[,3,,].last()}=0

Осталось последнее – организовать мониторинг сетевых интерфейсов и свободного пространства на жестких дисках. Возникает интересный момент – речь идет о мониторинге однотипных объектов, число которых может меняться со временем (например - разбиение жесткого диска на несколько разделов, добавление новых жестких дисков или удаление старых, установка внешних носителей памяти). Значит, нужен некий инструмент, способный автоматически находить, создавать, упорядочивать и удалять элементы данных, принадлежащие определенной группе. Этот инструмент – низкоуровневое обнаружение (Low-level discovery - LLD).

Низкоуровневое обнаружение даёт возможность автоматического создания элементов данных, триггеров и графиков для различных объектов на серверном и сетевом оборудовании. Благодаря LLD, Zabbix может осуществлять автоматический мониторинг файловых систем или сетевых интерфейсов, без необходимости создания вручную элементов данных для каждой файловой системы или сетевого интерфейса. Кроме того, Zabbix может удалять объекты, которые перестали обнаруживаться, по окончании заданного периода ожидания или немедленно, после очередного цикла обнаружения.

В Zabbix поддерживаются несколько встроенных типов элементов данных для обнаружения:

- обнаружение сетевых интерфейсов;
- обнаружение CPU и ядер CPU;
- обнаружение SNMP OID;
- обнаружение с использованием SQL запросов ODBC;
- обнаружение Windows служб.

Если стандартные функции LLD не удовлетворяют поставленной задаче – обнаружение можно организовать с помощью внешнего исполняемого скрипта, который Zabbix будет использовать в качестве инструмента LLD.

Вернемся к обнаружению сетевых интерфейсов и жестких дисков. Начнем с мониторинга загрузки трафиком сетевых интерфейсов - в шаблоне опроса Agent

Windows Monitor перейдите в подменю «Правила обнаружения» и создайте правило обнаружения.

В созданном правиле обнаружения заполните следующие поля (Рис. 10): Имя (название правила), Тип (тип LLD – в нашем случае Zabbix-агент), ключ – net.if.discovery (инструмент LLD для Zabbix-агента), Интервал обновления (как часто будет проводиться обнаружение – на этапе создания обнаружения установите значение 1 минута, после того, как добьетесь стабильной работы шаблона – установите 1 час), Период сохранения потерянных ресурсов (в случае, если при новом обнаружении ранее существовавший объект не будет обнаружен, как скоро можно его удалить – установите 0, то есть немедленно).

Правила обнаружения

Все шаблоны / Agent Windows Monitor Список обнаружений / Обнаружение сетевых интерфейсов Прототипы

Правило обнаружения Фильтры

* Имя: Обнаружение сетевых интерфейсов

Тип: Zabbix агент

* Ключ: net.if.discovery

* Интервал обновления: 1m

Пользовательские интервалы

Тип	Интервал	Период	Действие
Добавить			

* Период сохранения потерянных ресурсов: 0

Рисунок 10 – Настройка правила обнаружения

Теперь в правиле обнаружения необходимо создать так называемые прототипы элементов данных для подсчета трафика (для Zabbix-агента это net.if.in и net.if.out). Прототипы элементов данных отличаются от обычных элементов данных тем, что в их ключах (идентификаторах обращения к определенному объекту) присутствует переменная. По сути, имеет место цикл, который проходит по всем однотипным

объектам, в именах (адресах, или идентификаторах) которых отличается только один параметр.

По логике работы LLD, сначала net.if.discovery найдет сетевые интерфейсы, а элементы данных net.if.in и net.if.out считают различные показатели трафика для каждого из них (благодаря переменной в ключе). Элементы данных net.if.in и net.if.out имеют следующие ключи: net.if.in[if,<режим>], net.if.out[if,<режим>].

Аргумент if – как раз таки название (или уникальный идентификатор) сетевого интерфейса - для корректной работы прототипов элементов данных if заменяется макросом {#IFNAME}. Далее, после обнаружения сетевых интерфейсов, эти макросы заменяются реальными значениями и становятся основой для создания реальный элементов данных, триггеров и графиков (на основе прототипов триггеров и графиков, в автоматическом режиме).

Аргумент <режим> - может принимать значения bytes (счетчик байт, прошедших через интерфейс), packets (счетчик пакетов – его использовать мы не будем), errors (счетчик «искаженных» etherhnet-кадров), dropped (счетчик отброшенных etherhnet-кадров).

Создадим прототип элементов данных net.if.in[{#IFNAME},bytes], для подсчета входящего трафика. При заполнении полей прототипа элементов данных, пользуйтесь макросом {#IFNAME}. В поле Имя введите «Входящий трафик {#IFNAME}», в поле ключ – «net.if.in[{#IFNAME},bytes]», Единицы измерения – byte (для корректного масштабирования значений сетевого трафика), правило Предобработки установите «Изменение в секунду» (рис. 11):

Прототипы элементов данных

Все шаблоны / Agent Windows Monitor

Список обнаружений / Обнаружение сетевых интерфейсов

Прототипы эл

Прототип элемента данных

Предобработка

* Имя	Входящий трафик {#IFNAME}		
Тип	Zabbix агент		
* Ключ	net.if.in[{#IFNAME},bytes]		
Тип информации	Числовой (целое положительное)		
Единица измерения	byte		
* Интервал обновления	1m		
Пользовательские интервалы	Тип	Интервал	Период
	Добавить		
* Период хранения истории	60d		
* Период хранения динамики изменений	60d		

Рисунок 11 – Создание прототипа элемента данных

Правило Предобработка указывает, как хранить полученные данные – например, для этого случая необходимо использовать правило Изменение в секунду. Zabbix отдает разность полученных значений за единицу времени (или шаг опроса) - для фиксации динамики изменений параметров. В прошлых версиях Zabbix этот параметр назывался «Дельта (скорость в секунду)». Счетчики трафика на сетевых интерфейсах просто увеличиваются при прохождении через них данных, поэтому для корректного отображения именно скорости, а не количества трафика, используется Изменение в секунду.

Аналогичным образом создайте остальные прототипы элементов данных - net.if.out[{#IFNAME},bytes], net.if.in[{#IFNAME},dropped], net.if.out[{#IFNAME},dropped], net.if.in[{#IFNAME},errors], net.if.out[{#IFNAME},errors]. У Вас должно получиться следующее (рис. 12):

Прототипы элементов данных						
Все шаблоны / Agent Windows Monitor Список обнаружений / Обнаружение сетевых интерфейсов Прототипы элементов данных 6 Прототип						
<input type="checkbox"/>	Мастер	Имя ▲	Ключ	Интервал	История	Динами
<input type="checkbox"/>	...	Входящий трафик {#IFNAME}	net.if.in[{#IFNAME},bytes]	1m	60d	60d
<input type="checkbox"/>	...	Исходящий трафик {#IFNAME}	net.if.out[{#IFNAME},bytes]	1m	60d	60d
<input type="checkbox"/>	...	Отброшено входящих, {#IFNAME}	net.if.in[{#IFNAME},dropped]	1m	60d	60d
<input type="checkbox"/>	...	Отброшено исходящих, {#IFNAME}	net.if.out[{#IFNAME},dropped]	1m	60d	60d
<input type="checkbox"/>	...	Ошибок, входящих - {#IFNAME}	net.if.in[{#IFNAME},errors]	1m	60d	60d
<input type="checkbox"/>	...	Ошибок, исходящих - {#IFNAME}	net.if.out[{#IFNAME},errors]	1m	60d	60d

Рисунок 12 –Прототипы элементов данных для мониторинга трафика

Создайте 2 прототипа графиков – для отображения загрузки интерфейсов трафиком и всех видов ошибок на интерфейсах. Прототипы графиков создаются в рамках правил обнаружения. В названиях прототипов графиков обязательно используйте макрос {#IFNAME}. Помещать и загрузку трафика, и уровень ошибок на один график не рекомендуется из-за большого различия величин загрузки трафиком и количества ошибок (несколько порядков). Скорость трафика на интерфейсе может быть несколько Мбит/с, а ошибок может быть 1-2, и из-за масштабирования всех кривых эти ошибки будут не видны на фоне уровня загрузки трафиком.

Еще один важный момент – в ОС Windows существует множество объектов, подпадающих под понятие «сетевой интерфейс» и обнаруживающихся с помощью LLD-инструмента net.if.discovery (это виртуальные шлюзы IPv6, адаптеры, всевозможные «псевдоинтерфейсы» и виртуальный интерфейсы платформ виртуализации, туннелей, некоторых служб и приложений). Когда Вы создадите прототипы элементов данных, Zabbix обнаружит и внесет в узел сети host.windows.home массу объектов (Рис. 13):

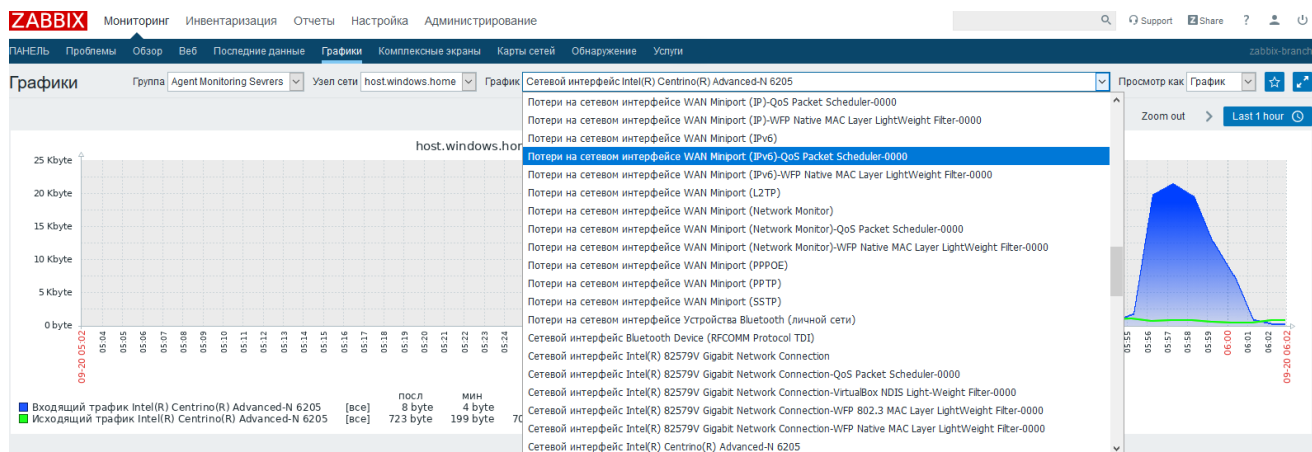


Рисунок 13 – Сетевые интерфейсы в ОС Windows

Для того, чтобы корректно отфильтровать полезную информацию, воспользуйтесь фильтрацией для правил обнаружений с помощью регулярных выражений. LLD-инструмент net.if.discovery будет получать информацию о сетевых интерфейсах, и регистрировать только те данные, макросы {#IFNAME} которых пройдет через фильтр регулярного выражения. На рисунке 14 показан процесс подключения набора регулярных выражений win-eth (обратите внимание на синтаксис, в поле вносится значение «@win-eth»):

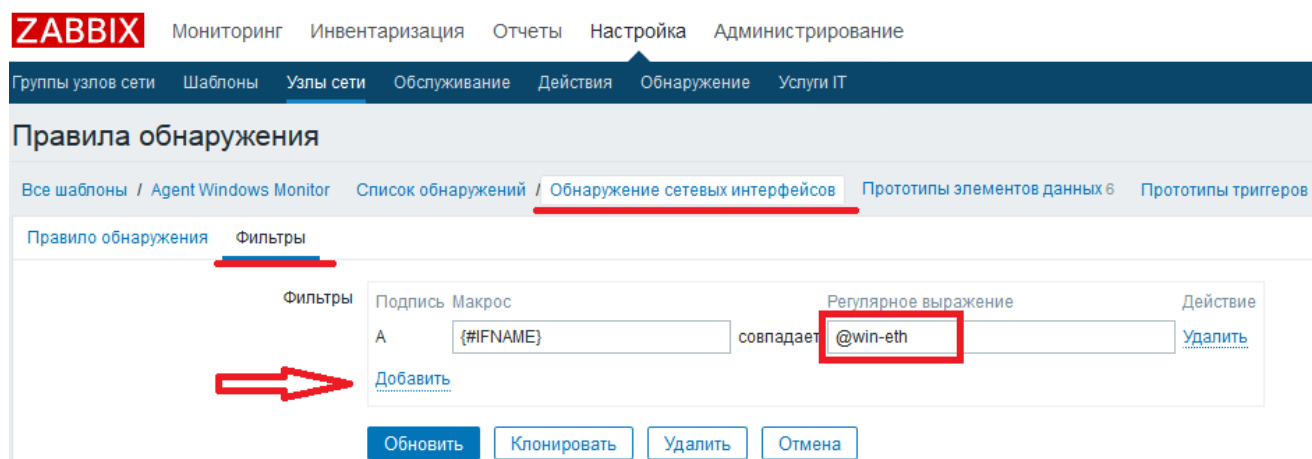


Рисунок 14 – Создание фильтра регулярного выражения

Осталось создать само регулярное выражение. Перейдите в меню Администрирование – Общие – Регулярные выражения и создайте новое регулярное выражение «win-eth». Обратите внимание, знак @ перед ним не ставится (Рис. 15):

<input type="checkbox"/> win-eth	1	» WFP	[Результат ЛОЖЬ]
	2	» NDIS	[Результат ЛОЖЬ]
	3	» Loopback	[Результат ЛОЖЬ]
	4	» VMware	[Результат ЛОЖЬ]
	5	» Bridge	[Результат ЛОЖЬ]
	6	» Miniport	[Результат ЛОЖЬ]
	7	» Debug	[Результат ЛОЖЬ]
	8	» Kernel	[Результат ЛОЖЬ]
	9	» Platform	[Результат ЛОЖЬ]
	10	» Virtual	[Результат ЛОЖЬ]
	11	» Adapter	[Результат ЛОЖЬ]
	12	» MAC	[Результат ЛОЖЬ]
	13	» Driver	[Результат ЛОЖЬ]
	14	» VirtualBox	[Результат ЛОЖЬ]
	15	» Tunnel	[Результат ЛОЖЬ]
	16	» QoS	[Результат ЛОЖЬ]
	17	» Bluetooth	[Результат ЛОЖЬ]

Рисунок 15 – Создание регулярного выражения

После создания регулярного выражения, все объекты, не прошедшие фильтр, будут удалены и больше не появятся в списке обнаруженных (фильтрация происходит одновременно с обнаружением – если частота обнаружения установлена, например, 1 час, то и фильтр к обнаруживаемым объектам будет применяться один раз в час, в процессе обнаружения). Проверьте, что фильтрация работает – в элементах данных Узла сети должны исчезнуть ненужные объекты, и, соответственно, должны исчезнуть лишние графики (Рис. 16):

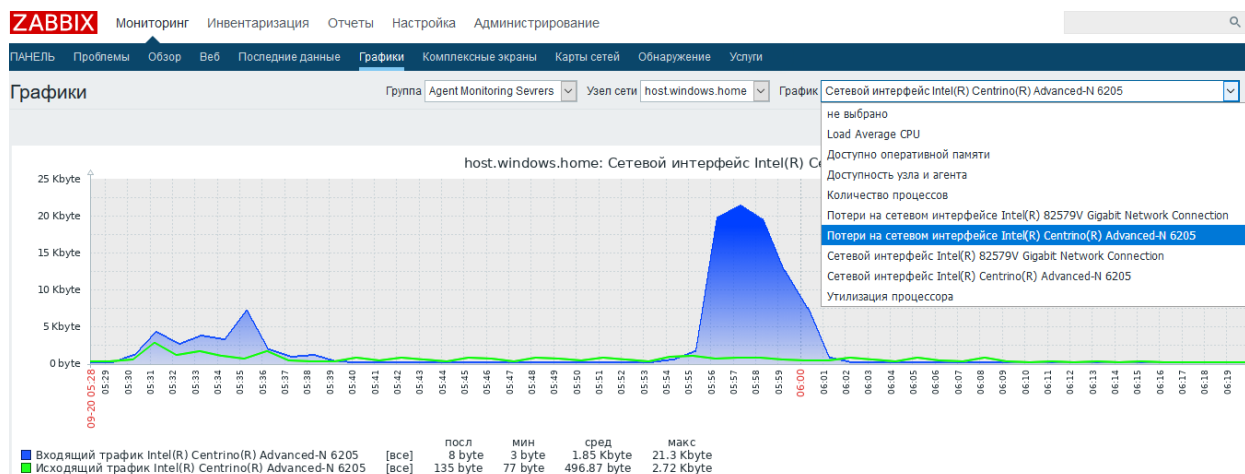


Рисунок 16 – Результат фильтрации сетевых интерфейсов

Следующий шаг – создание еще одного низкоуровневого обнаружения, для мониторинга свободного дискового пространства на энергонезависимых носителях. Инструмент низкоуровневого обнаружения накопителей памяти - `vfs.fs.discovery`, прототип элемента данных для отображения доли свободного дискового пространства (в процентах) - `vfs.fs.size[#{FSNAME},pfree]`, где `{#FSNAME}` – макрос, несущий в себе букву диска (C:/, D:/ и т.п.). Создайте LLD самостоятельно (учтите, фильтр для него не нужен), прототип графика и триггера (с порогом, например, 20%).

На этом создание шаблона опроса для серверов с ОС Windows с помощью Zabbix-агента завершено. Благодаря этому шаблону мы получаем данные о производительности системы, состоянии памяти и жестких дисков. Убедитесь, что обнаружения, прототипы триггеров и графиков работают корректно. В результате работы шаблона, для просмотра должны быть доступны графики и инвентарные данные, настроены триггеры и обнаружения.

Осталось визуализировать получаемые значения. Воспользуйтесь инструментом Комплексный экран. Он служит для размещения на одной веб-странице нескольких графиков. Комплексные экраны в Zabbix бывают двух типов. Первые создаются в Шаблонах опроса, создаются автоматически для Узлов сети, к которым присоединен Шаблон (это плюс) и доступны для просмотра только с карты сетей, если на ней размещен интересующий Вас узел сети (это минус). Вторые создаются на панели меню Мониторинг – Комплексные экраны, и они доступны только из раздела меню

Мониторинг – Комплексные экраны, и они несут только те элементы, которые Вы настроили при создании комплексного экрана (это минус, полностью отсутствует автоматизация и масштабирование). Для обслуживания ОС Windows воспользуйтесь первым типом комплексных экранов. Войдите в соответствующее меню в Шаблоне, и создайте комплексный экран host.linux.home размером 3x3. Войдите в режим конструктора, и для каждой ячейки задайте график или прототип графика, который Вы хотите видеть на комплексном экране, поэкспериментируйте. Отследите плюсы и минусы использования прототипов графиков на комплексном экране. Пример комплексного экрана приведён на рисунке 17 (обратите внимание на часы – это аналог Виджета Панелей мониторинга).

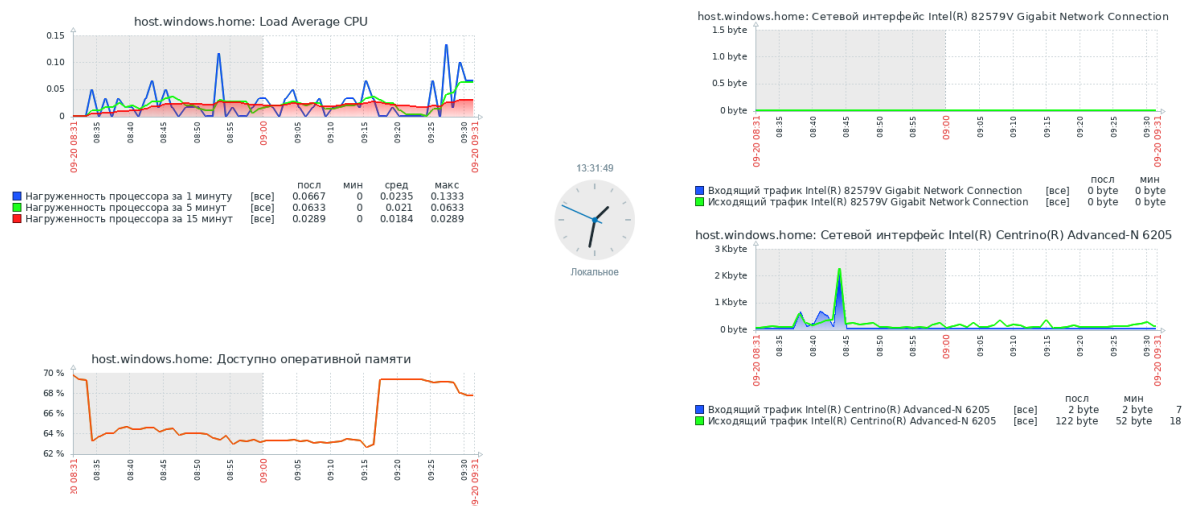


Рисунок 17 – Пример комплексного экрана

Работы с Шаблоном опроса для ОС Windows закончены. При необходимости шаблон всегда можно дополнить, обратившись к документации разработчиков и выбрать недостающие Вам элементы.