



Платформа Радар

Описание публичного API

Версия 3.6.7

Оглавление

1. Общие сведения	3
2. Особенности работы с API через Karaken	4
2.1. Получение списка подключенных инстансов	4
2.2. Выполнение запросов к подчиненным инстансам	5
3. Использование стандартных фильтров	7
4. Описание методов API	9
4.1. Инциденты	9
4.1.1. Получение списка типов инцидентов	9
4.1.2. Получение списка инцидентов	11
4.1.3. Обновление статуса инцидентов	21
4.1.4. Обновление уровня рисков инцидентов	22
4.1.5. Создание инцидента	22
4.2. Активы	24
4.2.1. Получение списка активов	24
4.2.2. Получение перечня групп активов	25
4.3. Пользователи	26
4.3.1. Получение списка пользователей	26
4.4. Хранилища	27
4.4.1. Получение списка хранилищ	27
4.4.2. Получение информации о хранилище	28
4.4.3. Обновление информации о хранилище	29
4.4.4. Получение данных из хранилища в формате JSON	30
4.4.5. Получение всех ключей из хранилища	31
4.4.6. Установление значения по ключу	31
4.4.7. Удаление ключа	32
4.4.8. Удаление хранилища	33
4.4.9. Очистка хранилища	33
4.4.10. Создание хранилища	34
4.5. Сервисы	35
4.5.1. Проверка состояния сервисов	35

1. Общие сведения

API для внешних потребителей используется для интеграции внешних потребителей с **Платформой Радар** на программном уровне.

Для авторизации в методах используется доверенный ключ API.

Доверенные ключи API используются для межсервисного взаимодействия. Управление ключами API реализовано в веб-интерфейсе, на вкладке **Кластер - API ключи** (см. рисунок 1).

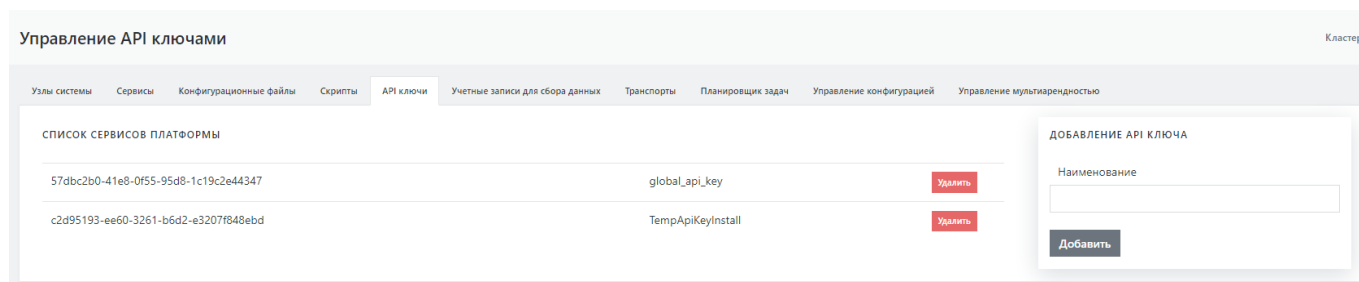


Рисунок 1 – Управление ключами API

Для вызова методов вам понадобится значение *global_api_key*.

Не рекомендуется удалять ключ *global_api_key* во избежание потери работоспособности **Платформы Радар**.

2. Особенности работы с API через Karaken

Платформа Радар, начиная с версии 3.2.0, позволяет выбирать инстанс, к которому должен обращаться метод API.

MASTER_KARAKEN_HOST - это основная точка входа для обработки запросов, для обращений API используются порты 9000 и 9009.

В случае использования порта 9000 обращение идет к основному инстансу.

В случае использования порта 9009 обращение идет к указанному инстансу. В этом случае в заголовке передается параметр PgrSelectedInstance в значении которого указывается идентификатор инстанса.

2.1. Получение списка подключенных инстансов

Тип метода: GET

Метод:

`https://<MASTER_KARAKEN_HOST>:9000/karaken/instance/list`

Заголовки (headers):

`PgrApiKey: api_key_string`

`Content-Type: application/json`

Входные параметры: отсутствуют

Выходные параметры:

Параметр	Описание
id	Идентификатор инстанса
name	Наименование инстанса
url	URL инстанса с номером используемого порта
sort_order	Порядок сортировки

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/karaken/instance/list' \  
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \  
--header 'Content-Type: application/json' \  

```

Пример ответа:

```
[
  {
    "id": "38c62166-0a5e-f51a-d583-6de7c200ef8a",
    "name": "172.30.254.70",
    "url": "https://127.0.0.1:9000",
    "sort_order": 0
  },
  {
    "id": "38c62166-0a5e-f51a-d583-6de7c200ef9a",
    "name": "172.30.254.70 (2)",
    "url": "https://127.0.0.1:9000",
    "sort_order": 0
  }
]
```

2.2. Выполнение запросов к подчиненным экземплярам

Для выполнения запроса к подчиненному экземпляру необходимо в заголовке, помимо APIKey, передать идентификатор требуемого экземпляра. В этом случае метод будет обработан указанным экземпляром.

Например, если сделать запрос к получению списка инцидентов с указанием идентификатора экземпляра из списка экземпляров в заголовке PgrSelectedInstance: <string>:

```
GET https://<MASTER_KARAKEN_HOST>:9009/cruddy/public/api/v1/incidents?
page=1&per_page=1&order=id%20DESC

Headers:
PgrApiKey: api_key_string
PgrSelectedInstance: 38c62166-0a5e-f51a-d583-6de7c200ef9a
Content-Type: application/json
```

то будет получен ответ:

```
[
  {
    "objects": [
      {
        "id": "c017287c-7e30-42c2-bcfd-d62e5e9d71b5",
        "description": "..."
      },
      {
        "id": "c017287c-7e30-42c2-bcfd-d62e5e9d71b6",
        "description": "..."
      }
    ],
    "total": 2
  }
]
```

Как видно из ответа, компонент Karaken возвращает массив объектов, содержащих в себе детали запроса в подчиненный экземпляр. Указание конкретного экземпляра приводит к ограничению количества элементов массива к одному конкретно указанному экземпляру.

Если же не указать заголовок PgrSelectedInstance, то ответ будет содержать в себе все подмножество подключенных экземпляров.

В таком режиме не рекомендуется использовать фильтрацию или пагинацию, т.к. данные среди всех инстансов могут не совпадать. При этом, при указании конкретного инстанса с помощью заголовка `PgrSelectedInstance` возможно осуществлять все API операции через мастер сервер со всеми подчиненными инстансами. Разница будет лишь в том, что формат ответа будет отличаться от прямого взаимодействия, а само тело ответа будет находиться в ключе `data` внутри элемента массива запрашиваемого инстанса.

3. Использование стандартных фильтров

Многие описанные ниже методы поддерживают стандартные фильтры - возможность отфильтровать возвращаемые данные по выходному параметру по условиям:

- **like**. Позволяет отфильтровать по вхождению в параметре. Пример:

```
?id=like,31549173
```

Будут найдены все идентификаторы, содержащие 31549173

- **gt**. Позволяет отфильтровать по значению параметра, больше указанного значения. Пример:

```
?created_at=gt,2020-01-01
```

Будут отображены записи, созданные после 2020-01-01

- **lt**. Позволяет отфильтровать по значению параметра, меньше указанного значения. Пример:

```
?created_at=lt,2020-01-01
```

Будут отображены записи, созданные до 2020-01-01

- **in**. Позволяет отфильтровать по вхождению параметра в список значений. Пример:

```
?id=in,31549173-fa4c-4a5c-a9e1-1107dbb8a48d,31549173-fa4c-4a5c-a9e1-1107dbb8a49d
```

Будут отображены идентификаторы, входящие в указанный список.

- **notin**. Позволяет отфильтровать по исключению вхождения параметра в список значений. Пример:

```
?id=notin,31549173-fa4c-4a5c-a9e1-1107dbb8a48d,31549173-fa4c-4a5c-a9e1-1107dbb8a49d
```

Будут отображены идентификаторы, не входящие в указанный список.

- **eq**. Позволяет задать точное соответствие параметра значению. Пример:

```
?id=eq,31549173-fa4c-4a5c-a9e1-1107dbb8a48d
```

Будет найден идентификатор с указанным значением.

- **noteq**. Позволяет задать исключение точного соответствия параметра значению. Пример:

```
?id=noteq,31549173-fa4c-4a5c-a9e1-1107dbb8a48d
```

Будут найдены идентификаторы кроме идентификатора с указанным значением.

- **isnull**. Позволяет отфильтровать по пустому значению параметра. Пример:

```
?fileid=isnull
```

Будут отображены записи с пустыми идентификаторами файлов.

- **isnotnull**. Позволяет отфильтровать по непустому значению параметра. Пример:

```
?fileid=isnotnull
```

Будут отображены записи с непустыми идентификаторами файлов.

Для некоторых методов необходимо указывать имя параметра с именем сущности через точку.
Например `?finding.created_at=gt,2020-01-01`

Также в query строке запроса можно указать дополнительные параметры: `page` (номер страниц), `per_page` (количество строк в странице) для пагинации, `order` для сортировки, `search` для поиска по нескольким полям сущности.

4. Описание методов API

4.1. Инциденты

4.1.1 Получение списка типов инцидентов

Тип метода: GET

Метод:

`https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/findings`

Заголовки (headers):

`PgrApiKey: api_key_string`

`Content-Type: application/json`

Входные параметры: поддерживаются стандартные фильтры

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/public/api/v1/findings' \  
--header 'PgrApiKey: 500000000-4000-0000-9000-1000000000000' \  
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{  
  "objects": [  
    {  
      "title": "MS-WIN-Изменение политики аудита",  
      "description": "На одном или нескольких узлах обнаружены изменения политики аудита. Это событие может указывать на несанкционированную деятельность злоумышленника. Поэтому к нему следует относиться с большим вниманием: подобные действия могут понизить уровень безопасности в компании. Групповая политика – это функция семейства операционных систем Microsoft Windows NT, которая обеспечивает управление рабочей средой учетных записей пользователей и компьютеров. Групповая политика обеспечивает централизованное управление и настройку операционных систем, приложений и пользовательских параметров в среде Active Directory. Набор конфигураций групповой политики называется объектом групповой политики (Group Policy Object, GPO). Версия групповой политики, называемая локальной групповой политикой (LGPO или LocalGPO), позволяет управлять объектами групповой политики без использования Active Directory на автономных компьютерах. ",
```

Продолжение на следующей странице:

```
    "risk_impact": "Незапланированное изменение правила политики аудита неавторизованным пользователем представляет большой риск. В худшем случае ИТ-система может оказаться уязвимой к различным атакам злоумышленников и вредоносного ПО. Воспользовавшись этой уязвимостью, злоумышленники могут получить полный контроль над ИТ-системой и возможность красть, изменять и удалять данные или устанавливать программы-вымогатели без ведома владельцев системы. \r\n\r\nh2. Ссылки \r\n\r\nhhttps://docs.microsoft.com/en-us/windows/desktop/srvnodes/group-policy \r\nhhttps://www.varonis.com/blog/group-policy/ ",
    "solution": "* Убедитесь, что действие запланировал и выполнил авторизованный пользователь, используя установленные процессы и процедуры управления изменениями. \r\n* Рассмотрите возможность отслеживания изменений и обновлений рабочих систем с помощью системы управления изменениями (например, системы отслеживания). Сопоставляйте такие события с утвержденными/авторизованными изменениями. \r\n* Измените права на защищенный доступ к журналам событий соответствующим образом. Это можно сделать локально или через групповую политику.",
    "created_at": "2023-04-10T13:08:01.29391Z",
    "updated_at": "2023-04-10T13:08:01.29391Z",
    "id": "31549173-fa4c-4a5c-a9e1-1107dbb8a48d",
    "display_id": 67,
    "mitigation": "Профилактика вредоносной деятельности эффективнее, чем исправление ее последствий. Далее приведен список основных рекомендаций по повышению безопасности ваших систем: \r\n\r\n* Если настройки аудита были изменены, верните их в исходное состояние. \r\n* Если известен идентификатор входа в систему, используйте его для сопоставления с другими связанными событиями (например, идентификатор события 4624). \r\n* Убедитесь, что только доверенные сотрудники могут изменять настройки рабочих систем и что все такие изменения авторизуются и регистрируются системой управления изменениями. \r\n* Если внесенное изменение является несанкционированным, клиент должен выяснить причину его внесения и потенциальную степень его влияния на систему. ",
    "synopsis": "Обнаружены изменения политики аудита. ",
    "local": false,
    "type": "policy_violation",
    "identifier": {
      "maxpatrol": {},
      "nessus": {},
      "redcheck": {}
    },
    "comment": "",
    "fallback_raw_risklevel": 0,
    "new_version": false,
    "client_note": "",
    "internal_note": "",
    "cpes": [],
    "category_id": "",
    "customer_created": false,
    "software_compliance": false,
    "itsm_last_synced_at": null,
    "updated_by": "",
    "created_by_customer": "",
    "edited_by": "",
    "front_link": "http://127.0.0.1/rmc/incidents/types/show/31549173-fa4c-4a5c-a9e1-1107dbb8a48d"
  },
  "total": 1
}
```

4.1.2 Получение списка инцидентов

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/incidents

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры: поддерживаются стандартные фильтры

Пример вызова:

```
curl --location
'https://127.0.0.1:9000/cruddy/public/api/v1/incidents?page=1&per_page=1&order=id%
20DESC' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{
  // массив инцидентов
  "objects": [
    {
      // идентификатор инцидента
      "id": "00000000-0000-0000-0000-00000000004b",
      // ОПИСАНИЕ УГРОЗЫ
      "description": "Модуль аналитики киберугроз обнаружил попытку
установить связь с использованием подозрительного сертификата TLS. Злоумышленники
могут использовать сертификаты с истекшим сроком действия или ненадежные
сертификаты для своих целей. Протокол защиты транспортного уровня (Transport Layer Security)
TLS обеспечивает шифрование соединения между сервером и клиентами. В соответствии
со стандартом PCI, «для обеспечения соответствия требованиям стандарта защиты
данных PCI (PCI DSS) в отношении безопасности платежных данных следует до 30 июня 2018
года прекратить использование протокола SSL и ранней версии протокола TLS и
внедрить усовершенствованный протокол шифрования TLS версии 1.1 или выше
(настоятельно рекомендуется использовать TLS v1.2)». ",
```

Продолжение ответа на следующей странице:

```

// ПОСЛЕДСТВИЯ РЕАЛИЗОВАННОЙ УГРОЗЫ
"risk_impact": "Связь с использованием вредоносных сертификатов
TLS является
подозрительной и может оказать негативное влияние на
корпоративную сеть. В случае
успеха злоумышленника наиболее опасными являются следующие
риски: \r\n \r\n* Раскрытие
информации: Уязвимость, которая может привести к раскрытию
учетных данных жертвы. В
результате киберпреступник может получить действительные учетные
данные, а с их помощью
- доступ к конфиденциальной информации. \r\n* Повышение
привилегий: Злоумышленник,
успешно воспользовавшийся этой уязвимостью, может запустить
произвольный код от имени
администратора. В этом случае он также получает возможность
устанавливать программы,
просматривать, изменять и удалять данные, создавать новые
учетные записи с полными
правами пользователя. \r\n* Удаленное выполнение кода: Эта
уязвимость позволяет
злоумышленнику получить доступ к чужому вычислительному
устройству и вносить изменения,
независимо от географического расположения этого устройства.
\r\n* Распространение
вредоносного контента или спама, а также перенаправление доменов
на страницы с
вредоносным контентом и выдача себя за владельцев учетных
записей с целью
распространения фальшивого контента или вредоносных ссылок.
\r\n* Сбор учетных данных
для продажи третьим сторонам. \r\n \r\nh2. Ссылки
\r\n\r\nhttps://www.greenhousedata.com/blog/how-secure-is-https-
tls-ssl-increasinglyused-to-transmit-malware \r\nhttp://www.certificate-
transparency.org/what-is-ct
\r\nhttps://www.csoonline.com/article/3212965/why-ssl-tls-
attacks-are-on-the-rise.html
\r\nhttps://www.pcisecuritystandards.org/document_library ",
// РЕКОМЕНДАЦИИ ПО УСТРАНЕНИЮ УГРОЗЫ
"solution": "* Немедленно заблокируйте трафик, направленный на
целевой узел.
\r\n* Проверьте узел на предмет взлома. \r\n* Проанализируйте,
является ли такое
поведение признаком действий злоумышленников. \r\n*
Просканируйте узел с помощью
антивирусной программы на предмет наличия угроз и устраните их в
случае выявления.
\r\n* Установите корневой центр сертификации на все подключаемые
к узлу клиенты. Для
этого используйте групповые политики Windows: 1) Откройте
соответствующий объект
групповой политики (GPO) 2) В дереве консоли выберите узел
«Доверенные корневые центры
сертификации»: имя объекта политики/конфигурация
компьютера/настройки Windows/настройки
безопасности/политики открытого ключа/доверенные корневые центры
сертификации 3)
Выберите пункт «Все задачи» в меню «Действие» и нажмите на
кнопку [Импорт...] для
добавления корневого сертификата.",

```

Продолжение ответа на следующей странице:

```

// РЕКОМЕНДАЦИИ ПО УМЕНЬШЕНИЮ РИСКА
    "mitigation": "* Отключите узел-источник от сети. \r\n*
Используйте только
    сертификаты, подписанные доверенным центром сертификации. \r\n*
Проверяйте брандмауэр и
    групповые политики. ",
    // Статус инцидента
    "status": "new",
    // Уровень значимости инцидента
    "risklevel": 7,
    // идентификатор актива связанного с инцидентом
    "service_asset_id": "000000000-0000-0000-0000-0000000000153",
    // дата создания инцидента
    "created_at": "2021-10-21T07:04:35.717156Z",
    // дата обновления инцидента
    "updated_at": "2021-10-21T07:04:36.969609Z",
    // идентификатор типа инцидента
    "finding_id": "5000ed94-823b-4b28-bf05-2aeade5632ac",
    // результаты анализа инцидента
    "analysis_output": "",
    // Краткое описания инцидента
    "synopsis": "Обнаружена попытка установить связь с
использованием подозрительного сертификата TLS.",
    // Наименование инцидента
    "title": "Обнаружена попытка установить связь с использованием
подозрительного сертификата TLS",
    // Уровень риска инцидента
    "risk": "medium",
    // Дата принятия инцидента
    "acknowledged_at": null,
    // Скоринговое число инцидента
    "immediate_action_score": 0.71502495,
    // Период реакции на инцидент
    "throughput_period": "normal",
    // Дата изменения периода реакции на инцидент
    "throughput_period_change": "2021-10-21T07:04:36.261123Z",
    // идентификатор правила корреляции
    "logmule_identififier": "",
    // Идентификатор последнее происшествя
    "last_occurrence_id": "000000000-0000-0000-0000-0000000000128",
    // Назначено на пользователя (идентификатор)
    "user_id": "",
    // Дата изменения инцидента
    "updated_by": "93adf94b-0f93-45d1-8b3c-a15a38399d49",
    // Назначено на группу пользователей (идентификатор)
    "group_id": "eedb34c7-9c94-4226-8504-ec1ea3ab46d",
    // Инцидент принят пользователем (идентификатор)
    "acknowledged_by": "",
    // Инцидент изменен пользователем (идентификатор)
    "edited_by": "",
    // Наименование актива связанного с инцидентом
    "service_asset_name": "127.0.0.1",
    // Наличие актива связанного с инцидентом
    "service_asset_active": true,
    // Кол-во происшествий
    "occurrence_count": 5,
    // Логин назначенного пользователя
    "user_short_name": "",

```

Продолжение ответа на следующей странице:

```

// Наименование группы назначенных пользователей
"group_name": "users",
// Человеко-читаемый идентификатор типа инцидента
"finding_display_id": 53,
// кол-во переоткрытий инцидента
"reopened_count": 0,
// тип события
"event_type": "logmule_result",
// тип инцидента
"finding_type": "network_anomaly",
// айпи адрес последнего происшествия
"last_occurrence_ip": "127.0.0.1",
// Дата последней смены статуса инцидента
"last_status_change": "2021-10-21T07:04:35.983333Z",
// Дата проведения сканирования
"last_scan": null,
// Дата последнего происшествия
"last_occurrence": "2021-10-21T07:04:33.255Z",
// Уязвимость эксплуатируется удаленно?
"remote_exploitable": false,
// Сетевая видимость актива
"service_asset_network_exposure": 3,
// Отображаемое имя инцидента
"display_title": "Обнаружена попытка установить связь с
использованием подозрительного сертификата TLS",
// Время реакции на инцидент
"customer_retention_time": 0,
// известен с даты
"visible_since": null,
// известен в кол-ве дней
"visible_since_in_days": 0,
// Пользователь последний изменивший статус инцидента
"last_customer_status_change": null,
// Объект актива связанного с инцидентом
"service_asset": {
  "id": "00000000-0000-0000-0000-000000000153",
  "type": "Host",
  "name": "127.0.0.1",
  "description": "",
  "coordinates": "--- [] ",
  "active": true,
  "scan_id": "",
  "value": 3,
  "client_note": "",
  "internal_note": "",
  "location": "",
  "network_exposure": 3,
  "responsible_person": "",
  "technical_specialist": "",
  "responsible_group_id": "eedb34c7-9c94-4226-8504-
ecc1ea3ab46d",
  "edited_by": ""
},
// Происшествия связанные с инцидентом
"occurrences": [
  {
    // идентификатор происшествия
    "id": "c17e0980-89e5-4c8e-8e89-9ee678747c7b",

```

Продолжение ответа на следующей странице:

```

// тип события
"event_type": "logmule_result", // manual_source
// ip хоста вызвавшего происшествие
"ip": "127.0.0.1",
// порт хоста вызвавшего происшествие
"port": 0,
// мак адрес хоста вызвавшего происшествие
"mac": "",
// начало происшествия
"start_occurrence": "2021-09-10T10:33:30Z",
// окончание происшествия
"end_occurrence": "2021-09-10T10:33:30Z",
// идентификатор инцидента связанного с происшествием
"service_asset_finding_id": "00000000-0000-0000-
0000-000000000004b",

// дата создания происшествия
"created_at": "2021-09-10T10:43:49.150985Z",
// дата изменения происшествия
"updated_at": "2021-09-10T10:43:49.150985Z",
// FQDN хоста вызвавшего происшествие
"fqdn": "WINSRV02.demo.local",
"logmule_result": {
  // идентификатор результата сработки правила
  "id": "48d1c174-6d3e-4859-85bd-70da6a5e3aff",
  // идентификатор правила корреляции
  "rule_id": "f1a5609b-bb13-4f8f-948b-
92147b617f78",

  // Дата создания сработки
  "created_at": "2021-09-15T09:21:59.44004Z",
  // Дата изменения сработки
  "updated_at": "2021-09-15T09:21:59.44004Z",
  // результат анализа сработки правила
  "analysis_output": "Зафиксирована неуспешная
попытка аутентификации с узла \"\" (\") с IP-адресом 192.168.200.2 под УЗ с
истекшим сроком действия пароля \"akuchelev\".",
  // событие вызвавшее сработку правила
  "event": {
    // логлайны вызвавшие сработку
    "@loglines": {
      "10s": [
        {
          "@__data_class__": [
            "logmule.logline",
            "Logline"
          ],
          "@timestamp": "2021-09-15T12:22:30.798416+00:00",
          "action": "authenticate",
          "epoch": 1631708550.798416,
          "event": {
            "auth": {
              "key": {
                "length": 0
              }
            }
          }
        }
      ]
    }
  },
  "method": {
    "description": "A user or computer
logged on to this computer from the network - 3",
    "id": "3"
  }
}

```

Продолжение ответа на следующей странице:

to log on.",

```
"protocol": {
  "name": "NTLM",
  "version": "-"
},
"category": "host_authentication",
"description": "An account failed

"logsource": {
  "application": "os",
  "host": "127.19.0.2",
  "input": "file-input",
  "name": "Microsoft Windows",
  "product": "windows",
  "subsystem": "authentication",
  "vendor": "microsoft"
},
"severity": 6.0,
"subcategory": "host_authentication_failed",
"timestamp": "2021-09-15T12:22:30.7984162Z",
"uuid": "AAAAAGFBuzOrp+C7hegeVZv+G96QLMhg",
"worker": {
  "host": "c96a88b69f66",
  "internal": false,
  "ip": "127.19.0.2"
},
"initiator": {
  "host": {
    "fqdn": [],
    "hostname": [],
    "internal": false,
    "ip": [
      "192.168.200.2"
    ]
  },
  "socket": {
    "port": 3466
  },
  "user": {
    "domain": "-",
    "id": "S-1-0-0",
    "name": "-"
  }
},
"observer": {
  "event": {
    "id": "4625",
    "type": "security"
  },
  "host": {
    "fqdn": [
      "v-demo-dc01.demo.local"
    ],
    "hostname": [],
    "internal": false,
    "ip": []
  }
},
```

Продолжение ответа на следующей странице:


```

        "outcome": {
            "description": "unknown - 0xC0000071",
            "name": "failure",
            "original": "0xC0000071"
        },
        "reportchain": {
            "collector": {
                "host": {
                    "fqdn": [],
                    "hostname": [
                        "v-stand-09"
                    ],
                    "internal": false,
                    "ip": []
                },
                "timestamp": "2021-09-13T01:39:46.393963+03:00"
            },
            "relay": {
                "host": {
                    "fqdn": [],
                    "hostname": [],
                    "internal": false,
                    "ip": [
                        "127.0.0.1"
                    ]
                }
            }
        },
        "routing_key": "#.microsoft.windows.os.authentication.#",
        "target": {
            "auth": {
                "process": {
                    "name": "NtLmSsp"
                }
            },
            "host": {
                "fqdn": [
                    "v-demo-dc01.demo.local"
                ],
                "hostname": [],
                "internal": false,
                "ip": []
            },
            "process": {
                "path": {
                    "original": "-"
                }
            },
            "user": {
                "domain": "DEMO",
                "id": "S-1-0-0",
                "name": "akuchelev"
            }
        }
    },

```

Продолжение ответа на следующей странице:

```

{
  "@__data_class__": [
    "logmule.logline",
    "Logline"
  ],
  "@timestamp": "2021-09-15T12:22:35.798416+00:00",
  "action": "authenticate",
  "epoch": 1631708555.798416,
  "event": {
    "auth": {
      "key": {
        "length": 0
      },
      "method": {
        "description": "A user or
computer logged on to this computer from the network - 3",
        "id": "3"
      },
      "protocol": {
        "name": "NTLM",
        "version": "-"
      }
    },
    "category": "host_authentication",
    "description": "An account failed to log
on.",
    "logsource": {
      "application": "os",
      "host": "127.19.0.2",
      "input": "file-input",
      "name": "Microsoft Windows",
      "product": "windows",
      "subsystem": "authentication",
      "vendor": "microsoft"
    },
    "severity": 6.0,
    "subcategory": "host_authentication_failed",
    "timestamp": "2021-09-15T12:22:35.798416Z",
    "uuid": "AAAAAGFBuzMFtX1RDhFshDyljRn8svp2",
    "worker": {
      "host": "c96a88b69f66",
      "internal": false,
      "ip": "127.19.0.2"
    }
  },
  "initiator": {
    "host": {
      "fqdn": [],
      "hostname": [],
      "internal": false,
      "ip": [
        "127.0.0.1"
      ]
    },
    "socket": {
      "port": 3466
    },
    "user": {
      "domain": "-",
      "id": "S-1-0-0",
      "name": "-"
    }
  }
}

```

Продолжение ответа на следующей странице:

```

    }
    "observer": {
      "event": {
        "id": "4625",
        "type": "security"
      },
      "host": {
        "fqdn": [
          "v-demo-dc01.demo.local"
        ],
        "hostname": [],
        "internal": false,
        "ip": []
      }
    },
    "outcome": {
      "description": "unknown - 0xC0000071",
      "name": "failure",
      "original": "0xC0000071"
    },
    "reportchain": {
      "collector": {
        "host": {
          "fqdn": [],
          "hostname": [
            "v-stand-09"
          ],
          "internal": false,
          "ip": []
        }
      },
      "timestamp": "2021-09-13T01:39:46.393963+03:00"
    },
    "relay": {
      "host": {
        "fqdn": [],
        "hostname": [],
        "internal": false,
        "ip": [
          "127.0.0.1"
        ]
      }
    }
  },
  "routing_key": "#.microsoft.windows.os.authentication.#",
  "target": {
    "auth": {
      "process": {
        "name": "NtLmSsp"
      }
    },
    "host": {
      "fqdn": [
        "v-demo-dc01.demo.local"
      ],
      "hostname": [],
      "internal": false,
      "ip": []
    }
  },

```

Продолжение ответа на следующей странице:

```

        "process": {
        "path": {
                "original": "-"
            }
        },
        "user": {
            "domain": "DEMO",
            "id": "S-1-0-0",
            "name": "akuchelev"
        }
    }
}
],
"@over": [
    "initiator.host.ip",
    "initiator.host.fqdn",
    "initiator.host.hostname",
    "target.user.name"
],
"@routing_key": "#.microsoft.windows.os.authentication.#",
// дата события
"@timestamp": "2021-09-15T12:22:30+00:00",
"@values": {
    "10s": 2
},
"@window_list": [
    "10s"
],
"end_event_time": 1631708555,
"initiator": {
    "host": {
        "fqdn": [],
        "hostname": [],
        "ip": [
            "192.168.200.2"
        ]
    }
},
"start_event_time": 1631708550,
"target": {
    "user": {
        "name": "****"
    }
}
},
"acknowledged": false,
"risklevel": 4,
"occurred_at": "2021-09-15T12:22:30Z",
"occurrence_id": "c17e0980-89e5-4cce-8e89-9ee678747c7b",
"error": "",
"service_asset_id": "00000000-0000-0000-0000-000000000153",
"asset_info": {
    "ip": "192.168.200.2",
    "hostname": "",
    "fqdn": "",
    "mac": ""
},
},

```

Продолжение ответа на следующей странице:

```

        "incident_identifier": "****"
    }
},
    // ссылка на инцидент
    "front_link": "http://127.0.0.1:8080/rmc/incidents/show/00000000-0000-0000-
0000-000000000004b"
}
],
    // общий счетчик кол-во инцидентов в ответе
    "total": 4
}

```

4.1.3 Обновление статуса инцидентов

Тип метода: PUT

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/incidents/update/status

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Описание
id	Идентификатор инцидента
status	Статус инцидента. Может принимать следующие значения: <ul style="list-style-type: none"> new: "Новый" assigned_customer: "Назначен" working_customer: "В работе" feedback_required: "Запрошена информация" verify_close: "Ожидает проверки" risk_accepted: "Риск принят" closed: "Закрыт" invalid: "Недействительный"

Пример вызова:

```

curl --location
'https://127.0.0.1:9000/cruddy/public/api/v1/incidents/update/status' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
--data '{
    "id": "00000000-0000-0000-0000-000000000004b", //string
    "status": "new" //string
}'

```

Выходные параметры: Отсутствуют

4.1.4 Обновление уровня рисков инцидентов

Тип метода: PUT

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/incidents/update/critical

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Описание
id	Идентификатор инцидента
risklevel	Уровень риска - число с запятой от 0.0 до 10.0

Пример вызова:

```
curl --location
'https://127.0.0.1:9000/cruddy/public/api/v1/incidents/update/critical' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
--data '{
  "id": "00000000-0000-0000-0000-00000000004b", //string
  "risklevel": 4.1
}'
```

Выходные параметры: Отсутствуют

4.1.5 Создание инцидента

Тип метода: POST

Метод: https://<MASTER_KARAKEN_HOST>:9009/cruddy/v1/create

Заголовки (headers):

PgrApiKey: api_key_string

PgrSelectedInstance: string //В качестве примера указано создание инцидента на указанном инстансе

Content-Type: application/json

Входные параметры:

Параметр	Описание
body	Параметры создаваемого инцидента

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/create' \
--header 'PgrApiKey: 500000000-4000-0000-9000-1000000000000' \
--header 'Content-Type: application/json'
--data '{
  "ServiceAssetFindingDBFields": {
    "Id": "string",
    "Description": "string",
    "RiskImpact": "string",
    "Solution": "string",
    "Mitigation": "string",
    "Status": "string",
    "Risklevel": float32,
    "ServiceAssetId": "string",
    "CreatedAt": "Time",
    "ReopenedAt": "Time",
    "UpdatedAt": "Time",
    "FindingId": "string",
    "AnalysisOutput": "string",
    "Synopsis": "string",
    "Title": "string",
    "Risk": "string",
    "AcknowledgedAt": "Time",
    "AlertType": "string",
    "ClientNote": "string",
    "InternalNote": "string",
    "External": boolean,
    "ImmediateActionScore": float32,
    "ThroughputPeriod": "string",
    "ThroughputPeriodChange": "Time",
    "CustomerCreated": boolean,
    "CVisibleSince": "Time",
    "CVisibleSinceInDays": int,
    "CReopenedCount": int,
    "CLastCustomerStatusChange": "Time",
    "CCustomerRetentionTime": int,
    "LogmuleIdentifier": "string",
    "CRemoteExploitable": boolean,
    "COccurrenceCount": int,
    "LastOccurrenceId": "string",
    "ItsmLastSyncedAt": "Time",
    "ItsmSyncStatus": "string",
    "ExternalId": int,
    "ItsmSyncError": "string",
    "UserId ": "string",
    "UpdatedBy": "string",
    "GroupId ": "string",
    "AcknowledgedBy": "string",
    "CreatedByCustomer": "string",
    "EditedBy": "string",
    "IncidentGroupId ": "string",
    "DisplayId": "string"
  }
}'
```

Выходные параметры и пример ответа:

```
{"id": "2dccfb43-f30a-4389-a6cf-94691b7b0a4a"}
```

4.2. АКТИВЫ

4.2.1 Получение списка активов

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/serviceAssets

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры: Поддерживаются стандартные фильтры

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/public/api/v1/serviceAssets' \  
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \  
--header 'Content-Type: application/json'
```

Выходные параметры и примеры ответа:

```
{  
  "objects": [  
    {  
      "id": "e2ad28ad-7249-4e71-b724-8617d44a39d9",  
      "type": "Host",  
      "name": "192.168.15.15",  
      "description": "",  
      "coordinates": "--- []",  
      "active": true,  
      "scan_id": "",  
      "value": 3,  
      "client_note": "",  
      "internal_note": "",  
      "location": "",  
      "network_exposure": 3,  
      "responsible_person": "",  
      "technical_specialist": "",  
      "responsible_group_id": "",  
      "edited_by": "",  
      "risk": "none",  
      "network_interfaces": [  
        {  
          "id": "2dccfb43-f30a-4389-a6cf-94691b7b0a4a",  
          "name": "192.168.15.15",  
          "ip": "192.168.15.15",  
          "mac": "",  
          "fqdn": [],  
          "os": "",  
          "service_asset_id": "e2ad28ad-7249-4e71-b724-8617d44a39d9",  
          "edited_by": ""  
        }  
      ]  
    }  
  ]  
}
```

Продолжение ответа на следующей странице:


```
    ],
    "all_open_count": 1,
    "front_link": "http://127.0.0.1/rmc/service_assets/show/e2ad28ad-7249-4e71-
b724-8617d44a39d9"
  }
],
"total": 1
}
```

4.2.2 Получение перечня групп активов

Тип метода: GET

Метод:

`https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/serviceAssetGroups`

Заголовки (headers):

`PgrApiKey: api_key_string`

`Content-Type: application/json`

Входные параметры: Поддерживаются стандартные фильтры

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/public/api/v1/serviceAssetsGroups' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
curl --location 'https://127.0.0.1:9000/cruddy/public/api/v1/serviceAssetsGroups' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json' {
  "objects": [
    {
      "id": "929941bd-9037-4059-b5be-568d1029e62b",
      "name": "Филиал 17",
      "network_ranges": [],
      "domain": "",
      "itsm_synced": false,
      "created_at": "2023-04-10T14:39:45.400048Z",
      "updated_at": "2023-04-10T14:39:45.400048Z",
      "regex": "",
      "subject_id": "",
      "object_id": "",
      "is_kii": false,
      "is_fincert": false,
      "responsible_person": ""
    }
  ]
}
```

Продолжение ответа на следующей странице:

```

"technical_specialist": "",
"system_id": "",
"responsible_group_id": "",
"edited_by": "",
"service_assets": [
  {
    "id": "e2ad28ad-7249-4e71-b724-8617d44a39d9",
    "type": "Host",
    "name": "192.168.15.15",
    "description": "",
    "coordinates": "--- [] ",
    "active": true,
    "scan_id": "",
    "value": 3,
    "client_note": "",
    "internal_note": "",
    "location": "",
    "network_exposure": 3,
    "responsible_person": "",
    "technical_specialist": "",
    "responsible_group_id": "",
    "edited_by": "",
    "network_interfaces": [
      {
        "id": "2dccfb43-f30a-4389-a6cf-94691b7b0a4a",
        "name": "192.168.15.15",
        "ip": "192.168.15.15",
        "mac": "",
        "fqdn": [],
        "os": "",
        "service_asset_id": "e2ad28ad-7249-4e71-b724-8617d44a39d9",
        "edited_by": ""
      }
    ]
  }
],
"total": 1
}

```

4.3. Пользователи

4.3.1 Получение списка пользователей

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/public/api/v1/users

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры: Поддерживаются стандартные фильтры

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/public/api/v1/users' \
--header 'PgrApiKey: 500000000-4000-0000-9000-1000000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{
  "objects": [
    {
      "id": "93adf94b-0f93-45d1-8b3c-a15a38399d49",
      "email": "",
      "first_name": "",
      "last_name": "",
      "username": "admin"
    }
  ],
  "total": 1
}
```

4.4. Хранилища

4.4.1 Получение списка хранилищ

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/list

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
page	Да	query параметр страницы просмотра
per_page	Да	query параметр кол-ва на страницу просмотра
order	Да	query параметр сортировки по полю хранилища
name	Нет	query параметр фильтрации по имени
is_large	Нет	query параметр фильтрации по полю is_large

Пример вызова:

```
curl --location
'https://127.0.0.1:9000/cruddy/v1/kv/list?page=1&per_page=10&order=name' \
--header 'PgrApiKey: 500000000-4000-0000-9000-1000000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{
  "list": [
    {
      "id": "210d3e78-410e-451b-8a5d-dbefb111a804",
      "name": "Название",
      "description": "Описание",
      "values_scheme": "[{\"name\": \"value\", \"type\": \"string\",
      \"is_key\": false}]",
      "is_large": false,
      "mask_values": false,
      "type": "pg",
      "version": 1,
      "source": "",
      "scheme": "vstore",
      "db_name": "vs_61255815f84b4eb6a79aebcc8f7e64dd",
      "user": "",
      "password": "",
      "created_at": "2023-10-03T13:29:37.571691Z",
      "updated_at": "2023-10-03T13:29:58.174256Z",
      "store_count": 1
    }
  ],
  "total": 1
}
```

4.4.2 Получение информации о хранилище

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804' \
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{
  "info": {
    "id": "210d3e78-410e-451b-8a5d-dbefb111a804",
    "name": "Название",
    "description": "Описание",
    "values_scheme": "[{\"name\": \"value\", \"type\": \"string\", \"is_key\":
false}]",
    "is_large": false,
    "mask_values": false,
    "type": "pg",
    "version": 1,
    "source": "",
    "scheme": "vstore",
    "db_name": "vs_61255815f84b4eb6a79aebcc8f7e64dd",
    "user": "",
    "password": "",
    "created_at": "2023-10-03T13:29:37.571691Z",
    "updated_at": "2023-10-03T13:29:58.174256Z",
    "store_count": 1
  }
}
```

4.4.3 Обновление информации о хранилище

Тип метода: PUT

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища
data	Да	JSON с данными о хранилище

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json' \
--data '{
  "id": "210d3e78-410e-451b-8a5d-dbefb111a804",
  "name": "Хранилище",
  "description": "Описание",
  "values_scheme": "[{\"name\": \"value\", \"type\": \"string\", \"is_key\":
false}]",
  "is_large": false,
  "mask_values": false,
  "type": "pg",
  "version": 1,
  "source": "",
  "scheme": "vstore",
  "db_name": "vs_61255815f84b4eb6a79aebcc8f7e64dd",
  "user": "",
  "password": "",
  "created_at": "2023-10-03T13:29:37.571691Z",
  "updated_at": "2023-10-03T13:29:58.174256Z",
  "store_count": 1
}'
```

Выходные параметры и пример ответа: Отсутствуют

4.4.4 Получение данных из хранилища в формате JSON

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/download

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804/download' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
# Ключи и их значения формата [{"Идентификатор": "_id", "TTL": "_ttl", "Значение": "value"}]
[
  {
    "_id": "asd",
    "_ttl": 1698526804000,
    "value": "123"
  }
]
```

4.4.5 Получение всех ключей из хранилища

Тип метода: GET

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/keys

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804/keys' \
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа:

```
{
  "keys": [
    "asd"
  ],
  "ttl": null
}
```

4.4.6 Установление значения по ключу

Тип метода: PUT

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/keys

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища
data	Да	JSON с параметрами

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804/keys' \  
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \  
--header 'Content-Type: application/json' \  
--data '{  
  {  
    "trace_id": "8abb139f-7d0c-440b-9d36-210d28d2cde6", // Идентификатор toller  
    "key": "111", // Наименование ключа  
    "values": { // Значения ключа  
      "value": "122"  
    }  
  },  
  "ttl": 1698699600000, // TTL  
  "update": true // true - обновить существующее значение ключа, false - создать  
  // новый ключ со значением; значение key должно быть уникальным  
}'
```

Выходные параметры и пример ответа: Отсутствуют

4.4.7 Удаление ключа

Тип метода: DELETE

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/keys/trace_id/{tid}?key=111

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища
tid	Да	Идентификатор toller

Параметр	Обязательность	Описание
key	Да	Наименование ключа

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804/keys/trace_id/8abb139f-7d0c-440b-9d36-210d28d2cde6?key=111' \
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа: Отсутствуют

4.4.8 Удаление хранилища

Тип метода: DELETE

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/trace_id/{tid}

Заголовки (headers):

PgrApiKey: api_key_string
Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища
tid	Да	Идентификатор toller

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-dbefb111a804/trace_id/8abb139f-7d0c-440b-9d36-210d28d2cde6' \
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа: Отсутствуют

4.4.9 Очистка хранилища

Тип метода: DELETE

Метод:

https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/{id}/truncate/trace_id/{tid}

Заголовки (headers):

PgrApiKey: api_key_string
Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
id	Да	Идентификатор хранилища
tid	Да	Идентификатор toller

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/210d3e78-410e-451b-8a5d-  
dbefb111a804/truncate/trace_id/8abb139f-7d0c-440b-9d36-210d28d2cde6' \  
--header 'PgrApiKey: 500000000-4000-0000-9000-100000000000' \  
--header 'Content-Type: application/json'
```

Выходные параметры и пример ответа: Отсутствуют

4.4.10 Создание хранилища

Тип метода: POST

Метод:

`https://<MASTER_KARAKEN_HOST>:9000/cruddy/v1/kv/create`

Заголовки (headers):

PgrApiKey: api_key_string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
data	Да	Параметры создаваемого хранилища
tid	Да	Идентификатор toller

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/cruddy/v1/kv/create' \
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \
--header 'Content-Type: application/json' \
--data '{
  "name": "Хранилище",
  "description": "Описание1",
  "is_large": false,
  "type": "pg",
  "scheme": "vstore",
  "values_scheme":
  "[{\\"name\\":\\"value\\",\\"type\\":\\"string\\",\\"is_key\\":false}]",
  "mask_values": false,
  "tollerId": "9b6f109e-afa9-4319-af98-63ba9594a394"
}'
```

Выходные параметры и пример ответа: Отсутствуют

```
{
  "store_id": "27e1fccc-728f-4b69-97be-6babeae0fb75"
}
```

4.5. Сервисы

4.5.1 Проверка состояния сервисов

Тип метода: POST

Метод:

https://<MASTER_KARAKEN_HOST>:9009/ca-<ip>/execute

Заголовки (headers):

PgrApiKey: api_key_string

TempApiKey: api_token //Параметр создается в разделе Платформы Радар Кластер - API ключи

PgrSelectedInstance: string

Content-Type: application/json

Входные параметры:

Параметр	Обязательность	Описание
data	Да	Команда проверки сервиса в формате base64

Команда проверки сервиса указывается в виде:

```
systemctl status <сервис>.service | grep "Active:"
```

где, <сервис> - наименование сервиса **Платформы Радар**

Команда проверки сервиса должна быть закодирована в формат base64. Например, команда

```
systemctl status pangeoradar-logcollector.service | grep "Active:"
```

в формате base64 будет выглядеть так:

```
c3lzdGVtY3RsIHN0YXR1cyBwYW5nZW9yYWRhci1sb2djb2xsZWN0b3Iuc2Vydm1jZSB8IGdyZXAgIkFjdG12Z  
Toi
```

Пример вызова:

```
curl --location 'https://127.0.0.1:9000/ca-127.0.0.1/execute' \  
--header 'PgrApiKey: 50000000-4000-0000-9000-100000000000' \  
--header 'TempApiToken: 50000000-f000-d000-e000-600000000000' \  
--header 'PgrSelectedInstance: a3326fc7-4663-efee-605d-4cacf76d350d' \  
--header 'Content-Type: application/json' \  
--data \  
'c3lzdGVtY3RsIHN0YXR1cyBwYW5nZW9yYWRhci1sb2dtdWx1Mi5zZXJ2aWNlIC1uIDAgLS1uby1wYWdlc  
g=='
```

Выходные параметры и пример ответа:

```
[  
  {  
    "id": "9c1194b4-3c4a-4a94-851b-359515d47871",  
    "name": "127.0.0.1",  
    "data": {  
      "result": "● pangeoradar-logmule2.service - Pangeo Radar Logmule2\  
Loaded: loaded (/etc/systemd/system/pangeoradar-logmule2.service; enabled; vendor  
preset: enabled)\n Active: active (running) since Tue 2023-10-03 16:26:41 MSK; 1  
day 1h ago\n Main PID: 14105 (pangeoradar-log)\n Tasks: 10 (limit: 4915)\n Memory: 23.8M\n CGroup: /system.slice/pangeoradar-logmule2.service\  
└─14105 /opt/pangeoradar/bin/pangeoradar-logmule2 serve --  
conf=/opt/pangeoradar/configs\  
",  
      "error": null,  
      "status_code": 200,  
      "time": 0  
    }  
  }  
]
```