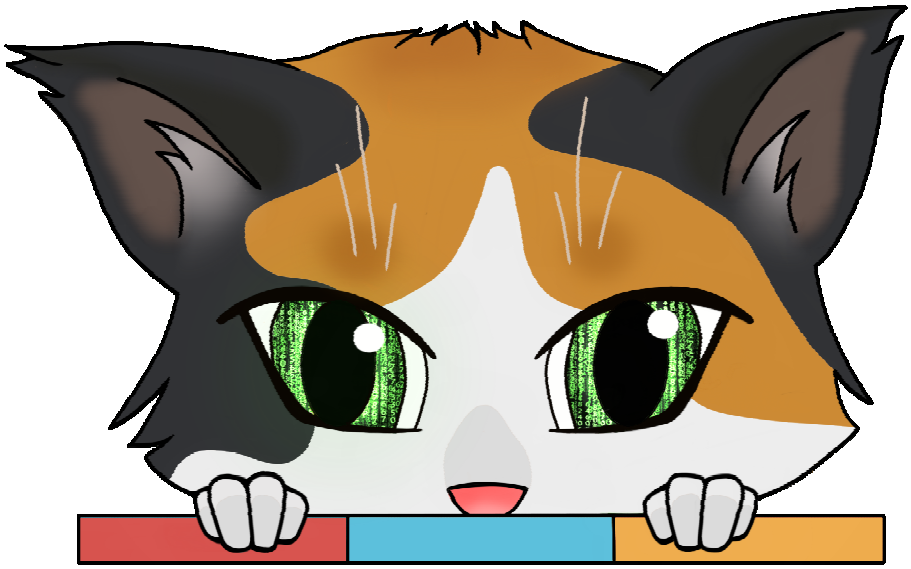
Network Systems Group

Богатка



Программно-определяемая глобальная сеть для массового подключения удалённых объектов

WEB API
Руководство разработчика

Версия 1.3.4

Обновлено 03.09.2020

Москва 2020

§ СОДЕРЖАНИЕ §

ООО «ЭН–ЭС–ДЖИ»
Россия 105187 Москва
ул. Вольная, д.35
Тел./факс: (495) 727–19–59 (многоканальный)

http://www.nsg.ru/
mailto:info@nsg.net.ru
mailto:sales@nsg.net.ru
mailto:support@nsg.net.ru

§1. Введение

§1.1. Общие сведения и термины

Web API Богатка позволяет сторонним приложениям получить информацию, содержащуюся в системе, по принципу "запрос — ответ". API работает посредством запросов HTTP и возвращает данные в формате JSON.

Метрикой в данном документе называется информация, посылаемая Богаткой в другие приложения.

Данный документ содержит описание Web API для разработчиков сторонних приложений.

§1.2. Богатка и Zabbix

Web API Богатка не имеет непосредственного отношения к взаимодействию между системами Богатка и Zabbix и может использоваться независимо от последнего. Наиболее существенное различие в работе совместно с Zabbix или без него состоит в том, что в Zabbix хранятся данные за некоторый установленный промежуток времени. Сама по себе Богатка всегда хранит для каждой метрики только одно последнее значение, и именно это значение выводится в её API при запросе метрики.

При наличии системы Zabbix сторонние приложения могут получать из неё данные как напрямую с помощью её собственного Web API, так и через посредство системы Богатка, которая может запрашивать информацию из Zabbix, обрабатывать её заданным образом при помощи скриптов (см. Руководство Администратора, раздел "Скрипты для Zabbix API") и отдавать её как частный случай данных.

§1.3. Источники данных и использование полученных данных

В качестве метрики из системы может быть получена любая информация, доступная на клиентских устройствах или в серверной части системы. В частности, можно выделить следующие категории метрик:

1. Датчики и скрипты на клиенте — генерируют данные по своему расписанию или по событиям. Для скриптов на клиенте уместо в данном контексте использовать термин *виртуальные датчики*, чтобы отличать их от скриптов, исполняемых сервером.

Эти данные, если предусмотрено в конфигурации, отсылаются на сервер Богатка, который является Zabbix-прокси. Дальнейшая судьба этих данных зависит от их типа и от конфигурации обеих систем:

— Метрики с ключем bogatka.* обрабатываются на серверной стороне специальным образом. Они могут делать что угодно, в зависимости от реализации обработчика. (В том числе, например, и посылать данные в Zabbix).

— Все остальные метрики ретранслируются в Zabbix. Если данная метрика указана в шаблоне Zabbix, то её значения сохраняются, отображаются на страницах Zabbix и могут быть истребованы сторонними приложениями, в т.ч. обратно Богаткой. Метрики, не указанные в шаблонах, игнорируются.

2. Скрипты Богатки. Выполнятся вручную пользователем, или по расписанию (см. Руководство Администратора, Планировщик задач). Могут:

— Загружать и исполнять на клиенте любые другие скрипты и получать с их помощью требуемую информацию. В частности, скрипты могут опрашивать вне расписания любые

физические и виртуальные датчики, имеющихся в конфигурации клиента, а могут вычислять требуемую информацию самостоятельно.

— Собирать любую информацию на стороне сервера.

— Запрашивать информацию из Zabbix и обрабатывать её заданным образом.

В зависимости от настроек скрипта, результат его исполнения также может передаваться в Zabbix и храниться там.

3. Сводные системные метрики — генерируются на сервере: число клиентов общее, по группам, по классам, по статусам и т.п. Всегда посылаются в Zabbix.

4. Системные метрики для клиента. Частично хранятся в БД конфигурации Богатка (принадлежность к классу, группам, параметры конфигурации), частично — в оперативной памяти сервера (статус, различные средние и суммарные показатели). В Zabbix по умолчанию не передаются, но теоретически могут.

Составить исчёрпывающий список метрик не представляется возможным, поскольку они могут предоставлять любую информацию, которая может быть измерена или вычислена на борту. Например:

— Если клиент оборудован датчиком геопозиционирования, то в состав метрик могут входить географические координаты.

— Если клиент имеет датчики температуры, напряжения, заряда батарей и т.п. в составе собственной конструкции, или в составе отдельных компонент (процессора, сотовых модулей), или в качестве внешних аксессуаров — их показания тоже могут быть метриками.

— Если скрипты, исполняемые на борту, дают на выходе какую-либо информацию (например, список MAC-адресов локальных хостов, или какой-то комплексный показатель качества связи) — это тоже метрика.

и т.д. Таким образом, набор возможных метрик практически безграничен, и при выборе передаваемых метрик следует исходить не из возможностей клиентского оборудования и системы в целом, а из потребностей запрашивающего приложения.

§2. Основы работы с Web API

§2.1. Аутентификация в системе

Прежде чем выполнять команды API, необходимо авторизоваться в системе. Для авторизации следует использовать имя и пароль пользователя, имеющего права администратора системы. Используется авторизация JWT (JSON Web Token).

Далее все примеры приводятся с использованием команды *curl* в оболочке *bash*. Для работы с API из программ следует использовать соответствующие библиотеки (они есть во всех современных языках программирования).

Для авторизации необходимо выполнить запрос HTTP POST в следующем формате:

```
$ curl -k -H "Content-Type: application/json" -X POST \
   -d '{"username":"admin", "password":"admin"}' https://my.bogatka.ru:port/api/sign_in
```

где имя и пароль пользователя, адрес и порт будут соответствовать реальной системе. При успешной авторизации будет получен ответ в формате JSON следующего вида:

```
{"jwt":"eyJhbGciOiJIUzUx...","username":"admin"}
```

Здесь jwt — это токен для авторизации последующих запросов.

В случае неудачной авторизации получим:

```
$ curl -k -H "Content-Type: application/json" -X POST \
   -d '{"username":"admin", "password":"nopsw"}' https://my.bogatka.ru:port/api/sign_in

{"error":{"message":"invalid_credentials"}}
```

§2.2. Выполнение команд

Для выполнения команды API надо послать запрос HTTP POST с указанием метода и, опционально, параметрами. Пример:

```
$ curl -k -H "Authorization: Bearer eyJhbGciOiJIUzUx..." \
   -H "Content-Type: application/json"  \
   -X POST -d '{"method":"get.clients_list","params":{"group": "Серые"}}' \
   https://my.bogatka.ru:port/api/bogatka
```

где в поле авторизации указывается токен, полученный в запросе на авторизацию.

Метод get.clients_list, использованный в данном примере, с параметром group возвращает список всех клиентов, которые принадлежат к указанной группе. Успешный ответ записывается в формате JSON с ключем result (отформатировано для удобства чтения):

```
{
   "result":{
      "clients_list":[
         {"address":null,"description":"ATM11","lock":false,"name":"NSG1810_1410001547"},
         {"address":null,"description":"ATM12","lock":false,"name":"NSG1810_1410001357"},
         {"address":null,"description":"ATM34","lock":false,"name":"NSG1810_1410001494"},
         {"address":null,"description":"ATM15","lock":false,"name":"NSG1810_1410001382"},
         {"address":null,"description":"ATM36","lock":false,"name":"NSG1810_1410001411"}
      ],"group":"Серые"
   }
}
```

Если возникает какая-либо ошибка, то возвращается запись с ключем error:

```
$ curl -k -H "Authorization: Bearer eyJhbGciOiJIUzUx…" \
   -H "Content-Type: application/json"  \
   -X POST -d '{"method":"get.clients_list","params":{"group": "Голубые"}}'  \
   https://my.bogatka.ru:port/api/bogatka
{"error":{"data":"Голубые","message":"no such group"}}
```

Если будет указан неверный токен или истечет его время жизни, то вернется ошибка invalid_token:

```
$ curl -k -H "Authorization: Bearer eyJhbGciOiJIUzUx…" \
   -H "Content-Type: application/json"  \
   -X POST -d '{"method":"get.clients_list","params":{"group": "Голубые"}}'  \
   https://my.bogatka.ru:port/api/bogatka
{"error":{"message":"invalid_token"}}
```

§3. Методы Web API

§3.1. Метод get.clients_list

Метод get.clients_list возвращает список клиентов, входящих в заданную группу. Формат запроса:

```
{
   "method":"get.clients_list",
   "params":{
      "group": <группа клиентов>
   }
}
```

В параметре group можно указать имя группы, её идентификатор или true:

"group": "Серые"	Выводит список клиентов для конкретной группы
"group": 12	Выводит список клиентов для конкретной группы
"group": true	Выводит список всех клиентов

Ответ содержит список клиентов и имя группы. Для каждого клиента возвращаются 4 ключа (в алфавитном порядке): адрес, описание, статус блокировки и системное имя. Пример вывода (отформатировано):

```
{
   "result":{
      "clients_list":[
         {"address":null,"description":"ATM11","lock":false,"name":"NSG1810_1410001547"},
         {"address":null,"description":"ATM12","lock":false,"name":"NSG1810_1410001357"},
         {"address":null,"description":"ATM34","lock":false,"name":"NSG1810_1410001494"},
         {"address":null,"description":"ATM15","lock":false,"name":"NSG1810_1410001382"},
         {"address":null,"description":"ATM36","lock":false,"name":"NSG1810_1410001411"}
      ],"group":"Серые"
   }
}
```

§3.2. Метод get.metrics

Метод get.metrics возвращает заданные метрики для заданного клиента, списка клиентов, или группы. Формат запроса:

```
{
   "method":"get.metrics",
   "params":{
      "clients": <список имён клиентов>,
      "group": <группа клиентов>,
      "metrics": <список метрик>
   }
}
```

Если в списке всего один клиент, то квадратные скобки можно опустить:

```
      "clients": "NSG1810_1410001354"
```

Параметр group аналогичен таковому в методе get.clients_list. В запросе обязательно должен быть хотя бы один из параметров clients или group. Если указаны оба, то параметр group игнорируется.

В списке метрик могут использоваться имена скриптов и ключи датчиков. Имена скриптов определяются на сервере Богатка (см. Руководство Администратора, Шаблоны). Ключи датчиков либо определяются в конфигурации клиента, либо указываются в настройках скрипта на сервере. (В последнем случае метрика содержит тот же результат, что и метрика с именем самого этого скрипта.)

Ответ содержит список клиентов с метриками. Элемент списка имеет вид:

```
{ "client": <имя клиента>,
   "metrics": <список метрик>
}
```

Элемент списка метрик имеет вид:

```
{
   "name": <имя метрики (скрипт или ключ датчика)>,
   "data": <значение метрики>,
   "source": <источник>,
   "timestamp": <отметка времени>
}
```

где <источник> принимает значение script либо zabbix.

Пример (отформатировано для удобства чтения):

```
curl -k -H "Authorization: Bearer eyJhbGciOiJIUzUx..." \
   -H "Content-Type: application/json"  \
   -X POST \
   -d '{
      "method":"get.metrics",
      "params":{
         "clients":["NSG1810_1410001354", "NSG1810_1410001376"],
         "metrics":["m2_oper", "csq[m2]"]
         }
      }'  \
   https://my.bogatka.ru:port/api/bogatka
```

В данном случае m2_oper — это имя скрипта, определённого на сервере Богатка, csq[m2] — имя датчика, определённого в конфигурации клиента.

Возвращает (отформатировано):

```
{
   "result": [
      {
         "client":"NSG1810_1410001354",
         "metrics":[
            {"data":"MTS\n", "name":"m2_oper", "source":"script", "timestamp":1598745611},
            {"data":"18", "name":"csq[m2]", "source":"zabbix", "timestamp":1598777829}
         ]
      },
      {
         "client":"NSG1810_1410001376",
         "metrics":[
            {"data":"MEGAFON\n","name":"m2_oper","source":"script","timestamp":1598745611},
            {"data":"22","name":"csq[m2]","source":"zabbix","timestamp":1598777790}
         ]
      }
   ]
}
```

§3.3. Метод get.ai_notes

Метод get.ai_notes возвращает сообщения искусственного интеллекта (ИИ) для заданного клиента, списка клиентов, или группы. Формат запроса:

```
{
   "method":"get.metrics",
   "params":{
      "clients": <список имён клиентов>,
      "group": <группа клиентов>,
   }
}
```

Параметры clients и group используются точно так же, как и в методе get.metrics .

Пример (отформатировано):

```
curl -k -H "Authorization: Bearer eyJhbGciOiJIUzUx..." \
   -H "Content-Type: application/json"  \
   -X POST \
   -d '{
      "method":"get.metrics",
      "params":{
         "clients":["NSG1810_1410001495","NSG1810_1410001397"]
         }
      }' \
   https://my.bogatka.ru:port/api/bogatka
```

Возвращает (отформатировано):

```
{  "result": [
   {
      "client": "NSG1810_1410001495",
      "ai_notes": [{
         "message":"Сотовый модуль не стартует по неизвестной причине.",
         "port":"m2"
      }]
   },
   {
      "client": "NSG1810_1410001397",
      "ai_notes": [{
         "message":"SIM-карта не найдена ни в одном слоте.",
         "port":"m1
      }]
   }
]}
```