API для провайдеров

Пользователь

`[user_id]` - идентификатор пользователя провайдера. Может содержать символы A-Z, a-z, 0-9 подчеркивание и тире.

Максимальная длинна 32 символа.

Создание пользователя

Новый пользователь создается при попытки создать для него какое-либо свойство: указать категории, присвоить ip.

Удаление пользователя

```
DELETE /users/[user_id]
```

Проверка пользователя на существование

```
HEAD /users/user1
```

В случае успеха возвращает код HTTP ответа 200. Если пользователь не найден – возвращает код HTTP ответа 404

Получение списка активных пользователей

```
GET /active_users/
```

возвращает ответ в формате

```
["geek", "bugtest", "hammer"]
```

Под активными подразумеваются пользователи, у которых есть хотя бы один IP-адрес.

Получение списка всех пользователей

```
GET /users
```

возвращает ответ в формате

```
[{
  "name": "geek",
  "safesearch": "off",
  "safeyoutube": "off",
  "status": "enabled",
  "filter": [1, 2],
  "ip": ["192.168.5.6", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"],
  "whitelist": ["nic.ru", "yandex.ru"],
  "blacklist": []
},{
  "name": "bugtest",
  "safesearch": "off",
  "safeyoutube": "on",
  "status": "enabled",
  "filter": [1, 2, 3],
  "ip": ["192.168.5.8"],
  "whitelist": ["ya.ru", "google.com"],
  "blacklist": []
}]
```

По умолчанию возвращает первые 100 пользователей.

```
GET /users?start=100&stop=200
```

Параметры start и stop указывают на то, что надо вернуть пользователей начиная с 100 и до 200

Поиск пользователя

```
GET /search/user1
```

возвращает ответ в формате

```
[{
  "name": "user1",
  "safesearch": "off",
  "safeyoutube": "off",
  "status": "enabled",
  "filter": [],
  "ip": [],
  "whitelist": [],
  "blacklist": []
}]
```

ищет пользователя user1 по точному совпадению имени или ip адреса

Поиск возможен по ip или имени пользователя. В поиске можно использовать маски.

```
GET /search/192.168.0.*
```

Вернет всех пользователей, ip адрес которых начинается с 192.168.0.

```
GET /search/user*
```

Вернет всех пользователей, имя которых начинается с user

Получение информации по пользователю:

```
GET /users/[user_id]
```

возвращает ответ в формате

```
{
  "name": "[user_id]",
  "safesearch": "off",
  "safeyoutube": "off",
  "status": "enabled",
  "filter": [3, 4, 5, 6],
  "ip": ["192.168.5.6", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"],
  "whitelist": ["yandex.ru", "nic.ru"],
  "blacklist": []
}
```

Обновление информации о пользователе

```
PUT /users/[user_id]

Content-type: application/json

{
  "name": "[user_id]"
  "safesearch": "on",
  "safeyoutube": "off",
  "status": "enabled",
  "filter": [2, 3, 4, 5],
  "ip": ["192.168.5.8", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"],
  "whitelist": ["yandex.ru", "nic.ru"],
  "blacklist" ["mail.ru", "pornhub.com"]
}
```

Отключение фильтрации

Пользователь может отключить фильтрацию без сброса настроек

```
POST /users/[user_id]/status/disabled

Content-type: application/json
```

Включение фильтрации

```
POST /users/[user_id]/status/enabled

Content-type: application/json
```

Безопасный поиск

Если опция **Безопасный поиск** включена и пользователь делает запрос к поисковой системе, то в ответ он получает специальную страницу блокировки, которая осуществляет перенаправление на URL. URL безопасного поиска указывается в файле конфигурации. По умолчанию это http://search.skydns.ru

Включение безопасного поиска по умолчанию

```
PUT /config/

Content-type: application/json

{ "safesearch": true }
```

Отключение безопасного поиска по умолчанию

```
PUT /config/

Content-type: application/json

{ "safesearch": false }
```

Получение настройки безопасного поиска по умолчанию

```
GET /config/
```

формат ответа:

```
{
  "filter": [2, 3, 4, 5],
  "safesearch": true,
  "safeyoutube": false,
  "safesearchredirect": "http://search.skydns.ru"
}
```

Включение безопасного поиска по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "safesearch": true }
```

Отключение безопасного поиска по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "safesearch": false }
```

Получение настройки безопасного поиска по умолчанию для пользователей

```
GET /userconfig/
```

формат ответа:

```
{
  "filter": [3, 4, 5, 6],
  "safesearch": true,
  "safeyoutube": false
}
```

Включение безопасного поиска

```
POST /users/[user_id]/safesearch/on

Content-type: application/json
```

Отключение безопасного поиска

```
POST /users/[user_id]/safesearch/off

Content-type: application/json
```

Установка адреса перенаправления безопасного поиска

```
PUT /config/

Content-type: application/json

{ "safesearchredirect": "http://search.skydns.ru" }
```

Очистка адреса перенаправления безопасного поиска

```
PUT /config/

Content-type: application/json

{ "safesearchredirect": "" }
```

Безопасный YouTube

Если эта опция включена, пользователь будет автоматически перенаправляться на безопасную версию YouTube.

Включение безопасного YouTube по умолчанию

```
PUT /config/

Content-type: application/json

{ "safeyoutube": true }
```

Отключение безопасного YouTube по умолчанию

```
PUT /config/

Content-type: application/json

{ "safeyoutube": false }
```

Включение безопасного YouTube по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "safeyoutube": true }
```

Отключение безопасного YouTube по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "safeyoutube": false }
```

Адреса пользователя

Получение ip пользователя

```
GET /users/[user_id]/ip/
```

возвращает ответ в формате

```
["192.168.5.6", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"]
```

Добавление ip пользователя (происходит добавление к имеющимся)

```
POST /users/[user_id]/ip/

Content-type: application/json

["127.0.0.1"]
```

или списка адресов

```
POST /users/[user_id]/ip/

Content-type: application/json

["127.0.0.1", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"]
```

Сохранение/обновление ip пользователя (происходит замещение имеющихся)

```
PUT /users/[user_id]/ip/

Content-type: application/json

["127.0.0.1"]
```

или списка адресов

```
PUT /users/[user_id]/ip/

Content-type: application/json

["127.0.0.1", "2001:0db8:85a3:0000:0000:8a2e:0370:7334"]
```

Удаление всех ip пользователя и отключение его от сервиса

```
DELETE /users/[user_id]/ip/
```

Удаление одного из ip пользователя

```
DELETE /users/[user_id]/ip/[127.0.0.1]
```

Общая информация

Получение группированного списка категорий с названиями и идентификаторами категорий

```
GET /categorygroups/
```

ответ в формате

```
[{
  "group": "Общие",
  "categories": {
    "3": "Сайты, распространяющие вирусы",
    "4": "Фишинг",
    ...
  }
}, {
  "group": "Черные сайты",
  "categories": {
    "6" : "Наркотики",
    ...
  }
},
...
]
```

Получение списка категорий с названиями и идентификаторами категорий

```
GET /categories/
```

возвращает ответ в формате json

```
{
  "3": "Сайты, распространяющие вирусы",
  "4": "Фишинг",
  ...
}
```

Получение категорий сайта

```
GET /site/login.vk.com
```

Пример ответа:

```
{
  "domain": "vk.com",
  "categories": [29],
}
```

Здесь vk.com - это домен, который нашелся в нашей базе, а [29] - это список категорий (в данном случае состоящий из одной категории **социальные сети**). В дальнейшем могут добавляться новые поля.

Категории для блокировки

Добавление категории в список блокируемых по умолчанию

```
PUT /config/

Content-type: application/json

{ "filter": [1, 2] }
```

Удаление категории из списка блокируемых по умолчанию

```
PUT /config/

Content-type: application/json

{ "filter": [] }
```

Добавление категории в список блокируемых по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "filter": [1, 2] }
```

Удаление категории из списка блокируемых по умолчанию для пользователей

```
PUT /userconfig/

Content-type: application/json

{ "filter": [] }
```

Получение списка категорий пользователя

```
GET /users/[user_id]/filter/
```

ответ в формате

```
[1, 2, 3]
```

Добавление категории в список пользователя

```
POST /users/[user_id]/filter/

Content-type: application/json

[1, 2]
```

Сохранение нового списка категорий пользователя

```
PUT /users/[user_id]/filter/

Content-type: application/json

[1, 2, 3]
```

Отключение отдельной категории для пользователя

```
DELETE /users/[user_id]/filter/2
```

Отключение всех категорий для пользователя

```
DELETE /users/[user_id]/filter/
```

Черный список

Получение черного списка

```
GET /users/[user_id]/blacklist/
```

ответ в формате

```
["www.ya.ru"]
```

Добавление домена в черный список

```
POST /users/[user_id]/blacklist/

Content-type: application/json

["www.ya.ru"]
```

Замена черного списка новым

```
PUT /users/[user_id]/blacklist/

Content-type: application/json

["www.ya.ru"]
```

Удаление домена из черного списка

```
DELETE /users/[user_id]/blacklist/www.ya.ru
```

Удаление черного списка

```
DELETE /users/[user_id]/blacklist/
```

Установка опции Работать только по белому списку

Для работы пользователя в режиме запрещено все, кроме того, что явно занесено в белый список, нужно занести корневой домен в черный список:

```
POST /users/[user_id]/blacklist/

Content-type: application/json

["-"]
```

Для удаления корневого домена:

```
DELETE /users/[user_id]/blacklist/-
```

Белый список

Получение белого списка

```
GET /users/[user_id]/whitelist/
```

ответ в формате

```
["www.ya.ru"]
```

Добавление домена в белый список

```
POST /users/[user_id]/whitelist/

Content-type: application/json

["www.ya.ru"]
```

Замена белого списка новым

```
PUT /users/[user_id]/whitelist/

Content-type: application/json
```

```
["www.ya.ru"]
```

Удаление домена из белого списка

```
DELETE /users/[user_id]/whitelist/www.ya.ru
```

Удаление белого списка

```
<pre>DELETE /users/[user_id]/whitelist/
```

Глобальный черный список (имеет приоритет над списками пользователя)

Получение глобального черного списка

```
GET /blacklist/
```

ответ в формате

```
["www.ya.ru"]
```

Добавление домена в глобальный черный список

```
POST /blacklist/

Content-type: application/json

["www.ya.ru"]
```

Замена глобального черного списка новым

```
PUT /blacklist/

Content-type: application/json

["www.ya.ru"]
```

Удаление домена из глобального черного списка

```
DELETE /blacklist/www.ya.ru
```

Удаление глобального черного списка

```
DELETE /blacklist/
```

Установка опции Работать только по глобальному белому списку

Для работы пользователя в режиме запрещено все, кроме того, что явно занесено в белый список, нужно занести корневой домен в черный список:

```
POST /blacklist/

Content-type: application/json

["-"]
```

Для удаления корневого домена:

```
DELETE /blacklist/-
```

Глобальный белый список (имеет приоритет над списками пользователя)

Получение глобального белого списка

```
GET /whitelist/
```

ответ в формате

```
["www.ya.ru"]
```

Добавление домена в глобальный белый список

```
POST /whitelist/

Content-type: application/json

["www.ya.ru"]
```

Замена глобального белого списка новым

```
PUT /whitelist/

Content-type: application/json

["www.ya.ru"]
```

Удаление домена из глобального белого списка

```
DELETE /whitelist/www.ya.ru
```

Удаление глобального белого списка

```
DELETE /whitelist/
```

Статистика

Активность пользователя по часам

Где [date_from], [date_to] даты в формате YYYY-MM-DD

```
GET /users/[user_id]/stat/activity/hour?from=[date_from]&to=[date_to]

{
  "labels": ["2016-06-29 14:00:00", "2016-06-29 15:00:00"],
  "datasets": [{
    "label": "Requests",
    "data": [375, 275]
  },{
    "label": "Blocks",
    "data": [13, 0]
  }]
}
```

Где label - список временных интервалов, datasets список объектов содержащих данные для графиков и легенды графиков, data - список значений соответствующих временным интервалам, label - название графика (Requests - кол-во запросов, Blocks - кол-во блокировок)

Активность пользователя по дням

Где [date_from], [date_to] даты в формате YYYY-MM-DD

```
GET /users/[user_id]/stat/activity/day?from=[date_from]&to=[date_to]

{
  "labels": ["2016-06-29", "2016-06-30"],
  "datasets": [{
    "label": "Requests",
    "data": [5770, 3456]
  },{
    "label": "Blocks",
    "data": [8, 9]
  }]
}
```

Где labels - список временных интервалов, datasets - список объектов содержащих данные для графиков и легенды графиков, data - список значений соответствующих временным интервалам, label - название графика (Requests - кол-во запросов, Blocks - кол-во блокировок)

Домены

```
GET /users/[user_id]/stat/domains/[filter]?from=[date_from]&to=[date_to]

{
  "labels": ["example.com", "google.com", "asdfg.com"],
  "datasets": [{
    "label": "Requests",
    "data": [630, 474, 290]
  },{
    "label": "NXdomain",
    "data": [0, 0, 290]
  },{
    "label": "Blocks",
    "data": [630, 0, 0]
  }]
}
```

Где [date_from], [date_to] даты в формате YYYY-MM-DD

[filter] может принимать значения all (возвращает все домены), www (возвращает только все домены, которые начинаются с www). labels - список доменов, datasets - наборы данных, data - список значений соответствующих каждому домену, label - название графика (Requests - кол-во запросов, Blocks - кол-во блокировок, NXdomain - кол-во ответов домен не существует)

Детальная

```
GET /users/[user_id]/stat/detail?from=[date_from]&to=[date_to]

{
  "timestamps": ["2016-06-29 15:00:00", "2016-06-29 16:00:00"],
  "reports": [{
    "labels": ["clients4.google.com", "www.google.ru"],
    "cats": [[48, 251], [48]],
    "datasets": [{
      "label": "Requests",
      "data": [29, 20]
    },{
      "label": "NXdomain",
      "data": [0, 0]
    },{
      "label": "Blocks",
      "data": [0, 0]
    }]
  },{
    "labels": ["live.github.com", "lb._dns-sd._udp.0.0.200.10.in-addr.arpa", "ssl.gstatic.com"],
    "cats": [[2], [2], [2]],
    "datasets": [{
      "label": "Requests",
      "data": [73, 48, 48]
    },{
      "label": "NXdomain",
      "data": [0, 48, 0]
    },{
```

```
      "label": "Blocks",
      "data": [0, 0, 0]
    }]
  }]
}
```

Где [date_from], [date_to] даты в формате YYYY-MM-DD

timestamps - список временных интервалов reports - список отчетов соответствующих временным интервалам labels - список доменов cats - списки категорий для доменов datasets - наборы данных data - список значений соответствующих каждому домену label - название графика (Requests - кол-во запросов, Blocks - кол-во блокировок, NXdomain - кол-во ответов домен не существует)

Статистика без агрегации

Полная статистика пользователя за последний час. Обновляется каждые пять минут.

```
GET /users/[user_id]/stat/raw

{
  "fields": ["created", "nxdomain", "address", "host", "blockedhost", "reasom", "cats", "blocked
  "data": [
    ["2016-07-11 17:50:26", "0", "10.200.1.88", "www.tns-counter.ru", "www.tns-counter.ru", "4",
    ["2016-07-11 17:50:26", "0", "10.200.1.88", "www.tns-counter.ru", "www.tns-counter.ru", "4",
  ]
}
```

Коды ответов

`200` OK - Успешный запрос

`201` Created - Успешный запрос создание элемента

`204` No Content - Успешный запрос с пустым ответом

`400` Bad Request - Некорректный запрос

`404` Resource is not found - Ресурс не существует

`422` Unprocessable Entity - Запрос не содержит требуемые поля

`500` Internal server error - Внутренняя ошибка сервера

Revision #5
Created 8 December 2023 11:35:28 by Виктор
Updated 11 December 2023 12:57:17 by Виктор