OSNOVO

cable transmission

РУКОВОДСТВО ПО ЭКСПЛУАТАЦИИ

Программный интерфейс API

(Application Programming Interface)

для WEB-управляемых коммутаторов

Прежде чем приступать к эксплуатации изделия,
внимательно прочтите настоящее руководство

www.osnovo.ru

SEC.RU | Короткий путь к информации Скачано с sec.ru

Оглавление

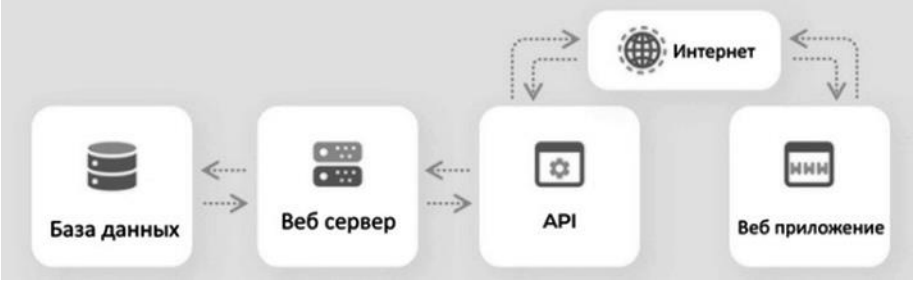
1. Application Programming Interface (Введение) 3
2. Login (Авторизация) 4
3. Logout (Выход из системы) 5
4. System Information (Информация о системе) 6
5. PoE Status (Статус PoE) 7
6. PoE Config (Конфигурация PoE) 8
7. Set PoE Config (Установка конфигурации PoE) 9
8. MAC Table Information (Таблица MAC-адресов) 11
9. Port Status (Статус портов) 12
Приложение А «Использование интерфейса API с системой мониторинга Zabbix» 13

SEC.RU | Короткий путь к информации
Скачано с sec.ru

1. Application Programming Interface (Введение)

API (Application Programming Interface) — это программный интерфейс, который позволяет разным приложениям взаимодействовать между собой. Он обеспечивает передачу данных и выполнение различных функций, упрощая интеграцию между сервисами.

API выполняет роль связующего звена между различными программами, он содержит список доступных операций, правил взаимодействия и ограничений.



Ключевая задача API — обмен данными между программами. В типичной схеме взаимодействия данные передаются между сервером, API и пользовательским интерфейсом (мобильным приложением или сайтом). Если API перестает работать, доступ к связанным с ним сервисам тоже может быть ограничен.

Термин API может относиться к разным компонентам:

- отдельному программному обеспечению;
- модулю внутри приложения;
- набору функций, выполняющих определенные задачи;
- серверной части, обеспечивающей работу системы.

API является библиотекой готовых решений, которые разработчики могут использовать вместо создания функционала с нуля. Хотя API и протоколы решают схожие задачи, между ними есть важные различия. API описывает, как программы могут взаимодействовать друг с другом, а протокол определяет правила передачи данных.

SEC.RU | Короткий путь к информации Скачано с sec.ru

Внимание !

- Перед вызовом API требуется выполнить тот же процесс входа в систему и аутентификации, что и в веб-интерфейсе. После успешной аутентификации будет получен файл cookie для доступа к API.

- Запрос JSON для API инкапсулирован в протокол HTTP.

2. Login (Авторизация)

URL: /api/login
Method: POST
Request JSON

```
{
    "login": {
        "username" : "admin",
        "password" : "admin"
    }
}
```

Response JSON

```
{
  "response":{
    "status":"error",
    "message":"Wrong username or password!"
    }
}
```

Or:

```
{
    "response":{
      "status":"success",
      " message":""
    }
}
```

*При авторизации генерируется временный token коммутатора.

SEC.RU | Короткий путь к информации Скачано с sec.ru

Внимание !

После успешного входа в систему сервер отправит заголовок Set-Cookie. Последующие команды, отправленные с соответствующим файлом cookie в HTTP-сообщении, могут затем выполнять API, описанный ниже.

3. Logout (Выход из системы)

URL: /api/logout
Method: POST
Request JSON

```
{
      "logout": {
      }
}
```

Response JSON:

```
{
     "response":{
        "status":"error",
        "message":"Wrong session!"
   }
}
```

Or:

```
{
     "response":{
        "status":"success",
        "message":""
   }
}
```

Внимание !

Для выхода из системы не требуется никаких параметров. Когда файл cookie "logout" будет включен в HTTP-заголовок, файл cookie немедленно станет недействительным, что приведет к выходу пользователя из системы.

SEC.RU | Короткий путь к информации Скачано с sec.ru

4. System Information (Информация о системе)

URL: /api/get_sysinfo
Method: GET Request
JSON: null
Response JSON:


```
{
  "system": {
    "information": {
      "model_name": "24G-2G-2F",
      "description": "24G-2G-2F",
      "system_name": "24G-2G-2F",
      "hardware_version": "1.0",
      "mechanical_version": "1.0",
      "firmware_version": "V1.0",
      "mac_addr": "00:00:00:00:00:02",
      "serial_number": "LK30262402XXXXXX",
      "uptime": "0d0h53min58s",
      "location": "",
      "contact": "",
      "system_date": "",
      "cpu_load": "",
      "ram": {
        "total": "",
        "free": ""
      },
      "temperature_1": 0,
      "temperature_2": 0,
      "temperature_3": 0
    }
  }
}
```


SEC.RU | Короткий путь к информации Скачано с sec.ru

5. PoE Status (Статус PoE)

URL: /api/get_poe_status
Method: GET Request
JSON: null
Response JSON:

```
{
  "poe": {
    "total_power_allocate": 600,
    "total_power_used": 150,
    "total_current_used": 278
  },
  "ports": [
    {
      "id": 1,
      "poe": {
        "pd_class": "-",
        "port_status": "No PD detected",
        "power_allocate": 0,
        "power_used": 0,
        "current_used": 0
      }
    },
    ... ...
    ]
}
```

Section:

Name	Data type	Unit	Note
total_power_allocate	Integer	0.1 watt	All port power limit sum
total_power_used	Integer	0.1 watt	Total power used by all ports
total_current_used	Integer	mA	Real current use by all ports
id	Integer	-	<Port number>
power_allocate	Integer	watt	Power limit on the port
power_used	Integer	0.1 watt	Real power used by the port
current_used	Integer	mA	Real current use by the port

SEC.RU | Короткий путь к информации Скачано с sec.ru

6. PoE Config (Конфигурация PoE)

URL: /api/get_poe_config
Method: GET Request
JSON: null
Response JSON:

```
{
  "poe": {
    "total_power_watts": 110
  },
  "ports": [
    {
      "id": 1,
      "power_limit_user": 90,
      "poe": {
        "mode": "Enabled",
        "extend_mode": "On"
      }
    },
  ... ...
  ]
}
```

Section:

Name	Data type	Unit	Note
total_power_watts	Integer	watt	
power_limit_user	Integer	watt	
id	Integer	-	<Port number>
mode	String	-	"Enabled", "Disabled"
extend_mode	String	-	"On", "Off"

SEC.RU | Короткий путь к информации Скачано с sec.ru

7. Set PoE Config (Установка конфигурации PoE)

URL: /api/set_poe_config
Method: POST
Request JSON:

```
{
  "ports": [
    {
      "id": 1,
       "poe":{
            "mode":              "Disabled",
            “extend_mode”:      “On”
       }
    },
    ... ...
  ]
}
```

Response JSON:

```
{
   "poe": {
      "total_power_watts": 110
   },
   "ports": [
      {
         "id": 1,
         "power_limit_user": 90,
         "poe": {
            "mode": "Enabled",
            “extend_mode”: “On”
         }
      },
      ... ...
   ]
}
```


SEC.RU | Короткий путь к информации Скачано с sec.ru

Section:

Name	Data type	Allowed / Value	Default Value
total_power_watts	Integer	Depend on product	System maximum power supported
id	Integer	<Port number>	
power_limit_user	Integer	1-90watt	30 for AT port, 90 for BT port
mode	String	"Enabled", "Disabled"	"Enabled"
extend_mode	String	"On", "Off"	"Off"

Внимание !

По причине ограничений каждая конфигурация не должна включать установки более чем для 6и портов.

- Рекомендуется настраивать порты последовательно;
- При отправке в формате JSON используйте компактный формат, чтобы избежать ненужных символов, таких как перевод строки или пробелы.

Пример:

{"ports":[{"id":1,"power_limit_user":90,"poe":{"mode":"Disabled","extend_mode":"On"}},{"id":2,"power_limit_user":90,"poe":{"mode":"Disabled","extend_mode":"On"}},{"id":3,"power_limit_user":90,"poe":{"mode":"Disabled","extend_mode":"On"}},{"id":4,"power_limit_user":30,"poe":{"mode":"Disabled","extend_mode":"On"}},{"id":5,"power_limit_user":30,"poe":{"mode":"Disabled"}},{"id":6,"power_limit_user":30,"poe":{"mode":"Disabled","extend_mode":"On"}}]}

SEC.RU | Короткий путь к информации Скачано с sec.ru

8. MAC Table Information (Таблица MAC-адресов)

URL: /api/get_dynamic_mac_table
Method: GET
Request JSON: null
Response JSON


```
{
  "mac_table": [
    {
      "type": "Dynamic",
      "mac": "C0:18:50:46:6D:CE",
      "vid": 1,
      "port": 2
    },
    {
      "type": "Dynamic",
      "mac": "F0:00:00:FD:9B:D4",
      "vid": 1,
      "port": 8
    }
    ... ...
  ]
}
```


Внимание !

По причине ограниченнной производительности процессора и необходимости сканирования системой таблицы MAC-адресов размером 8 КБ, для моделей с 16 и 24 портами требуется больше времени для возврата команды (примерно 15 сек). Поэтому при выполнении этой команды облачной платформе следует установить более длительный тайм-аут.

SEC.RU | Короткий путь к информации Скачано с sec.ru

9. Port Status (Статус портов)

URL: /api/get_port_status
Method: GET
Request JSON: null
Response JSON:

```
{
  "ports": [
    {
      "id": 1,
      "status": {
        “link”  :  “Up”
        "speed":  "1G",
        “duplex”:  “Full”
      }
    },
    … …
    ]
}
```

Section:

Name	Data type	Unit	Note
link	String	-	“Up”, “Down”
speed	String	-	“10M”,”100M”, “1G”
duplex	String	-	“Full”, “Half”

SEC.RU | Короткий путь к информации Скачано с sec.ru

Приложение А «Использование интерфейса API с системой мониторинга Zabbix»

Zabbix — это система мониторинга с открытым исходным кодом, которая позволяет в реальном времени следить за состоянием IT-инфраструктуры: серверов, сетевого оборудования, приложений и т.п. Zabbix удобен тем, что позволяет гибко настраивать мониторинг в соответствии с потребностями конкретной компании или проекта.

Дополнительные возможности системы мониторинга Zabbix:

- обнаружение проблем в режиме реального времени;
- большое количество каналов оповещения;
- визуализация данных;
- единый интерфейс для всей архитектуры;
- мониторинг в многопользовательской среде;
- простой процесс установки и интеграции с уже существующей системой;
- неограниченное масштабирование.

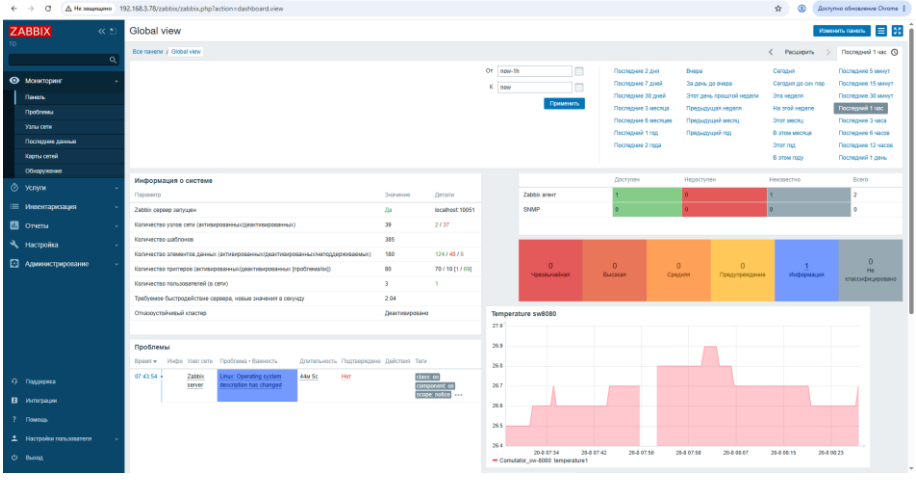


Рис.1 Интерфейс системы мониторинга Zabbix

SEC.RU | Короткий путь к информации
Скачано с sec.ru

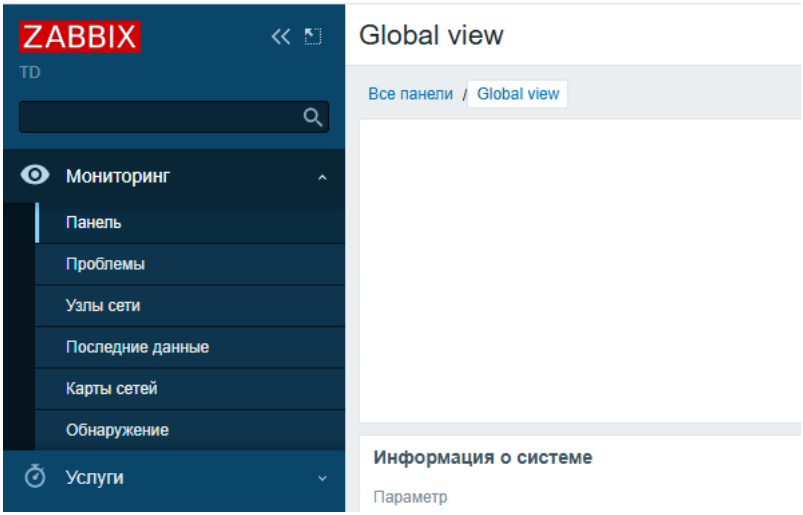


Рис.2 Расположение панели управления (в левой части)

Подключение коммутатора к системе мониторинга Zabbix.

В меню *Мониторинг* (Рис.2) или *Настройки* выбрать раздел *Узлы сети*;

В правом верхнем углу нажать *Создать узел сети*, далее откроется окно (Рис.3):

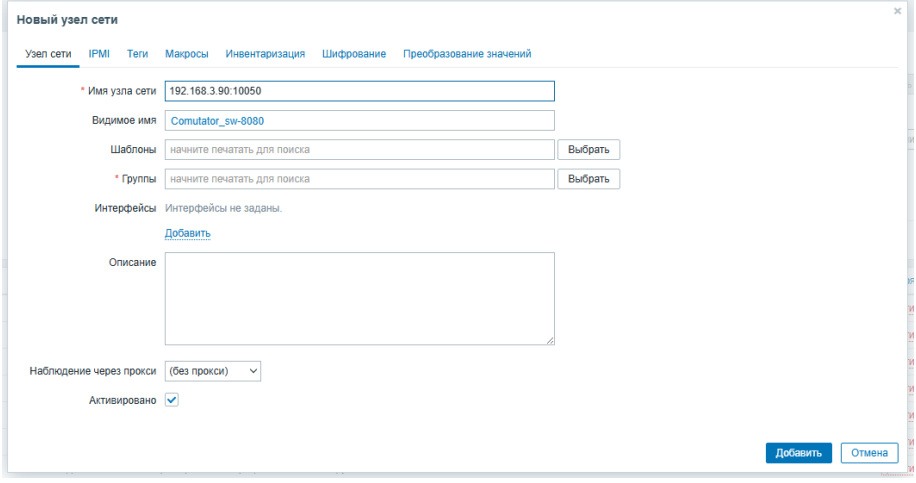


Рис.3 Добавление коммутатора в систему мониторинга

SEC.RU | Короткий путь к информации
Скачано с sec.ru

Поля, отмеченные*, обязательны для заполнения.

Имя узла сети – вводится IP-адрес добавляемого устройства (коммутатора);

Группы – требуется создать новую или выбрать из существующих;

Для завершения нажать **Добавить** в правом нижнем углу окна. Коммутатор появится в списке устройств:

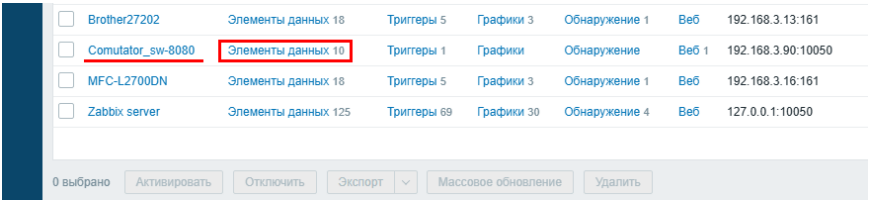


Рис.4 Список узлов сети (устройств)

Для продолжения настройки следует нажать *Элементы данных* (Рис.4) и на открывшейся странице в правом верхнем углу нажать *Создать элемент данных*, откроется окно (Рис.5):

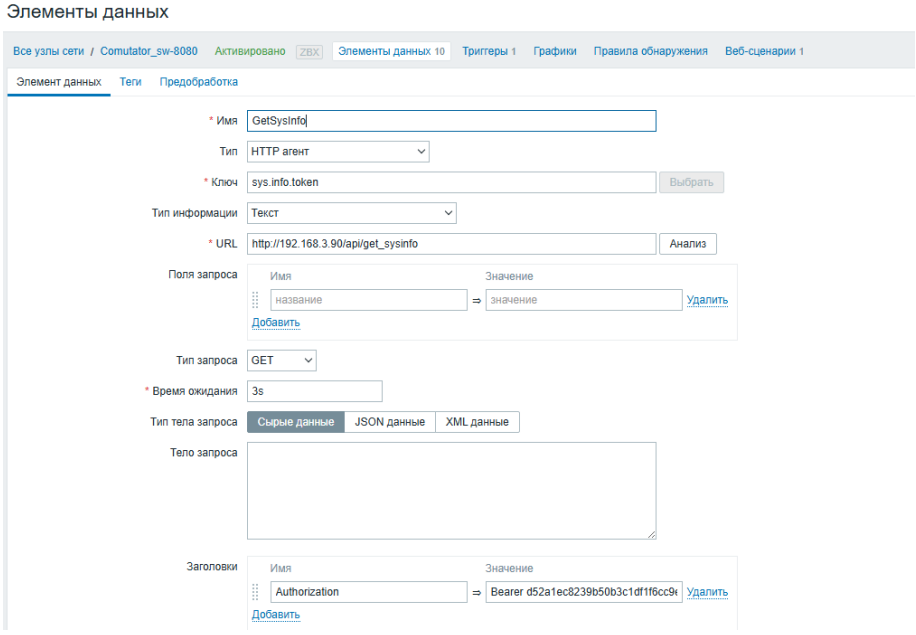


Рис.5 Ввод параметров элемента данных (устройства)

SEC.RU | Короткий путь к информации Скачано с sec.ru

Поля, отмеченные*, обязательны для заполнения.

Имя – название элемента данных;

Тип – HTTP агент;

Ключ – является идентификатором устройства (коммутатора);

Тип информации – Текст;

Интерфейс узла сети – остается пустым, если не был установлен на устройстве (коммутаторе);

URL – http запрос к устройству (коммутатору);

Тип запроса – получить;

Время ожидания – установлено по умолчанию;

Тип тела запроса – установлено по умолчанию;

Заголовки Имя – Authorization;

Заголовки Значение – Bearer (пробел) token* коммутатора (постоянный).

Для завершения нажать **Добавить** в правом нижнем углу окна. Данные от коммутатора будут поступать в автоматическом режиме, согласно сделанным настройкам.

* Для получения постоянного token следует отправить запрос коммутатору (например используя программу POST, Рис.6):

http://IP *коммутатора*/api/login - временный token выдается при авторизации;

http://IP *коммутатора*/api/get_permanent_token - получение постоянного token;

http://IP *коммутатора*/api/regenerate_permanent_token - генерация нового постоянного token.

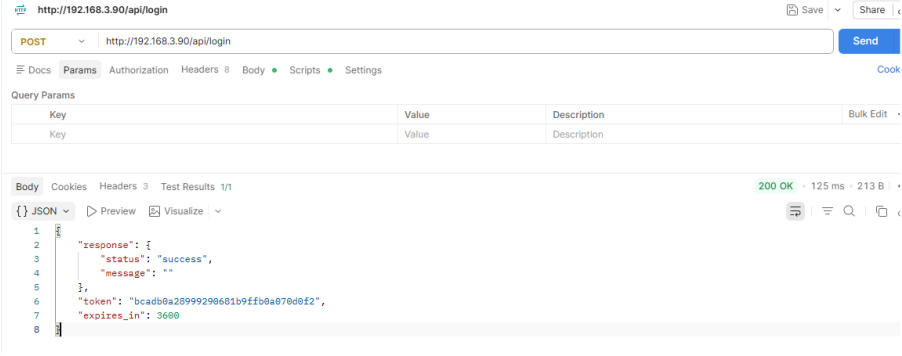


Рис.6 Получение постоянного token

SEC.RU | Короткий путь к информации
Скачано с sec.ru

Создание зависимого элемента данных

Зависимый элемент данных (Dependent Item) - это специальный тип элемента, который получает нужную часть данных (для конкретной метрики) из другого элемента - **мастер-элемента (Master Item)**.

Когда мастер-элемент возвращает новые данные (например, в формате JSON, XML или просто текст), зависимые элементы автоматически получают эти данные. Чтобы выделить нужную часть из общего массива данных, применяется предварительная обработка (препроцессинг). Для этого используются специальные инструменты:

JSON Path — для работы с JSON-данными.
XPath — для XML-структур.
Regex (Регулярные выражения) — для поиска и извлечения подстрок.

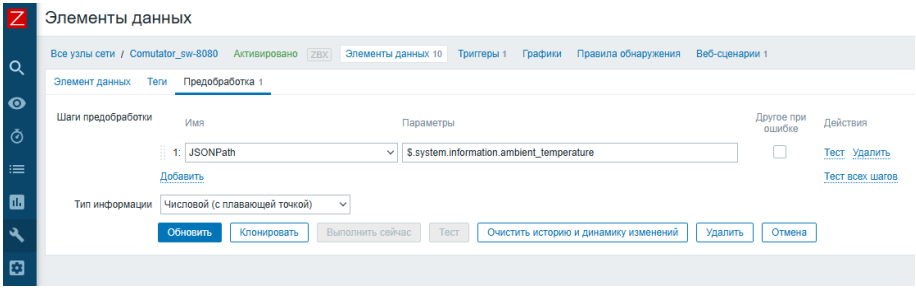


Рис.7 Создание зависимого элемента данных

Просмотр полученных данных

В разделе *Мониторинг* (Рис.8) выбрать *Последние данные*:

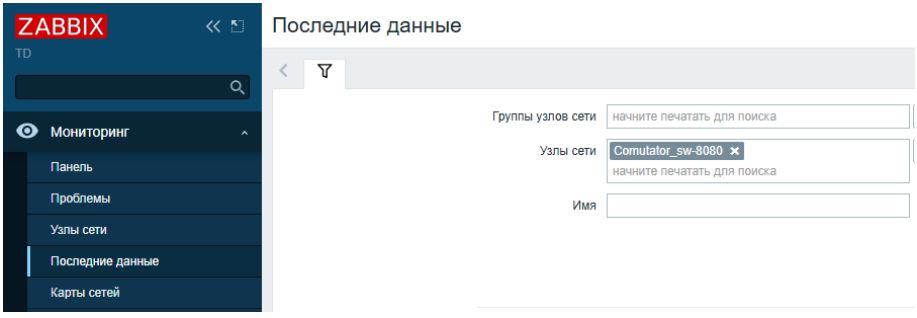


Рис.8 Раздел Мониторинг

SEC.RU | Короткий путь к информации
Скачано с sec.ru

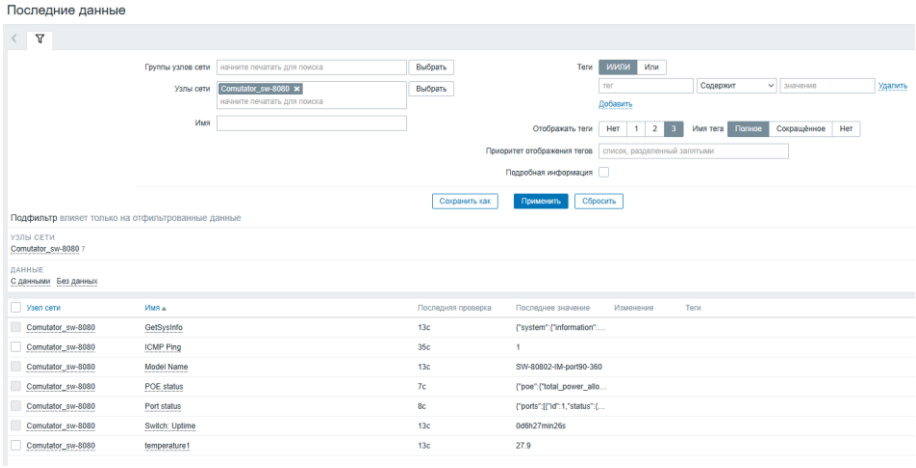


Рис.9 Список доступных данных коммутатора (колонка *Имя*)

Для просмотра интересующих данных в колонке *Имя* нажать нужный параметр (например, GetSySInfo Рис.10).

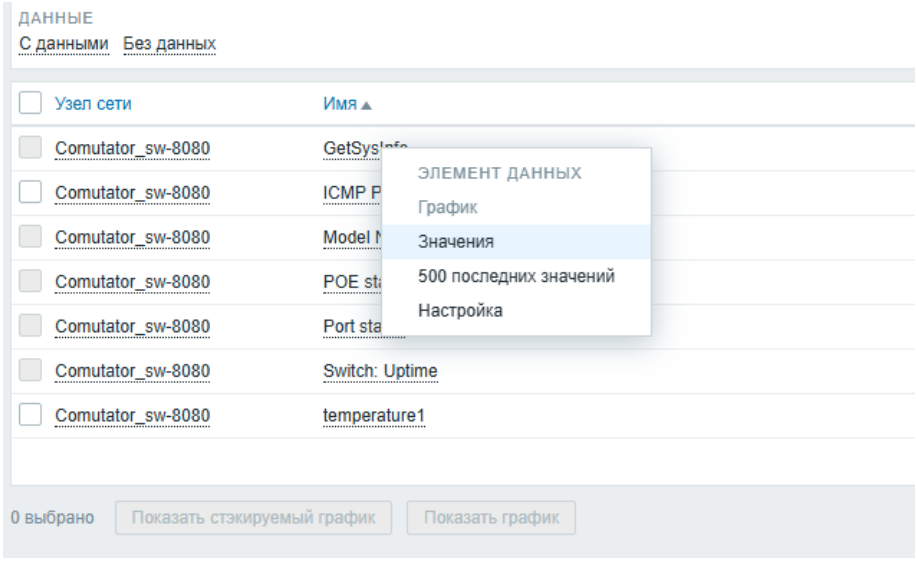


Рис.10 Выбор данных для просмотра/вывода

И выбрать нужное из выпадающего списка:

SEC.RU | Короткий путь к информации
Скачано с sec.ru

Данные будут представлены в текстовом виде (Рис.11):

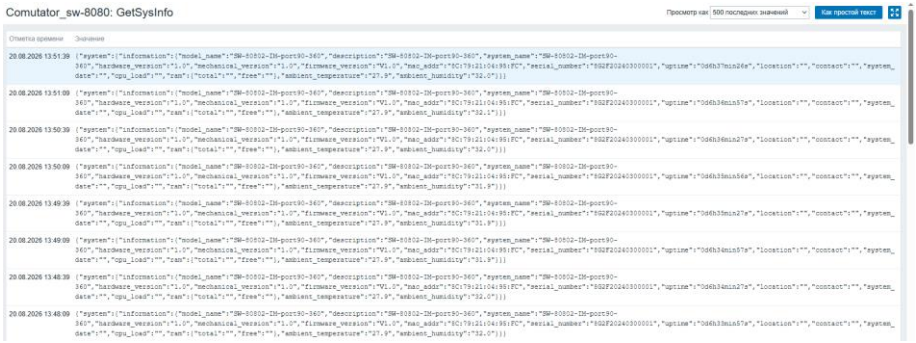


Рис.11 Вывод данных в текстовом виде

Данные могут быть представлены и в графическом виде, если это возможно для данного параметра (например значения температуры Рис.12).

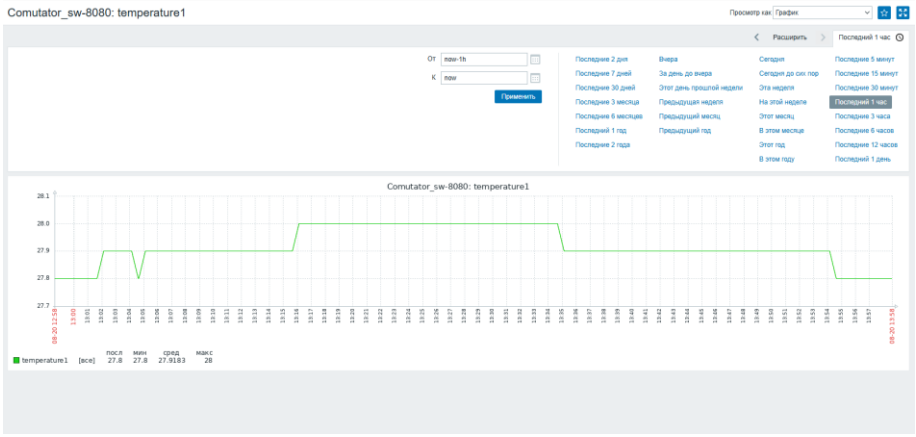


Рис.12 Вывод данных в графическом виде

SEC.RU | Короткий путь к информации
Скачано с sec.ru