Документация API VirusTotal

Release 1.0

Дроботун Евгений

Apr 24, 2020

Оглавление


VirusTotal

ГЛАВА 1

О проекте

Перевод официальной документации по API функциям сервиса VirusTotal (3 версия). Оригинал находится здесь.

Note: Последняя правка: Apr 24, 2020. На данный момент переведено около 40% документации.

Евгений Дроботун, drobotun@xakep.ru

ГЛАВА 2

Вступление

2.1 Краткий обзор

VirusTotal API 3 версии на данный момент находится в стадии бета-версии. Данная версия заменит VirusTotal API 2 версии с течением определенного времени. Версия 3 VirusTotal API основана на спецификации http://jsonapi.org/ и была разработана с учетом простоты использования и единообразия.

irusTotal API 3 версии следует принципам REST и имеет предсказуемые, ориентированные на ресурсы URL-адреса. В этой версии API для запросов и ответов (в том числе и ответов с информацией об ошибках) используется JSON.

> Warning: VirusTotal API 3 версии уже достаточно стабилен, однако некоторые несовместимые изменения по-прежнему возможны. Мы рекомендуем вам начать использовать его для экспериментов и не критически важных проектов.

2.2 Различия публичного и Premium API

Хотя многие функции, предоставляемые API VirusTotal, свободно доступны всем зарегистрированным пользователям, некоторые из них доступны только нашими премиум-клиентами. Эти функции составляют VirusTotal Premium API, и они будут соответствующим образом идентифицированы в этом описании.

Premium API - это компонент расширенных сервисов VirusTotal для профессионалов.

Публичный (открытый) API, с другой стороны, представляет собой набор функций, доступных для всех пользователей без каких-либо затрат. Единственное, что вам нужно для использования открытого API, это зарегистрироваться в сообществе VirusTotal и получить ключ API, как описано в разделе “Начало работы”.

Note: Ограничения публичного API

- Публичный API ограничен 4 запросами в минуту и 1000 запросами в день.
- Публичный API не должен использоваться в коммерческих продуктах или услугах.
- Публичный API не должен использоваться в бизнес-процессах, которые не вносят новых файлов.

Note: Основные моменты Premium API

- Premium API не имеет ограничений по скорости запросов или суточных ограничений. Ограничения регулируются SLA (соглашением об уровне услуг).
- Premium API возвращает больше данных об угрозах и предоставляет больше функциональных возможностей.
- Premium API регулируется SLA (соглашением об уровне услуг), который гарантирует готовность данных.

Premium API имеет следующие преимущества перед публичным API:

- Позволяет выбрать частоту запросов и суточную квоту, которая наилучшим образом соответствует вашим потребностям.
- Позволяет загружать образцы для дальнейшего исследования, а также данные о сетевом трафике, которые они генерируют при выполнении, и подробные отчеты о выполнении.
- Возможно получение дополнительной информации об объектах, обработанных VirusTotal, например: предупреждения потока кода VBA для документов, исходные метаданные, выходные данные ExifTool, выходные данные IDS для зарегистрированных сетевых трасс, рейтинги популярности доменов, сертификаты SSL и т. д.
- Включает метаданные, сгенерированные исключительно VirusTotal: первая дата отправки файла, список имен файлов, с которыми файл был отправлен в VirusTotal, страны отправки, распространенность и т. д.
- Предоставит вам доступ к информации о поведении файлов, созданной в результате выполнения Windows PE, DMG, Mach-O и APK файлов в виртуализированной среде песочницы.
- Предоставляет сведения о “белых списках” и доверенных источниках.
- Позволяет задавать правила для запросов образцов, например: образцы с определенной сигнатурой; образцы, которые обнаружены более чем 10 антивирусными движками; образцы, которые содержат нужный раздел PE с определенным хэшем и т.д. Эти модификаторы поиска могут быть объединены для создания сложных запросов.
- Позволяет задавать правила для запросов URL, доменов, IP-адресов, например: все домены, зарегистрированные одним и тем же злоумышленником; все домены с TTL записи DNS A менее 5 секунд и т.д.
- Предоставляет возможность кластеризации файлов и поиска похожих файлов.
- Показывает расширенные связи, которые недоступны в публичном API, например: встроенные домены; встроенные IP-адреса; контактные домены и т. д.
- Позволяет программно взаимодействовать с VT Hunting, включая получение уведомлений о правилах YARA или автоматический запуск заданий ретро-поиска.

- Имеет строгое соглашение о предоставлении услуг (SLA), которое гарантирует доступность и готовность данных.

В частности, Premium API поддерживает следующие основные варианты использования:

- Автоматизация рабочих процессов с помощью набора данных VirusTotal, включая программное расширение предупреждений.
- Интеграция VirusTotal с вашим SIEM, SOAR, EDR или AV для целей расширения получаемой информации, а не обнаружения.
- Скачивание файлов для дальнейшего изучения в автономном режиме.
- Полная характеристика любого вида угрозы, которую можно наблюдать: файлы, URL-адреса, Домены, IP-адреса, SSL-сертификаты и т. д.

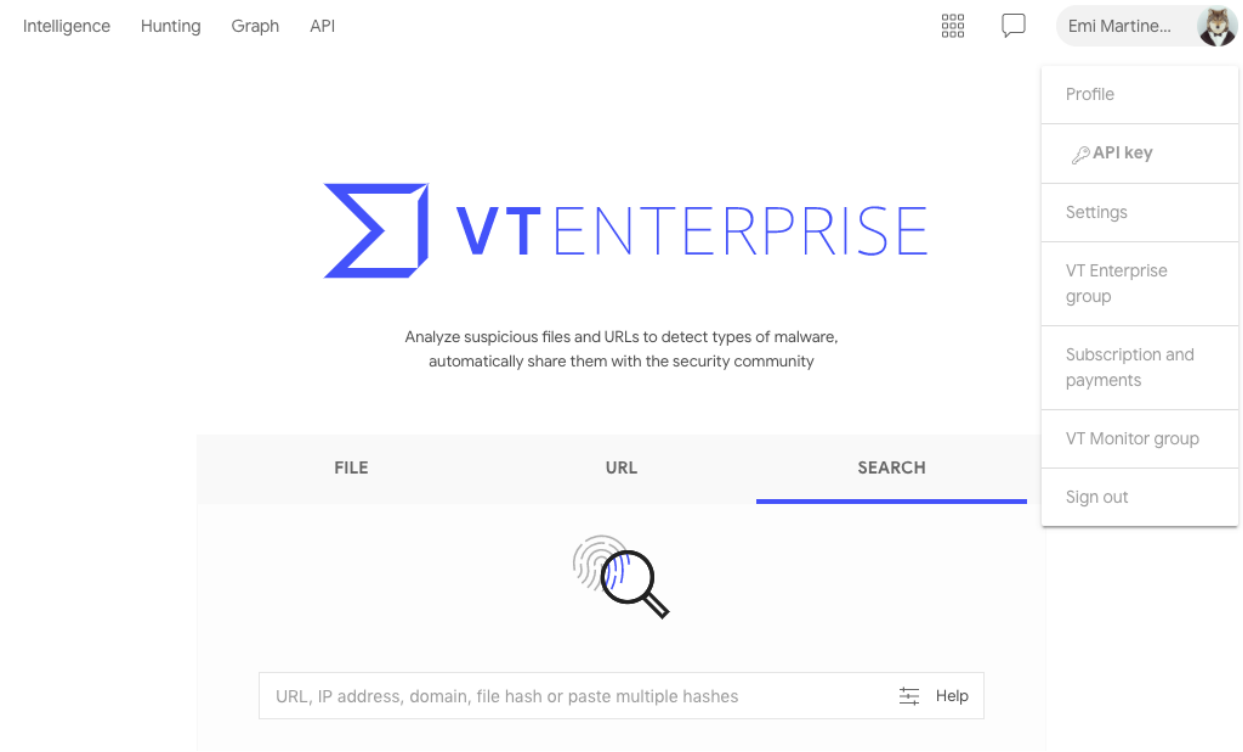
2.3 Начало работы

Для использования API необходимо зарегистрироваться в сообществе VirusTotal Community. Как только у вас будет действительная учетная запись VirusTotal Community, вы найдете свой личный ключ доступа к API в разделе личных настроек. Этот ключ - все, что вам нужно для использования VirusTotal API.

> Warning: VirusTotal API не должен использоваться в коммерческих продуктах или услугах, он не может использоваться в качестве замены антивирусных продуктов и не может быть интегрирован в любой проект, который может нанести прямой или косвенный ущерб антивирусной индустрии. Несоблюдение этих условий приведет к немедленному запрету доступа нарушителя к VirusTotal API.

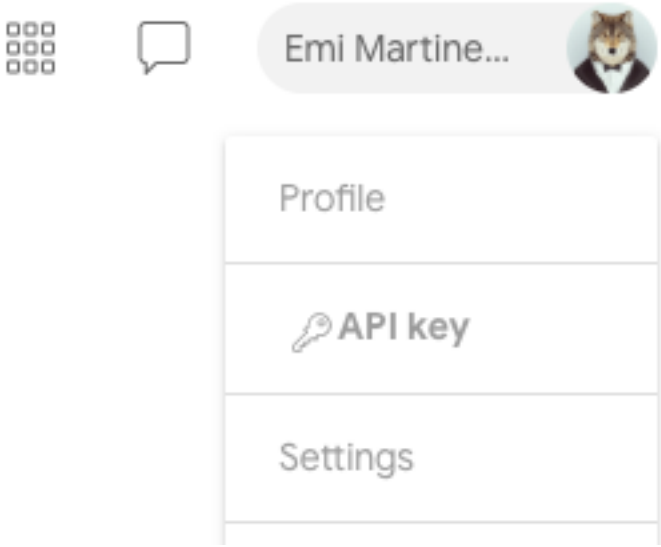
> Warning: При любых обстоятельствах Условия предоставления услуг и Политика конфиденциальности VirusTotal должны соблюдаться.

По умолчанию любой зарегистрированный пользователь VirusTotal Community имеет право на ключ API, который позволяет ему взаимодействовать с базовым набором функций API. Расширенные вызовы доступны через Premium API, который требует специальных привилегий. Свяжитесь с нами, если вы хотите узнать больше о том, как получить доступ к Premium API.



2.4 Аутентификация

Для аутентификации с помощью API вы должны включить заголовок x-apikey со своим личным ключом API во все ваши запросы. Ваш ключ API можно найти в пользовательском меню вашей учетной записи VirusTotal:



Ваш ключ доступа к API несет все ваши привилегии, поэтому держите его в безопасности и не делитесь им ни с кем. Всегда используйте HTTPS вместо HTTP для выполнения ваших запросов.

2.5 Ответы API

В большинстве случаев ресурс VirusTotal API возвращает ответ в формате JSON. Если не указано иное, ответ на успешный запрос будет иметь следующий формат:

Структура ответа

```
{
  "data": <response data>
}
```

<response data> обычно представляет собой объект или список объектов, однако это не всегда так. Примером этого является функция /files/upload_url, которая возвращает URL-адрес.

2.6 Ошибки

API VirusTotal следует обычным кодам ответа HTTP для указания успеха или неудачи. Коды в диапазоне 2xx указывают на успех. Коды в диапазоне 4xx указывают на ошибку в запросе (например, отсутствует параметр или ресурс не найден). Коды в диапазоне 5xx указывают на ошибку на серверах VirusTotal, что бывает крайне редко.

Неудачные запросы возвращают дополнительную информацию об ошибке в формате JSON:

Формат ответа в случае ошибки

```
{
  "error": {
    "code": "<error code>",
    "message": "<a message describing the error>"
  }
}
```

Код ошибки code представляет собой строку с одним из значений, приведенных в ниже.

Сообщение message содержит ,более подробную информацию об ошибке.

- 409 - Ошибка типа “AlreadyExistsError”. Ресурс уже существует.
- 401 - Ошибка типа “AuthenticationRequiredError”. Выполнение операции возможно аутентифицированным пользователем. Убедитесь, что вы предоставили свой ключ доступа к API.
- 400 - Ошибка типа “BadRequestError”. Запрос API является недопустимым или неправильным.
- 403 - Ошибка типа “ForbiddenError”. Выполнение запрошенной операции невозможно.
- 400 - Ошибка типа “InvalidArgumentError”. Некоторые аргументы, переданные в запросе неверные.
- 404 - Ошибка типа “NotFoundError” - Запрошенный ресурс не найден.
- 429 - Ошибка типа “QuotaExceededError”. Превышение одной из квот на число запросов(минутной, ежедневной или ежемесячной). Ежедневные квоты сбрасываются каждый день в 00: 00 UTC.
- 429 - Ошибка типа “TooManyRequestsError”. Большое число запросов.
- 401 - Ошибка типа “UserNotActiveError”. Учетная запись пользователя не активна.

- 401 - Ошибка типа “WrongCredentialsError”.- Ключ доступа к API является неверным.
- 503 - Ошибка типа “TransientError”. Временная ошибка сервера. Повторная попытка запроса может сработать.

2.7 Ключевые концепции

VirusTotal API (версии 3) базируется на трех ключевых понятиях: объекты (objects), коллекции (collections) и отношения (relationships).

Объект - это любой элемент, который может быть получен или обработан с помощью API. Файлы, URL-адреса, доменные имена и наборы правил поиска VirusTotal - это некоторые типы объектов, предоставляемые API.

Коллекция - это набор объектов. Объекты в коллекции обычно имеют один и тот же тип, но есть несколько исключений из этого правила. Некоторые операции API выполняются с объектами, а некоторые - с коллекциями.

Отношения - это связи между объектами, например: файл может быть связан с другим файлом, потому что один из файлов является ZIP-архивом, который содержит другой файл, URL-адрес может быть связан с файлом, потому что файл был загружен с URL-адреса, доменное имя связано со всеми URL-адресами в этом домене.

2.8 Объекты

Объект является ключевым понятием в API VirusTotal. Каждый объект имеет идентификатор и тип. Идентификаторы уникальны среди объектов одного типа. Это означает, что пара (тип, идентификатор) однозначно идентифицирует любой объект в API. В этой документации эти пары (тип, идентификатор) называются дескрипторами объектов.

Каждый объект имеет связанный с ним URL-адрес со следующей структурой:

```
https://www.virustotal.com/api/v3/{collection name}/{object id}
```

Обычно {collection name} - это множественная форма типа объекта, например, files - это коллекция, содержащая все объекты типа file, а analyses - это коллекция, содержащая все объекты analysis. Формат {object id} варьируется от одного типа объекта к другому.

GET-запрос на URL объекта возвращает информацию об этом объекте в следующем формате:

Пример ответа

```
{
  "data": {
    "type": "{object type}",
    "id": "{object id}",
    "links": {
      "self": "https://www.virustotal.com/api/v3/{collection name}/{object id}"
    },
    "attributes" : {
      "integer_attribute": 1234,
      "string_attribute": "this is a string",
      "dictionary_attribute": { "one": 1, "two": 2 },
```

(continues on next page)

(continued from previous page)

```
    "list_attribute": [ "foo", "bar", "baz" ]
   },
   "relationships" : {
     ..
   }
  }
}
```

Помимо идентификатора и типа, объект имеет набор атрибутов и отношений. Атрибуты могут быть любого типа, поддерживаемого JSON, включая списки и словари. Поле attributes всегда присутствует во всех объектах, а поле relationships является необязательными, в зависимости от того, просили ли вы включить данное поле при отправке запроса. Данный вопрос будет подробно обсуждаться в разделе "Отношения".

Каждый тип объекта имеет свой собственный заранее определенный набор атрибутов, вы не сможете добавлять или удалять атрибуты, вы можете только изменять значения существующих (в случае если они доступны для записи). Для изменения значений атрибутов объекта необходимо отправить PATCH-запрос по URL объекта. Если вы попытаетесь изменить атрибут только для чтения, вы получите сообщение об ошибке. PATCH-запрос должен содержать атрибуты, которые вы хотите изменить в структуре, подобной той, что показана в приведенном ниже примере. Любой атрибут, не включенный в запрос, останется неизменным.

Пример PATCH-запроса

```
{
  "data": {
    "type": "{object type}",
    "id": "{object id}",
    "attributes" : {
      "integer_attribute": 1234,
      "string_attribute": "this is a new string",
    }
  }
}
```

Обратите внимание, что идентификатор id и тип объекта object включены в PATCH-запрос, и они должны соответствовать указанным в URL.

Некоторые типы объектов также можно удалить, отправив DELETE-запрос на удаление по URL объекта.

2.9 Коллекции

Коллекции - это наборы объектов. Для большинства типов объектов существует коллекция верхнего уровня, представляющая все объекты этого типа. Доступ к этим коллекциям можно получить с помощью URL-адреса, например:

```
https://www.virustotal.com/api/v3/{collection name}
```

Большинство операций в API VirusTotal осуществляется путем отправки запросов к коллекции. Например, вы можете проанализировать файл, отправив POST-запрос в /api/v3/files, который успешно добавит новый элемент в коллекцию файлов. Вы можете создать новый набор правил VT Hunting,

отправив POST-запрос в ` /api/v3 /intelligence/hunting_rulesets. Отправка POST-запроса в коллекцию обычно приводит к созданию новых объектов.

Аналогично, DELETE-запрос, отправляемый в коллекцию, приводит к удалению всех объектов в этой коллекции. Конечно, для определенных коллекций, таких как files, urls или analyses нет метода DELETE для запросов, но вы можете использовать DELETE-запрос с /api/v3/intelligence/hunting_notifications, который, как вы уже поняли, удаляет все ваши уведомления VT Hunting.

Большинство коллекций являются итеративными, вы можете извлечь все объекты в коллекции, отправив в коллекцию последовательные GET-запросы. На каждый запрос вы получаете ряд объектов и курсор cursor, который можно использовать для продолжения итерации. Приведенный ниже фрагмент иллюстрирует ответ на GET-запрос на /api/v3/{collection name}.

Пример ответа коллекции

```
{
   "data": [
     { .. object 1 .. },
     { .. object 2 .. },
     { .. object 3 .. }
   ],
   "meta": {
     "cursor": "CuABChEKBGRhdGUSCQjA1.."
   },
   "links": {
       "next": "https://www.virustotal.com/api/v3/{collection name}?cursor=CuABChEKBGRhdGUSCQjA1..",
       "self": "https://www.virustotal.com/api/v3/{collection name}"
   }
}
```

Как следует из поля next в разделе links, вы можете использовать cursor в качестве параметра при последующем вызове для получения следующего набора объектов. Вы также можете использовать параметр limit для управления количеством объектов, возвращаемых при каждом вызове.

2.10 Отношения

Отношения - это способ, которым API-интерфейс VirusTotal выражает связи или зависимости между объектами. Объект может быть связан с объектами того же или другого типа. Например, файловый объект может быть связан с некоторым другим файловым объектом, который содержит первый, или файловый объект может быть связан с объектами URL, представляющими URL, встроенные в файл.

Отношения могут быть вида “один к одному” или “один ко многим”, в зависимости от того, связан объект с одним объектом или с несколькими объектами.

При извлечении какого-либо объекта с помощью GET-запроса можно также получить его связи с другими объектами. Это можно сделать, указав отношение, которое вы хотите получить в параметре relationships:

```
https://www.virustotal.com/api/v3/{collection name}/{object id}?relationships={relationship}
```

Можно указать несколько отношений в параметре relationships, перечислив их имена через запятую:

```
https://www.virustotal.com/api/v3/{collection name}/{object id}?relationships={relationship 1},{relationship 2}
```

Объекты, возвращаемые такими запросами, включают словарь отношений, где ключи - это имена запрашиваемых отношений, а значения - это либо дескриптор объекта (если отношение одно к одному), либо коллекция, как описано в разделе “Коллекции” (если отношение одно ко многим). Однако обратите внимание, что эти коллекции содержат не все связанные объекты, а только их дескрипторы (т. е. их тип и идентификатор).

Отношения в объекте

```
{
  "type": "{object type}",
  "id": "{object id}",
  "links": {
    "self": "https://www.virustotal.com/api/v3/{collection name}/{object id}"
  },
  "attributes" : {
     ..
  },
  "relationships" : {
     "{one-to-one relationship}": {
       "data": {
         "id": "www.google.com",
         "type": "domain"
       },
       "links": {
         "related": "https://www.virustotal.com/api/v3/{collection name}/{object id}/{one-to-one relationship}",
         "self": "https://www.virustotal.com/api/v3/{collection name}/{object id}/relationships/{one-to-one
↪relationship}"
       }
     },
     "{one-to-many relationship}": {
       "data": [
         { .. object descriptor 1 .. },
         { .. object descriptor 2 .. },
         { .. object descriptor 3 .. }
       ],
       "meta": {
         "cursor": "CuABChEKBGRhdGUSCQjA1LC...",
       },
       "links": {
         "next": "https://www.virustotal.com/api/v3/{collection name}/{object id}/relationships/{one-to-many
↪relationship}?cursor=CuABChEKBGRhdGUSCQjA1LC...",
         "self": "https://www.virustotal.com/api/v3/{collection name}/{object id}/relationships/{one-to-many
↪relationship}"
       },
     },
    "{relationship 2}": { ... },
    "{relationship 3}": { ... }
  }
}
```

Если вы внимательно посмотрите на поле links для связи в приведенном выше примере, вы увидите, что URL в поле self выглядит следующим образом:

```
https://www.virustotal.com/api/v3/{collection name}/{object id}/relationships/{relationship}
```

Отношения “один ко многим” - это просто коллекции, содержащие объекты, которые каким-то образом связаны с основным объектом, поэтому они обычно имеют свой собственный URL, который можно

использовать для перебора связанных объектов, отправляя GET-запросы на этот URL, как описано в разделе [“Коллекции”](#collections). При этом имеется два URL-адреса:

```
https://www.virustotal.com/api/v3/{collection name}/{object id}/relationships/{relationship}
https://www.virustotal.com/api/v3/{collection name}/{object id}/{relationship}
```

Первый URL - это коллекция, содержащая только дескрипторы (тип и идентификатор) для связанных объектов, второй - полные объекты со всеми их атрибутами. Если вас интересует только тип и идентификатор связанных объектов, вы должны использовать первый, так как более эффективно извлекать только дескрипторы, чем полные объекты.

Еще одно важное различие между обеими функциями заключается в том, что {object id}/relationships/{relationship} представляет отношение как независимую сущность и может поддерживать операции, которые изменяют отношение без изменения объектов. Напротив, {object id}/{relationship} представляет связанные объекты, а не отношение. Например, если вы хотите предоставить пользователю разрешения на просмотр графика, вы можете использовать:

```
POST https://www.virustotal.com/api/v3/graphs/{id}/relationships/viewers
```

Эта функция получает пользовательский дескриптор, она не изменяет ни пользователя, ни график, она просто создает связь между ними. С другой стороны, когда вы создаете новый комментарий для файла, вы используете:

```
POST https://www.virustotal.com/api/v3/files/{id}/comments
```

И в этом случае вы не только изменяете связь между файлом и комментарием, но и создаете новый объект комментария.

ГЛАВА 3

Объекты API

3.1 Файлы (files)

Файлы являются одним из наиболее важных типов объектов в VirusTotal API. У нас есть огромный набор данных из более чем 2 миллиардов файлов, которые были проанализированы VirusTotal на протяжении многих лет. Объект file может быть получен либо путем загрузки нового файла в VirusTotal, либо путем поиска уже существующего хэша файла, либо другими способами при поиске в службах VT Enterprise services. В объекте file вы найдете некоторые релевантные базовые атрибуты о файле и его связи с VirusTotal:

- хэш-суммы файлов, такие как md5, sha1 и sha256, которые однозначно идентифицируют файл;
- size - размер файла;
- first_submission_date - дата и время когда файл был впервые получен в VirusTotal (как временная метка UNIX);
- last_analysis_date - дата и время последнего анализа файла (как временная метка UNIX);
- last_modification_date - дата и время последнего изменения файла (как временная метка UNIX);
- times_submitted - число загрузок файла на сервер;
- last_analysis_results - результаты последнего анализа;
- names - имя файла meaningful_name, которое мы считаем наиболее содержательным;
- downloadable - показывает возможность скачивания файла с сервера;
- unique_sources - указывает, из скольких различных источников был получен файл.

JSON

```
{
  "data": {
    "type": "file",
```

(continues on next page)

(continued from previous page)

```
    "id": "<SHA256>",
    "links": {
      "self": "https://www.virustotal.com/api/v3/files/<SHA256>"
    },
    "attributes" : {
      "md5": "<string>",
      "sha1": "<string>",
      "sha256": "<string>",
      "size": <int>,
      "first_submission_date": "<unix_timestamp>",
      "last_submission_date": "<unix_timestamp>",
      "last_modification_date": "<unix_timestamp>",
      "times_submitted": <int>,
      "last_analysis_date": "<unix_timestamp>",
      "last_analysis_results": [ <objects> ],
      "names": [ <strings> ],
      "meaningful_name": "<string>",
      "downloadable": <boolean>,
      "unique_sources": <int>,
      ...
    }
  }
}
```

В словаре атрибутов присутствует также поля с информацией, извлеченной из самого файла. Эта информация раскрыта в следующих ключах:

- type_description - описание типа файла, с коротким его представлением type_tag, который можно использовать для поиска файлов этого типа;
- creation_date - извлекается, когда это возможно, из файла и указывает метку времени компиляции или сборки, может быть подделан создателями вредоносных программ;
- total_votes - общее количество голосов по результатам голосования пользователей VirusTotal Community. Поле reputation рассчитывается на основе голосов, полученных файлом, и репутации пользователей;
- vhash - значение т. н. нечеткого хэша, определяемого по алгоритму кластеризации, основанного на простом структурном хэше признаков, и который позволяет находить похожие файлы;
- tags - извлекаются из разных частей отчета и являются метками, которые помогают вам искать похожие образцы.

JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "type_description": "<string>",
      "type_tag": "<string>",
      "creation_date": "<unix_timestamp>",

      "ssdeep": "<string>",
      "vhash": "<vhash>",
```

(continues on next page)

(continued from previous page)

```
    "authentihash": "<string>",

    "reputation": <int>,
    "total_votes": { "harmless": <int>, "malicious": <int> },
    "tags": [ "<strings>" ]
   }
  }
}
```

Кроме того, VirusTotal вместе с каждым антивирусным сканированием запускает набор утилит, позволяющих собирать дополнительную информацию о файле. Вся эта информация содержится в поле attributes вместе с остальными ранее описанными полями.

3.1.1 androguard

Информация об Android файлах.

androguard показывает информацию о файлах Android APK, DEX и XML, извлеченных с помощью утилиты Androguard.

- Activities - список активностей (activities) приложения;
- AndroguardVersion - версия используемой утилиты Androguard;
- AndroidApplication - тип файла Android в формате целого числа;
- AndroidApplicationError - логическое переменная со значением False;
- AndroidApplicationInfo - тип файла Android (“APK”,” DEX”,”XML”);
- AndroidVersionCode - код версии Android, считанный из манифеста;
- AndroidVersionName - имя версии Android, считанное из манифеста;
- Libraries - список библиотек, используемых приложением;
- Main Activity - название основной активности (activitie), прочитанное из манифеста;
- MinSdkVersion - минимальная поддерживаемая версия SDK;
- Package - имя пакета, считанное из манифеста;
- Permissions - словарь с разрешениями, используемыми в качестве ключа и списка с 3 элементами в качестве значения:
  - тип разрешения (например, normal, dangerous);
  - короткий дескриптор разрешения;
  - дескриптор разрешения;
- Providers - список провайдеров (providers), используемых приложением;
- Receivers - список получателей (receivers), используемых приложением;
- RiskIndicator - словарь с показателями риска APK (structure) и PERM (permissions):
  - APK - показывает используемые компоненты и их количество (например, "EXECUTABLE": 3);
  - PERM - показывает типы разрешений и их количество (например, "DANGEROUS": 11);
- Services - список служб (services), используемых приложением;

- StringsInformation - список примечательных строк, найденных в приложении;
- TargetSdkVersion - версия Android, на которой приложение было протестировано;
- VTAndroidInfo - версия Androguard, используемая сервисом VirusTotal;
- certificate - сведения о сертификате приложения в виде словаря с полями:
  - Issuer - словарь с отличительными (уникальными) именами и значениями. Типичными записями являются DN (отличительное (уникальное) имя), CN (общее имя), O (организация);
  - Subject - словарь с RDN (перечнем уникальных имен) эмитента сертификата;
  - serialnumber - серийный номер сертификата;
  - thumbprint - хэш сертификата в шестнадцатеричном виде;
  - validfrom - дата начала действия сертификата в формате "%H:%M %p %m/%d/%Y";
  - validto - срок действия сертификата, в формате "%H:%M %p %m/%d/%Y";
- intent-filters - фильтр предполагаемых действий приложения исходя из активностей (activities), получателей (receivers) и служб (services).

Информация об APK файлах в виде JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "androguard": {
        "Activities": ["<strings>"],
        "AndroguardVersion": "<string>",
        "AndroidApplication": <int>,
        "AndroidApplicationError": <boolean>,
        "AndroidApplicationInfo": "<string>",
        "AndroidVersionCode": "<string>",
        "AndroidVersionName": "<string>",
        "Libraries": ["<strings>"],
        "Main Activity": "<string>",
        "MinSdkVersion": "<string>",
        "Package": "<string>",
        "Permissions": {"<string>": ["<strings>"], ... },
        "Providers": ["<strings>"],
        "Receivers": ["<strings>"],
        "RiskIndicator": {"APK": {"<string>": <int>, ... },
                          "PERM": {"<string>": <int>, ... }},
        "Services": ["<strings>"],
        "StringsInformation": ["<strings>"],
        "TargetSdkVersion": "<string>",
        "VTAndroidInfo": <float>,
        "certificate": {"Issuer": {"DN": "<string>", "O": "<string>", ... },
                        "Subject": {"DN": "<string>","O": "<string>", ... },
                        "serialnumber": "<string>",
                        "thumbprint": "<string>",
                        "validfrom": "<string:%H:%M %p %m/%d/%Y>",
                        "validto": "<string:%H:%M %p %m/%d/%Y>"},
        "intent-filters": {"Activities": {"<string>": {"action": ["<strings>"],
                                                       "category": ["<string>"]},
```

(continues on next page)

(continued from previous page)

```
                                ... },
                        "Receivers": {"<string>": {"action": ["<strings>"],
                                                   "category": ["<string>"]},
                                      ... },
                        "Services": {"<string>": {"action": ["<strings>"],
                                                  "category": ["<string>"]},
                                     ...}
      }
    }
  }
}
```

3.1.2 asf_info

Информация о Microsoft Advanced Streaming/Systems Format (ASF) файлах.

asf_info показывает информацию о Microsoft ASF files (.asf, .wma, .wmv).

- content_encryption_object - информация о том, как зашифрован файл:
  - key_id - ID тиспользуемого ключа;
  - license_url - url-адрес лицензии;
  - protection_type - тип используемой защиты (например, “DRM”);
  - secret_data - байты, содержащие секретные данные;
- extended_content_encryption_object - расширенная информация о том, как зашифрован файл:
  - CHECKSUM - контрольная сумма данных;
  - KID - ID тиспользуемого ключа;
  - EncodeType - тип кодирования;
  - LAINFO - информация о лицензионном соглашении;
  - DRMHeader - заголовок, используемый в DRM;
- script_command_objects - скрипты, используемые в файле:
  - action - действие, которое необходимо выполнить;
  - type - тип действия (например, URL, FILENAME, EVENT);
  - trigger_time - время активации скрипта.

Информация об ASF файлах в виде JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "asf_info": {
        "content_encryption_object": {"key_id": "<string>",
                                      "license_url": "<string>",
                                      "protection_type": "<string>",
```

(continues on next page)

(continued from previous page)

```
                                "secret_data": "<string>"},
      "extended_content_encryption_object": {"CHECKSUM": "<string>",
                                             "DRMHeader": "<string>",
                                             "EncodeType": "<string>",
                                             "KID": "<string>",
                                             "LAINFO":"<string>"},
      "script_command_objects": [{"action": "<string>",
                                  "trigger_time": <int>,
                                  "type":"URL"}, ... ]}
   }
  }
}
```

3.1.3 authentihash

Хэш для проверки PE-файлов.

authentihash - это хэш sha256, используемый корпорацией Microsoft для проверки того, что соответствующие разделы образа PE-файла не были изменены.

JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "authentihash": "<string>",
    }
  }
}
```

3.1.4 bundle_info

Информация о сжатых файлах.

bundle_info предоставляет информацию о сжатых файлах (ZIP, TAR, GZIP и т. д.).

- beginning - распакованный заголовок файла для некоторых форматов файлов (GZIP, ZLIB);
- extensions - расширения файлов и их количество внутри пакета;
- file_types - типы файлов и их количество внутри пакета;
- highest_datetime - самая последняя дата в содержащихся файлах, в формате "%H:%M %p %m/%d/%Y";
- lowest_datetime - самая старая дата в содержащихся файлах, в формате "%H:%M %p %m/%d/%Y";
- num_children - сколько файлов и каталогов находится внутри пакета;
- tags - интересные замечания о содержании (например, "contains-pe");
- type - тип пакета (например, "ZIP");

- uncompressed_size - несжатый размер содержимого внутри сжатого файла;
- vhash - хэш подобия (нечеткий хэш) для этого типа файлов.

Информация о сжатых файлах в виде JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "bundle_info": {
        "beginning": "<string>",
        "extensions": {"<string>": <int>, ... },
        "file_types": {"<string>": <int>, ... },
        "highest_datetime": "<string:%Y-%m-%d %H:%M:%S>",
        "lowest_datetime": "<string:%Y-%m-%d %H:%M:%S>",
        "num_children": <int>,
        "tags": ["<strings>"],
        "type": "<string>",
        "uncompressed_size": <int>,
        "vhash": "<string>"
      }
    }
  }
}
```

3.1.5 class_info

Информация о классах Java в байткод-файлах.

class_info предоставляет информацию о Java байткод-файлах.

- constants - константы, используемые в классе;
- extends - класс, от которого наследован данный класс;
- implements - интерфейсы реализованные в классе;
- methods - методы, принадлежащие к классу;
- name - имя класса;
- platform - платформа в виде строки, полученной из старшего и младшего номера версии;
- provides - представленные классы, поля и методы;
- requires - обязательные классы, поля и методы.

Информация о Java классе в виде JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "class_info": {
```

(continues on next page)

(continued from previous page)

```
      "constants": ["<strings>"],
      "extends": "<string>",
      "implements": ["<strings>"],
      "methods": ["<strings>"],
      "name": "<string>",
      "platform": "<string>",
      "provides": ["<strings>"],
      "requires": ["<strings>"]
    }
   }
  }
}
```

3.1.6 deb_info

Информация о Debian пакетах.

deb_info - предоставляет информацию о Debian пакетах.

- changelog - информация об изменениях в версии пакета:
  - Author - имя автора;
  - Date дата в формате “%a, %d %b %Y %H:%M%S %z”;
  - Debian revision - ревизия;
  - Debian version - версия;
  - Distributions - тип распространения;
  - Full version - полная версия системы;
  - Package - тип пакета;
  - Urgency - уровень срочности изменений;
  - Version history - история версий;
- control_metadata - общие (неизменные) поля пакета:
  - Maintainer - идентификатор того, кто осуществляет поддержку пакета;
  - Description - дескриптор пакета;
  - Package - имя пакета;
  - Depends - зависимости пакета;
  - Version - версия пакета;
  - Architecture - архитектура для запуска этого пакета (например, "i386");
- control_scripts - сценарии для запуска в операциях управления пакетами:
  - postinst - скрипт, выполняемый после инсталляции;
  - postrm - скрипт, выполняемый после удаления пакета;
- structural_metadata:
  - contained_files - количество файлов в пакете;
  - contained_items - количество пунктов в пакете;

- max_date - дата самого старого файла в формате “%Y-%m-%d %H:%M%S”;
- min_date - самая последняя дата файла в формате “%Y-%m-%d %H:%M%S”.

Информация о Debian пакете в виде JSON

```
{
 "data": {
            ...
  "attributes" : {
   ...
    "deb_info": {
     "changelog": {"Author": "<string>",
                "Date": "<string:%a, %d %b %Y %H:%M%S %z>",
                "Debian revision": "<string>",
                "Debian version": "<string>",
                "Distributions": "<string>",
                "Full version": "<string>",
                "Package": "<string>",
                "Urgency": "<string>",
                "Version history": "<string>"},
     "control_metadata": {"<string>": "<string>", ... },
     "control_scripts": {"postinst": "<string>",
                      "postrm": "<string>"},
     "structural_metadata": {"contained_files": <int>,
                          "contained_items": <int>,
                          "max_date": "<string:%Y-%m-%d %H:%M%S>",
                          "min_date": "<string:%Y-%m-%d %H:%M%S>"}
    }
  }
 }
}
```

3.1.7 dmg_info

Информация о монтируемых образах дисков macOS.

dmg_info сообщает данные о структуре файлов Apple.dmg). Большая часть данных поступает из метаданных внутренних файлов, которые могут содержаться в некоторых файлах, а в других - нет.

- blkx - список блоков в образе. Каждая запись содержит:
  - attributes - в формате шестнадцатеричного числа;
  - name - имя блока;
- data_fork_length - размер данных форка;
- data_fork_offset - смещение данных форка;
- dmg_version - версия DMG-файла;
- hfs - информация об HFS-элементах. В зависимости от конкретного случая могут присутствовать различные поля:
  - info_plist - содержимое списка свойств (plist) данного блока;
  - main_executable - основной исполняемый файл этого блока:
    - id - идентификатор;

        * path - путь в пакете;
        * sha256 - хэш содержимого;
        * size - размер файла в байтах;
      - num_files - количество файлов;
      - unreadable_files - количество нечитаемых файлов;
- plist - содержит сведения о конфигурации приложения, такие как идентификатор пакета, номер версии и отображаемое имя;
- plist_keys - ключи от записи plist;
- running_data_fork_offset - смещение начала используемых данных форка (обычно 0);
- resourcefork_keys - ключи, найденные в ресурсах форка;
- rsrc_fork_length - длина ресурсов форка;
- rsrc_fork_offset - смещение ресурсов форка;
- xml_lenght - размер списка свойств в DMG;
- xml_offset - смещение списка свойств в DMG.

Apple .dmg-файл

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "dmg_info": {
        "blkx": [{"attributes": "<string>", "name": "<string>"}, ... ],
        "data_fork_length": <int>,
        "data_fork_offset": <int>,
        "dmg_version": <int>,
        "hfs": {"info_plist": {"<string>": <value>, ... },
              "main_executable": {"id": <int>,
                                  "path": "<string>",
                                  "sha256": "<string>",
                                  "size": <int>},
              "<string>": <value>,
              ... },
        "plst": [{"attributes": "<string>", "name": "<string>"}],
        "plst_keys": ["<strings>"],
        "running_data_fork_offset": <int>,
        "resourcefork_keys": ["<strings>"],
        "rsrc_fork_length": <int>,
        "rsrc_fork_offset": <int>,
        "xml_length": <int>,
        "xml_offset": <int>
      }
    }
  }
}
```

3.1.8 dot_net_guids

Идентификаторы для сборок Microsoft .NET.

- dot_net_guids - список идентификаторов для сборок Microsoft .NET;
- mvid - ModuleVersionID, генерируемый во время сборки, в результате чего для каждой сборки создается новый идентификатор GUID;
- typelib_id - TypeLibID (если имеется), созданный Visual Studio при создании нового проекта по умолчанию.

ID сборки Microsoft .NET в виде JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "dot_net_guids": {
        "mvid": "<string>",
        "typelib_id": "<string>"
      }
    }
  }
}
```

3.1.9 elf_info

Информация о Unix ELF-файлах.

elf_info возвращает информацию о Unix ELF file format.

- exports - список экспортируемых элементов. Каждый элемент содержит имя и тип.
- header - некоторые описательные метаданные о файле:
  - type - тип файла (например "EXEC" (исполняемый файл);
  - hdr_version - версия заголовка;
  - num_prog_headers - количество записей в заголовке программы;
  - os_abi - тип бинарного интерфейса приложения (например "UNIX-Linux");
  - obj_version - 0x1 для оригинальных ELF-файлов;
  - machine - платформа (например "Advanced Micro Devices X86-64");
  - entrypoint - точка входа;
  - num_section_headers - число секций в заголовке;
  - abi_version - версия бинарного интерфейса приложения;
  - data - выравнивание данных в памяти (например "little endian");
  - class - класс файла (например "ELF32");
- imports - список импортируемых элементов. Каждый элемент содержит имя и тип;
- sections - секции ELF-файла:

  - name - имя секции;
  - address - виртуальный адрес секции;
  - flags - атрибуты секции;
  - offset - смещение секции;
  - type - тип секции;
  - size - размер секции в байтах;
- segments - они же заголовки программ. каждый элемент содержит тип сегмента и список ресурсов, задействованных в этом сегменте;
- shared_libraries - список общих библиотек, используемых этим исполняемым файлом.

Формат ELF-файла

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "elf_info": {
        "exports": [["<string>", "<string>"], ... ],
        "header": {"type": "<string>",
                   "hdr_version": "<string>",
                   "num_prog_headers": <int>,
                   "os_abi": "<string>",
                   "obj_version": "<string>",
                   "machine": "<string>",
                   "entrypoint": <int>,
                   "num_section_headers" <int>,
                   "abi_version": 0,
                   "data": "<string>",
                   "class": "<string>"},
        "imports": [["<string>", "<string>"], ... ],
        "sections": [{"name": "<string>",
                      "address": <int>,
                      "flags": "<string>",
                      "offset": <int>,
                      "type": "<string>",
                      "size": <int>}, ... ],
        "segments": [["<string>", ["<strings>"]], ... ],
        "shared_libraries": ["<strings>"]
      }
    }
  }
}
```

3.1.10 exiftool

Информация о метаданных EXIF из файлов.

exiftool это утилита для извлечения метаданных EXIF из файлов различных форматов. Представляемые метаданные могут различаться в зависимости от типа файла, и, учитывая природу метаданных EXIF, состав отображаемых полей может различаться.

Например:

- поля для Microsoft Windows PE-файлов:

CharacterSet, CodeSize, CompanyName, EntryPoint, FileDescription, FileFlagsMask, FileOS, FileSize, FileSubtype, FileType, FileTypeExtension, FileVersion, FileVersionNumber, ImageVersion, InitializedDataSize, InternalName, LanguageCode, LegalCopyright, LinkerVersion, MIMEType, MachineType, OSVersion, ObjectFileType, OriginalFileName,, PEType, ProductName, ProductVersion, ProductVersionNumber, Subsystem, SubsystemVersion, TimeStamp, UninitializedDataSize

- поля для JPEG-файлов:

Aperture, ApertureValue, BitsPerSample, BrightnessValue, CircleOfConfusion, ColorComponents, ColorSpace, Compression, CreateDate, DateTimeOriginal, DeviceType, EncodingProcess, ExifByteOrder, ExifImageHeight, ExifImageWidth, ExifVersion, ExposureCompensation, ExposureMode, ExposureProgram, ExposureTime, FNumber, FOV, FaceDetect, FileType, FileTypeExtension, Flash, FlashpixVersion, FocalLength, FocalLength35efl, FocalLengthIn35mmFormat, HyperfocalDistance, ISO, ImageHeight, ImageSize, ImageUniqueID, ImageWidth, InteropIndex, InteropVersion, LightValue, MIMEType, Make, MakerNoteVersion, MaxApertureValue, Megapixels, MeteringMode, Model, ModifyDate, Orientation, RawDataByteOrder, RawDataCFAPattern, ResolutionUnit, ScaleFactor35efl, SceneCaptureType, ShutterSpeed, ShutterSpeedValue, Software, SubSecCreateDate, SubSecDateTimeOriginal, SubSecModifyDate, SubSecTime, SubSecTimeDigitized, SubSecTimeOriginal, ThumbnailImage, ThumbnailLength, ThumbnailOffset, TimeStamp, WhiteBalance, XResolution, YCbCrPositioning, YCbCrSubSampling, YResolution

- поля для PDF_файла:

CreateDate, Creator, CreatorTool, DocumentID, FileType, FileTypeExtension, Linearized, MIMEType, ModifyDate, PDFVersion, PageCount, Producer, XMPToolkit

JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "exiftool": {
        "<string>": "<string>", ...
      }
    }
  }
}
```

3.1.11 image_code_injections

Инъекция кода в файл изображения.

image_code_injections возвращает содержимое внедренного кода в файлах изображений.

JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "image_code_injections": "<string>"
    }
  }
}
```

3.1.12 ipa_info

Информация об iOS App Store Package файле.

ipa_info - возвращает информацию о Apple IPA файлах.

- apps - каждый IPA может содержать несколько экземпляров приложения:
  - commands - список команд загрузки. Каждая запись отображается как значение ключа type;
  - vhash - vhash файла;
  - segments - список сегментов в файле:
    - name - имя сегмента;
    - fileoff - физический адрес сегмента;
    - vmsize - размер виртуального адреса;
    - vmaddr - виртуальный адрес;
    - filesize - размер сегмента;
    - sections - секции в сегменте:
      - type - тип секции;
      - flags - флаги секции (например "S_8BYTE_LITERALS");
      - name - имя секции;
    - tags общие замечания о файле (например "64 bits");
  - headers - некоторые описательные метаданные о файле:
    - cpu_type - общий тип процессора (например "i386");
    - cpu_subtype - подтип процессора (например "I386_ALL");
    - magic - “магический” идентификатор приложения;
    - size_cmds - размер команд;
    - num_cmds - количество команд;
    - flags - флаги файла (например "DYLDLINK", "NOUNDEFS");
    - file_type - тип файла (например "dynamically bound shared library");
  - libs - библиотеки, используемые в файле;

- plist - список, содержащий пары ключ-значение, которые идентифицируют и настраивают приложение. Некоторыми общими полями являются:
  - CBundleIdentifier уникальный идентификатор пакета;
  - CFBundleSupportedPlatforms - поддерживаемые платформы;
  - CFAppleHelpAnchor - имя HTML help-файла для пакета;
  - CFBundleIcons - информация об используемой иконке;
  - CFBundleShortVersionString - номер релиза или версии пакета;
  - CFBundleDisplayName - вдимое для пользователя имя пакета;
  - CFBundleName - вдимое для пользователя короткое имя пакета;
  - MinimumOSVersion - минимальная версия операционной системы, необходимая для запуска приложения;
- provision - приложения iOS должны содержать встроенный профиль инициализации:
  - TeamName - team name.
  - TeamIdentifier - team identifier.
  - Name - имя приложения;
  - AppIDName - имя идентификатора приложения;
  - ApplicationIdentifierPrefix - идентификатор подписи кода для запущенного приложения;
  - Platform - поддерживаемая платформа;
  - Version - версия приложения;
  - TimeToLive - время существования;
  - ExpirationDate - срок действия приложения в формате “%Y-%m-%d %H:%M%S”.
  - Entitlements - позволяет использовать определенную функцию или превращает приложение в отдельную службу;
    - application-identifier - полный идентификатор приложения;
  - UUID - уникальный идентификатор;
  - CreationDate - дата создания приложения в формате “%Y-%m-%d %H:%M%S”.

Файлы Apple IPA

```
{
  "data": {
              ...
    "attributes" : {
      ...
      "ipa_info": {
        "apps": [{"commands": [{"type": "<string>"}], ... ],
                 "vhash": "<string>",
                 "segments": [{"name": "<string>",
                               "fileoff": "<string>",
                               "vmsize": "<string>",
                               "filesize": "<string>",
                               "vmaddr": "<string>",
```

(continues on next page)

(continued from previous page)

```
                        "sections": [{"type": "<string>"
                                      "flags": ["<strings>"],
                                      "name": "<string>"}, ... ], } ...],
            "tags": ["<strings>"],
            "headers": {"cpu_subtype": "<string>",
                        "magic": "<string>",
                        "size_cmds": <int>,
                        "file_type": "<string>",
                        "num_cmds": <int>,
                        "flags": ["<strings>"]
                        "cpu_type": "<string>"},
            "libs":["<strings>"]} ... ],
      "plist": {"CBundleIdentifier": "<string>",
            "CFBundleSupportedPlatforms": "<string>",
            "CFAppleHelpAnchor": "<string>",
            "CFBundleIcons": "<string>",
            "CFBundleShortVersionString": "<string>",
            "CFBundleDisplayName": "<string>",
            "CFBundleName": "<string>",
            "MinimumOSVersion": "<string>", ... },
      "provision": {"TeamName": "<string>",
                "Name": "<string>",
                "TeamIdentifier": ["<strings>"],
                "AppIDName": "<string>",
                "ApplicationIdentifierPrefix": ["<strings>"],
                "Platform": ["<strings>"],
                "Version": <int>,
                "TimeToLive": <int>,
                "ExpirationDate": "<string:%Y-%m-%d %H:%M%S>",
                "Entitlements": {"application-identifier": "<string>", ... },
                "CreationDate": "<string:%Y-%m-%d %H:%M%S>",
                "UUID": "<string>", ... }
    }
   }
  }
}
```

3.1.13 isoimage_info

Информация о файлах ISO-образов.

isoimage_info - возвращает информацию о структуре ISO-файлов.

- application_id - приложение, использованное для создания файла;
- created - время создания файла в формате “%Y-%m-%d %H:%M:%S”;
- effective - фактическая дата тома в формате “%Y-%m-%d %H:%M:%S”;
- expires - дата истечения срока действия тома в формате “%Y-%m-%d %H:%M:%S”;
- file_structure_version - версия файловой структуры;
- max_date - самая “свежая” дата, содержащаяся в файле в формате “%Y-%m-%d %H:%M:%S”;
- min_date - самая старая содержащаяся дата файла в формате “%Y-%m-%d %H:%M:%S”;
- modified - дата последней модификации в формате “%Y-%m-%d %H:%M:%S”;

- num_files - количество файлов содержащихся ISO-образе;
- system_id - имя системы, которая может работать с начальными секторами (например "Win32");
- total_size - размер всех разделов в этом логическом томе;
- type_code - код типа формата (например "CD001");
- volume_id - идентификатор тома;
- volume_set_id - идентификатор объединенного тома.

Файл ISO-образа

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "isoimage_info": {
        "application_id": "<string>",
        "created": "<string:%Y-%m-%d %H:%M:%S>",
        "effective": "<string:%Y-%m-%d %H:%M:%S>",
        "expires": "<string:%Y-%m-%d %H:%M:%S>",
        "file_structure_version": <int>,
        "max_date": "<string:%Y-%m-%d %H:%M:%S>",
        "min_date": "<string:%Y-%m-%d %H:%M:%S>",
        "modified": "<string:%Y-%m-%d %H:%M:%S>",
        "num_files": <int>,
        "system_id": "<string>",
        "total_size": <int>,
        "type_code": "<string>",
        "volume_id": "<string>",
        "volume_set_id": "<string>"
      }
    }
  }
}
```

3.1.14 jar_info

Информация о файлах Java Archive.

jar_info возвращает информацию о Java jar-файлах.

- filenames - имена содержащихся файлов;
- files_by_type - типы и количество типов файлов, содержащихся в jar-файле;
- manifest - содержимое манифеста Jar;
- max_date - самая старая содержащаяся дата файла в формате"%Y-%m-%d %H:%M:%S";
- max_depth - максимальная глубина каталога jar-файла;
- min_date - самая "свежая" дата, содержащаяся в файле в формате "%Y-%m-%d %H:%M:%S";
- packages - предполагаемые пакеты, используемые в пакете .class-файлов;
- strings - примечательные строки, найденные в пакете .class-файлов;

- total_dirs - количество каталогов в пакете;
- total_files - количество файлов в пакете.

Java .jar-файлы

```
{
  "data": {
              ...
    "attributes" : {
      ...
      "jar_info": {
        "filenames": ["<strings>"],
        "files_by_type": {"<string>": <int>, ... },
        "manifest": "<string>",
        "max_date": "<string:%Y-%m-%d %H:%M:%S>",
        "max_depth": <int>,
        "min_date": "<string:%Y-%m-%d %H:%M:%S>",
        "packages": ["<strings>"],
        "strings": ["<strings>"],
        "total_dirs": <int>,
        "total_files": <int>
      }
    }
  }
}
```

3.1.15 macho_info

Информация о файлах Apple MachO.

macho_info возвращает информацию о файлах формата Apple MachO. Это список, содержащий элементы для каждого приложения:

- libs - библиотек, используемые в файле;
- headers - некоторые описательные метаданные о файле:
  - cpu_type - основной тип процессора (например i386);
  - cpu_subtype - подтип процессора (например I386_ALL);
  - magic - “магический” идентификатор приложения;
  - size_cmds - размер команд;
  - num_cmds - число команд;
  - flags флаги файлов (например DYLDLINK, NOUNDEFS);
  - file_type - тип файла (например dynamically bound shared library);
- commands - список команд загрузки. Каждая запись отображается как значение ключа type;
- segments - список сегментов файла:
  - name - имя сегмента;
  - fileoff - физический адрес сегмента;
  - vm size - размер виртуального адреса;

- vmaddr - виртуальный адрес;
- filesize - размер сегмента;
- sections - секции сегмента:
  - type - тип секции;
  - flags - флаги секции (например S_8BYTE_LITERALS);
  - name - имя секции;
- vhash - vhash файла;
- tags - общие замечания о файле (например 64 bits).

Формат файла Apple MachO

```
{
  "data": {
              ...
    "attributes" : {
      ...
      "macho_info": [
       {"libs": ["<strings>"],
        "headers": {"cpu_subtype": "<string>",
                    "magic": "<string>",
                    "size_cmds": <int>,
                    "file_type": "<string>",
                    "num_cmds": <int>,
                    "flags": ["<strings>"],
                    "cpu_type": "<string>"},
        "commands": [{"type": "<string>"}, ... ],
        "segments": [{"name": "<string>",
                      "fileoff": "<string>",
                      "vmsize": "<string>",
                      "filesize": "<string>",
                      "vmaddr": "<string>"}, ... ],
        "sections": [{"type": "<string>",
                      "flags": ["<strings>"],
                      "name": "<string>"}, ... ],
        "vhash": "<string>",
        "tags": ["<strings>"]} ...
    ]
    }
  }
}
```

3.1.16 magic

Идентификация файлов по “магическому числу”.

magic дает предположение о типе файла, основываясь на популярном инструменте синтаксического анализа из UNIX (команда file).

Предполагаемый тип файла

```
{
  "data": {
              ...
    "attributes" : {
      ...
      "magic": "<string>",
    }
  }
}
```

3.1.17 office_info

Информация о структуре файлов Microsoft Office.

office_info возвращает информацию о файлах Microsoft Office (до Office 2007). Включая информацию (Word) .doc, .dot, .wbk, (Excel) .xls, .xlt, .xlm, (PowerPoint) .pot, .pps.

- document_summary_info - некоторые метаданные о файле Office:
  - scale - True если требуется масштабирование миниатюры, False - в обратном случае;
  - links_dirty - мешают ли пользовательским ссылкам
  - line_count - количество строк;
  - hyperlinks_changed - одна или несколько гиперссылок в этой части были обновлены производителем исключительно в этой части;
  - characters_with_spaces - количество символов, включая пробелы;
  - version - целочисленный идентификатор приложения Microsoft Office;
  - shared_document - если документ является общедоступным;
  - paragraph_count - количество абзацев;
  - company - имя компании;
  - code_page - набор символов, используемый в документе;
- entries - список OLE-объектов в документе:
  - clsid - уникальный идентификатор приложения;
  - clsid_literal - читаемая версия clsid;
  - name - имя объекта;
  - sid - индекс записи в каталоге OLE;
  - size - размер объекта в байтах;
  - type_literal - тип объекта;
- ole - макросы, найденные в каталоге OLE:
  - macros - подробная информация о найденных макросах:
    - vba_code - код макроса;
    - stream_path - путь в дереве хранения OLE;
    - vba_filename - имя макроса;

        * patterns - примечательные паттерны в макросе (“exe-pattern”, “url-pattern”, и т. д.);
        * lengh - длина макроса;
        * properties - примечательные свойсвта макроса (“obfuscated”, “run-file”, и т. д.);
      - num_macros - количестово найденных макросов;
  - summary_info - оставшийся набор метаданных о файле Office. В зависимости от типа файла Office, некоторые поля могут отображаться, некоторые - нет:
      - last_author - пользователь, который последний редактировал этот файл;
      - creation_datetime - дата создания файла в формате “%Y-%m-%d %H:%M:%S”;
      - template - шаблон, используемый при создании файла;
      - author - исходный пользователь, создавший файл;
      - page_count - количество страниц в документе;
      - last_saved - дата последнего сохранения файла в формате “%Y-%m-%d %H:%M:%S”;
      - edit_time - время, затраченное на редактирование документа, в секундах;
      - word_count - количество слов в документе;
      - revision_number - номер редакции документа;
      - last_printed - дата последней печати документа в формате “%Y-%m-%d %H:%M:%S”;
      - application_name - имя приложения Office (например "Microsoft PowerPoint");
      - title - заголовок документа;
      - character_count - количество символов в документе;
      - security - 0 если пароль для документа не установлен;
      - code_page - набор символов, используемый в документе (например "Latin I");
  - tags - примечательные замечания обо всем документе, взятые из шаблонов и свойств макросов.

Информация о структуре файлов Microsoft Office

```
{
 "data": {
           ...
  "attributes" : {
    ...
    "office_info": {
     "documment_summary_info": {"scale": <boolean>,
                                "links_dirty": <boolean>,
                                "line_count": <int>,
                                "hyperlinks_changed": <boolean>,
                                "characters_with_spaces": <int>,
                                "version": <int>,
                                "shared_document": <boolean>,
                                "paragraph_count": <int>,
                                "company": "<string>",
                                "code_page": "<string>"},
     "entries": [{"clsid": "<string>",
                  "clsid_literal": "<string>",
                  "name": "<string>",
```

(continues on next page)

(continued from previous page)

```
                "type_literal": "<string>",
                "sid": <int>,
                "size": <int>,} ... ],
      "ole": {"macros": [{"vba_code": "<string>",
                          "stream_path": "<string>",
                          "vba_filename": "<string>",
                          "patterns": ["<strings>"],
                          "length": <int>,
                          "properties": ["<strings>"]}] ...,
              "num_macros": <int>},
      "summary_info": {"last_author": "<string>",
                       "creation_datetime": "<string:%Y-%m-%d %H:%M:%S>",
                       "template": "<string>",
                       "author": "<string>",
                       "page_count": <int>,
                       "last_saved": "<string:%Y-%m-%d %H:%M:%S>",
                       "edit_time": <int>,
                       "word_count": <int>,
                       "revision_number": "<string>",
                       "last_printed": "<string:%Y-%m-%d %H:%M:%S>",
                       "application_name": "<string>",
                       "title": "<string>",
                       "character_count": <int>,
                       "security": <int>,
                       "code_page": "<string>"},
      "tags": ["<strings>"]
    }
   }
  }
}
```

3.1.18 openxml_info

Информация об Microsoft OpenXML файлах.

openxml_info возвращает информацию о структуре файлов Microsoft Office Open XML (Office 2007+). Включая информацию (Word) .docx, .docm, .dotx, .dotm, (Excel) .xlsx, .xlsm, .xltx, .xltm, (PowerPoint) .pptx, .pptm, .potx, .potm, .ppam, .ppsx, .ppsm, .sldx, .sldm.

- content_types - сведения о типе MIME для частей пакета;
- docprops_app - некоторые свойства файла и поля могут отличаться в зависимости от типа файла:
  - TotalTime - общее время редактирования документа;
  - Words - количество слов;
  - ScaleCrop - режим отображения миниатюр;
  - SharedDoc - если документ является общедоступным;
  - Company - имя компании;
  - Lines - число строк;
  - AppVersion - версия приложения (в числовой форме);
  - LinksUpToDate - true означает, что гиперссылки обновляются, false - в противном случае;
  - Pages - количество страниц;

  - Application - имя приложения (например “Microsoft Office Word”);
  - CharactersWithSpaces - количество символов, включая пробелы;
  - Characters - количество символов без пробелов;
  - Paragraphs - количество частей;
  - Template - имя шаблона, используемого в документе;
  - DocSecurity: ``0 если пароль для документа не установлен;
  - HyperlinksChanged - одна или несколько гиперссылок в этой части были обновлены производителем исключительно в этой части;
- ‘‘ocprops_core: core properties for any Office Open XML document
  - dc:creator - создатель документа;
  - cp:revision - редакции документа;
  - dcterms:created - дата создания в формате “%Y-%m-%dT%H:%M:%SZ”;
  - dcterms:modified - дата последней модификации в формате “%Y-%m-%dT%H:%M:%SZ”;
  - cp:lastModifiedBy - пользователь, который сделал последнюю модификацию;
  - cp:lastPrinted - дата последней печати документа в формате “%Y-%m-%dT%H:%M:%SZ”;
- file_type - тип файла ("docx", "pptx", и т. д.);
- ole - макросы найденные в содержимом OLE:
  - macros - подробная информация о макросах:
    - vba_code - код макроса;
    - stream_path - путь в дереве хранения OLE;
    - vba_filename - имя макроса;
    - patterns - примечательные паттерны в макросе ("exe-pattern", "url-pattern", и т. д.);
    - lengh - длина макроса;
    - properties - примечательные свойсвта макроса ("obfuscated", "run-file", и т. д.);
  - num_macros - количестов макросов;
  - rels - отношения для файлов внутри пакета;
  - tags - примечания о интересном содержимом в пакете (например "macros").
  - type_content - информация, специфичная для каждого формата файла:
    - (Word, PowerPoint):
      - languages - ссылки на найденные языки (название и номер);
    - (Excel):
      - codifications - ссылки на используемые кодовые страницы (имя и номер);
      - workbook - информация о книге:
      - sheets - количество листов;
      - lowestEdited - самая низкая отредактированная версия;
      - calcPr - версия Excel.

- lastEdited - последняя отредактированная версия;
- rupBuild - версия сборки;
- language_guess - предполагаемый используемый язык (имя и номер);

* (Excel, PowerPoint):
  - printers - используется для печати этого документа.

Информация о Microsoft Office openxml

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "openxml_info": {
        "content_types": ["<strings>"],
        "docprops_app": {"TotalTime": "<string>",
                         "Words": "<string>",
                         "ScaleCrop": "<string>",
                         "SharedDoc": "<string>",
                         "Company": "<string>",
                         "Lines": "<string>",
                         "AppVersion": "<string>",
                         "LinksUpToDate": "<string>",
                         "Pages": "<string>",
                         "Application": "<string>",
                         "CharactersWithSpaces": "<string>",
                         "Characters": "<string>",
                         "Paragraphs": "<string>",
                         "Template": "<string>",
                         "DocSecurity": "<string>",
                         "HyperlinksChanged": "<string>"},
        "docprops_core": {"dc:creator": "<string>",
                          "cp:revision": "<string>",
                          "dcterms:created": "<string>",
                          "dcterms:modified": "<string>",
                          "cp:lastModifiedBy": "<string>",
                          "cp:lastPrinted": "<string>"},
        "file_type": "<string>",
        "ole": {"macros": [{"vba_code": "<string>",
                            "stream_path": "<string>",
                            "subfilename": "<string>",
                            "vba_filename": "<string>",
                            "patterns": ["<strings>"],
                            "length": <int>,
                            "properties": ["<strings>"]}, ... ],
                "num_macros": <int>},
        "rels": ["<strings>"],
        "tags": ["<strings>"],
        "type_content": {"languages": {"<string>": <int>, ... },
                         "codifications" : [["<string>", <int>] ... ],
                         "workbook": {"sheets": <int>,
                                      "lowestEdited": "<string>",
                                      "calcPr": "<string>",
                                      "lastEdited": "<string>",
```

(continues on next page)

(continued from previous page)

```
                        "rupBuild": "<string>"},
                    "language_guess": [["<string>", <int>], ... ],
                    "printers": ["<strings>"]}

    }
   }
  }
}
```

3.1.19 packers

Информация об упаковщике, используемом в файле.

packers - определtybt упаковщиков PE-файлов, используемых в Windows с помощью нескольких утилит и антивирусных средств.

- ключи - это названия утилит, значения - это идентифицированные упаковщики.

PEiD идентификатор упаковщика

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "packers": {"<string>": "<string>", ... }
    }
  }
}
```

3.1.20 pdf_info

Информация об Adobe PDF файлах.

pdf_info возвращает информацию о структуре файлов PDF:

- acroform - содержание Acroforms;
- automation - автоматическое действие, выполняемое при просмотре документа;
- embedded_file - содержимое встроенного файла;
- encrypted - документ имеет DRM или нуждается в пароле для чтения;
- flash - содержит встроенный Flash;
- header - заголовок документа (например %PDF-1.7);
- javascript - документ содержит JavaScript;
- jbig2_compression - документ сжат с применением JBIG2;
- js - документ содержит JavaScript;
- num_endobj - количество завершений объекта;
- num_endstream - количество завершений потока;

- num_launch_actions - количество запускаемых действий;
- num_obj - количество объектов;
- num_object_streams - количество потоков объектов;
- num_pages - количество страниц;
- num_stream - количество потоков;
- open action - автоматическое действие, выполняемое при просмотре документа;
- startxref - эта запись присутствует в документе;
- suspicious_colors - устанавливается, если количество цветов выражается более чем 3 байтами;
- trailer - содержит раздел трейлера;
- xref - таблица перекрестных ссылок.

Структура Acrobat PDF файлов

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "pdf_info": {
        "acroform": <int>,
        "autoaction": <int>,
        "embedded_file": "<string>",
        "encrypted": <int>,
        "flash": <int>,
        "header": "<string>",
        "javascript": <int>,
        "jbig2_compression": <int>,
        "js": <int>,
        "num_endobj": <int>,
        "num_endsctream": <int>,
        "num_launch_actions": <int>,
        "num_obj": <int>,
        "num_object_streams": <int>,
        "num_pages": <int>,
        "num_stream": <int>,
        "openaction": <int>,
        "startxref": <int>,
        "suspicious_colors": "<string>",
        "trailer": <int>,
        "xref": <int>
      }
    }
  }
```

3.1.21 pe_info

Информация о файлах формата Microsoft Windows Portable Executable.

pe_info возвращает информацию о структуре Майкрософт Windows PE-файлов (то есть исполняемые файлы, динамические библиотеки, драйверы и т. д.): разделы, точка входа, ресурсы, импорт, экспорт и т. д.

- debug - отладочная информация, если таковая имеется:
  - codeview‘‘ - CodeView отладочная информация, если таковая имеется:
    - age - почтоянно увеличивающееся значение;
    - guid - уникальный идентификатор;
    - name - путь к PDB-файлу;
    - signature - содержит "RSDS";
  - offset - размещение отладочной информации;
  - timedatestamp - метка времени в формате “%a %b %d %H:%M:%S %Y”;
  - type_str - человеко-читаемая версия информации о типе отладки;
  - type - информация о типе отладки;
  - size - размер блока отладочной информации;
- entry_point - точка входа;
- exports - экспортируемые функции;
- imphash - хэш секции импорта;
- imports - словарь с именами DLL в качестве ключей и списками импортированных функций в качестве значений;
- machine_type - платформа;
- overlay - информация о содержимом секции оверлея PE-файла (если эта секция присутствует в файле):
  - chi2 - проверочное значение хи-квадрат байтов из содержимого оверлея;
  - entropy - значение энтропии оверлея;
  - filetype - если возможно идентифицировать конкретный формат файла, его тип указывается здесь;
  - offset - расположение начала оверлея;
  - md5 - хэш содержимого оверлея;
  - size - размер в байтах;
- ‘‘resource_details: if the PE contains resources, some info about them.
  - chi2 - проверочное значение хи-квадрат байтов из содержимого ресурсов;
  - entropy - значение энтропии содержимого ресурсов.
  - filetype - если возможно идентифицировать конкретный формат файла, его тип указывается здесь;
  - lang - язык ресурса;
  - sha256 - хэш содержимого ресурса;
  - type - тип ресурса;
- resource_langs: информация о языках, найденных в ресурсе (имя и номер);

- resource_types: информация о типе ресурса (тип и номер);
- sections - информация о PE секциях:
  - entropy - значение энтропии содержимого секции;
  - md5 - хэш секции;
  - name - section name.
  - raw_size - размер инициализированных данных на диске (в байтах);
  - virtual_address - адрес первого байта раздела при загрузке в память, относительно базы;
  - virtual_size - общий размер раздела при загрузке в память (в байтах);
- timestamp - время компиляции в формате Unix Epoch.

Microsoft Windows PE-файл

```
{
 "data": {
           ...
  "attributes" : {
   ...
   "pe_info": {
    "debug": [{"codeview": {"age": <int>,
                            "guid": "<string>",
                            "name": "<string>",
                            "signature": "RSDS"},
               "offset": <int>,
               "size": <int>,
               "timedatestamp": "<string:%a %b %d %H:%M:%S %Y>",
               "type": <int>,
               "type_str": "<string>"}, ... ],
    "entry_point": <int>,
    "exports": ["<string>", ... ],
    "imphash": "<string>",
    "imports": {"<string>": ["<strings>"], ... },
    "machine_type": <int>,
    "overlay": {"chi2": <float>,
                "filetype": "<string>",
                "entropy": <float>,
                "offset": <int>,
                "md5": "<string>",
                "size": <int>},
    "resource_details": [{"chi2": <float>,
                          "entropy": <float>,
                          "filetype": "<string>",
                          "lang": "<string>",
                          "sha256": "<string>",
                          "type": "<string>"}, ... ],
    "resource_langs": {"<string>": <int>, ... },
    "resource_types": {"<string>": <int>, ... },
    "sections": [{"entropy": <float>,
                  "md5": "<string>",
                  "name": "<string>",
                  "raw_size": <int>,
                  "virtual_address": <int>,
```

(continues on next page)

(continued from previous page)

```
                "virtual_size": <int>}, ... ],
      "timestamp": <int>
    }
   }
  }
}
```

3.1.22 rombios_info

Информация о BIOS, EFI, UEFI и связанных с ними архивах.

rombios_info показывает информацию о файлах прошивок и встроенных программ.

- acpi_tables - таблицы ACPI (Advanced Configuration and Power interface), имеющиеся в прошивке;
- apple_data - метаданные из файлов прошивки Apple EFI, представленные в виде списка кортежей, с ключом и значениями. Некоторые типичные ключи и значения:
  - Board ID - идентификатор сборки;
  - Built by - наименование сборщика файла;
  - Date - дата создания файла в формате “%a %b %m %H:%M:%S %Z %Y”;
  - Revision - редакция сборки;
  - ROM Version - версия ROM;
  - Buildcave ID - идентификатор сборки внутренней прошивки;
- bios_information - некторые детали о файле BIOS:
  - BIOS Release - версия релиза;
  - Characteristics - характеристики BIOS, такие как "PCI supported", "8042 keyboard supported" и т. д.;
  - ROM Size - размер ROM в удобном для чтения формате (например "2MB");
  - Release Date - дата релизав формате “%m/%d/%Y”;
  - Runtime Size - размер среды выполнения в удобном для чтения формате (например "64. 0KB");
  - Starting Address Segment - в шестнадцатеричном формате;
  - Vendor - поставщик BIOS;
  - Version - полная версия файла BIOS;
- certs - сертификаты, найденные в файле прошивки:
  - valid_from - дата начала действия сертификата в формате “%Y-%m-%d %H:%M%S”;
  - subject - уникальные имена RDN и их значения;
  - valid_to- дата окончания действия сертификата в формате “%Y-%m-%d %H:%M%S”;
  - issuer - имя выпускающего удостоверяющего центра RDN;
- executable_files - количество обнаруженных исполняемых файлов;
- firmware_volumes - количество найденных томов прошивки;
- format - формат пакета (например "ROMFLASH_HEADER");

- manufacturer_strings - ссылки на производителей BIOS;
- nvar_variable_names - обнаруженные переменные NVAR;
- raw_objects - количество необработанных объектов;
- sections - количество секций;
- smbios_data - обнаруженные ключи и значения данных SMBIOS:
  - Version - версия файла;
- system_information - информация о платформе для этого файла:
  - SKU Number - SKU номер;
  - UUID - уникальный идентификатор;
  - Family - номер семейства;
  - Serial Number - серийный номер;
  - Version - версия;
  - Product Name - наименование;
  - Manufacturer - производитель BIOS.

Образ прошивки

```
{
 "data": {
           ...
  "attributes" : {
    ...
    "rombios_info": {
     "acpi_tables": ["<strings>"],
     "apple_data": [["<string>", "<string>"], ... ],
     "bios_information": {"BIOS Release": "<string>",
                          "Characteristics": ["<strings>"],
                          "ROM Size": "<string>",
                          "Release Date": "<string:%m/%d/%Y>",
                          "Runtime Size": "<string>",
                          "Starting Address Segment": "<string>",
                          "Vendor": "<string>",
                          "Version": "<string>"},
     "certs":[{"issuer": "<string>",
               "subject": "<string>",
               "valid_from": "<string:%Y-%m-%d %H:%M:%S>",
               "valid_to": "<string:%Y-%m-%d %H:%M:%S>"}, ... ],
     "executable_files": <int>,
     "firmware_volumes": <int>,
     "format": "<string>",
     "manufacturer_strings": {"<string>": <int>, ... },
     "nvar_variable_names": ["<strings>"],
     "raw_objects": <int>,
     "sections": <int>,
     "smbios_data": {"<string>": "<string>", ... },
     "system_information": {"Family": "<string>",
                            "Manufacturer": "<string>",
                            "Product Name": "<string>",
```

(continues on next page)

(continued from previous page)

```
                    "SKU Number": "<string>",
                    "Serial Number": "<string>",
                    "UUID": "<string>",
                    "Version": "<string>"}
    }
   }
  }
}
```

3.1.23 rtf_info

Информация о файлах формата Microsoft Rich Text.

rtf_info возвращает информацию о Microsoft RTF файлах.

- document_properties - структурированные метаданные о документе:
  - non_ascii_characters - количество не ASCII символов в документе;
  - embedded_drawings- количество рисунков, содержащихся в документе;
  - rtf_header - заголовок RTF (например "rtf1");
  - default_ansi_codepage - используемая кодовая страница (например "Western European");
  - read_only_protection - True если файл предназначен только для чтения;
  - user_protection - user protection.
  - default_character_set - используемый набор символов (например "ANSI");
  - custom_xml_data_properties - количество пользовательских объектов XML-данных;
  - dos_stubs - количество найденных “заглушек” DOS;
  - objects - список содержащихся объектов, с описанием типа и класса;
  - embedded_pictures - количество встроенных картинок;
  - default_languages - языки, обнаруженные в документе;
  - longest_hex_string - самая длинная шестнадцатеричная строка найденная в документе;
- summary_info - другие свойства документа:
  - revision_time - дата последнего изменения в формате “%Y-%m-%d %H:%M:%S”;
  - version_number - номер версии документа;
  - editing_time - общее время редактирования в минутах;
  - number_of_pages - number of pages in the document.
  - creation_time - дата создания в формате “%Y-%m-%d %H:%M:%S”;
  - operator - имя пользователя, создавшего документ;
  - number_of_non_whitespace_characters - количество символов не являющимися пробелами;
  - version - версия RTF отраженная в документе;
  - number_of_characters - количество символов в документе;
  - number_of_words - количество слов в документе.

Microsoft RTF файл

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "rtf_info": {
        "document_properties": {"non_ascii_characters": <int>,
                                "embedded_drawings": <int>,
                                "rtf_header": "<string>",
                                "default_ansi_codepage": "<string>",
                                "read_only_protection": <boolean>,
                                "user_protection": <boolean>,
                                "default_character_set": "<string>",
                                "custom_xml_data_properties": <int>,
                                "dos_stubs": <int>,
                                "objects": [{"type": "<string>",
                                           "class": "<string>"} ... ],
                                "embedded_pictures": <int>,
                                "default_languages": ["<strings>"],
                                "longest_hex_string": <int>},
        "summary_info": {"revision_time": "<string:%Y-%m-%d %H:%M:%S>",
                         "version_number": <int>,
                         "editing_time": <int>,
                         "number_of_pages": <int>,
                         "creation_time": "<string:%Y-%m-%d %H:%M:%S>",
                         "operator": "<string>",
                         "number_of_non_whitespace_characters": <int>,
                         "version": <int>,
                         "number_of_characters": <int>,
                         "number_of_words": <int>}
      }
    }
  }
}
```

3.1.24 signature_info

Информация о подписи PE-файлов.

signature_info содержит информацию о цифровой подписи для Windows Executable файлов, извлеченную с помощью утилиты Sigcheck.

- comments - из ресурсов файла (если обнаружено);
- copyright - из ресурсов файла (если обнаружено);
- counter signers - строка со счетчиком подписей Common Names;
- counter signers details - список словарей, детализирующих значение каждого сертификата из счетчика:
  - algorithm - алгоритм, используемый для создания пар ключей;
  - cert issuer - компания, выпустившая сертификат;
  - name - отличительное имя сертификата;
  - serial number - в шестнадцатеричном виде с разделением пробелом между байтами;

  - status - может иметь значение "Valid" или указать проблему с сертификатом, если таковая имеется (например "This certificate or one of the certificates in the certificate chain is not time valid.");
  - thumbprint - хэш сертификата в шестнадцатеричном представлении.
  - valid from - дата начала действия в формате “%H:%M %p %m/%d/%Y”;
  - valid to - дата истечения срока действия в формате “%H:%M %p %m/%d/%Y”;
  - valid usage - для чего может быть использован сертификат (например "Code Signing");
- description - из ресурсов файла (если обнаружено);
- file version -из ресурсов файла (если обнаружено);
- internal name - из ресурсов файла (если обнаружено);
- original name - из ресурсов файла (если обнаружено);
- product - из ресурсов файла (если обнаружено);
- signers - строка с подписывающими Common Names;
- singers details - список словарей с подробным описанием каждого сертификата подписавшего:
  - algorithm - алгоритм, используемый для создания пар ключей;
  - cert issuer - компания, выпустившая сертификат;
  - name - отличительное имя сертификата;
  - serial number - в шестнадцатеричном виде с разделением пробелом между байтами;
  - status - может иметь значение "Valid" или указать проблему с сертификатом, если таковая имеется (например "This certificate or one of the certificates in the certificate chain is not time valid.");
  - thumbprint - хэш сертификата в шестнадцатеричном представлении.
  - valid from - дата начала действия в формате “%H:%M %p %m/%d/%Y”;
  - valid to - дата истечения срока действия в формате “%H:%M %p %m/%d/%Y”;
  - valid usage - для чего может быть использован сертификат (например "Code Signing");
- signing date - дата подписания файла в формате “%H:%M %p %m/%d/%Y”;
- verified - статус сертификата. Возможные варианты: "Signed", "Unsigned", или если есть какие-либо проблемы с подписью (например "A - certificate was explicitly revoked by its issuer.");
- x509 - список сертификатов, найденных в файле, в случае, если Sigcheck не возвращает информацию о них:
  - algorithm - алгоритм, используемый для создания пар ключей;
  - cert issuer - компания, выпустившая сертификат;
  - name - отличительное имя сертификата;
  - serial number - в шестнадцатеричном виде с разделением пробелом между байтами;
  - thumbprint - хэш сертификата в шестнадцатеричном представлении.
  - valid from - дата начала действия в формате “%H:%M %p %m/%d/%Y”;
  - valid to - дата истечения срока действия в формате “%H:%M %p %m/%d/%Y”;
  - valid usage - для чего может быть использован сертификат (например "Code Signing").

JSON

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "signature_info": {
        "comments": "<string>",
        "copyright": "<string>",
        "counter signers": "<string>",
        "counter signers details": [{"algorithm": "<string>",
                                     "cert issuer": "<string>",
                                     "name": "<string>",
                                     "serial number": "<string>",
                                     "status": "<string>",
                                     "thumbprint": "<string>",
                                     "valid from": "<string:%H:%M %p %m/%d/%Y>",
                                     "valid to": "<string:%H:%M %p %m/%d/%Y>",
                                     "valid usage": "<string>"} ... ],
        "description": "<string>",
        "file version": "<string>",
        "internal name": "<string>",
        "original name": "<string>",
        "product": "<string>",
        "signers": "<string>",
        "signers details": [{"algorithm": "<string>",
                             "cert issuer": "<string>",
                             "name": "<string>",
                             "serial number": "<string>",
                             "status": "<string>",
                             "thumbprint": "<string>",
                             "valid from": "<string:%H:%M %p %m/%d/%Y>",
                             "valid to": "<string:%H:%M %p %m/%d/%Y>",
                             "valid usage": "<string>"}, ... ],
        "signing date": "<string:%H:%M %p %m/%d/%Y>",
        "verified": "<string>",
        "x509": [{"algorithm": "<string>",
                  "cert issuer": "<string>",
                  "name": "<string>",
                  "serial number": "<string>",
                  "thumbprint": "<string>",
                  "valid from": "<string:%H:%M %p %m/%d/%Y>",
                  "valid to": "<string:%H:%M %p %m/%d/%Y>",
                  "valid_usage": "<string>"}, ... ]
    }
  }
}
```

3.1.25 ssdeep

CTPH хэш содержимого файла.

ssdeep - программа для вычисления контекстно-зависимого кусочного хэша. Также называемый нечеткими хэшем, он позволяет идентифицировать похожие файлы.

ssdeep

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "ssdeep": "<string>"
    }
  }
}
```

3.1.26 swf_info

Информация о Adobe Shockwave Flash файлах.

swf_info возвращает информацию о файлах Shockwave Flash/Small Web Format:

- compression - тип используемого сжатия (наптимер zlib);
- duration - длина медиа-контента в секундах;
- file_attributes- особые атрибуты (например ActionScript3, UseGPU);
- flash_packages - список используемых Flash пакетов;
- frame_count- количество фреймов;
- frame_size - размер фреймов;
- metadata - содержимое метаданных файла;
- num_swf_tags - количество тэгов SWF;
- num_unrecognized_tags: количество нераспознанных тегов;
- suspicious_strings - список найденных подозрительных строк;
- suspicious_urls - список найденных подозрительных URL;
- tags - примечательные замечания о файле (например get-url, ext-interface);
- version - версия SWF.

SWF файл

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "swf_info": {
        "compression": "<string>",
        "duration": <float>,
        "file_attributes": ["<strings>"],
        "flash_packages": ["<strings>"],
        "frame_count": <int>,
        "frame_size": "<string>",
        "metadata": "<string>",
```

(continues on next page)

(continued from previous page)

```
      "num_swf_tags": <int>,
      "num_unrecognized_tags": <int>,
      "suspicious_strings": ["<strings>"],
      "suspicious_urls": ["<strings>"],
      "tags": ["<strings>"],
      "version": <int>
    }
   }
  }
}
```

3.1.27 trid

Тип файла идентифицированный с помощью утилиты TrID.

trid - утилита, предназначенная для идентификации типов файлов по их бинарным сигнатурам. Может дать несколько результатов, упорядоченных от более высокой до более низкой вероятности идентификации формата файла (в процентах).

TrID

```
{
  "data": {
            ...
    "attributes" : {
      ...
      "trid": [
        {"file_type": "<string>", "probability": <float>}, ...
      ]
    }
  }
}
```

3.2 Поведение файлов (file behaviour)

Отчеты о поведении файлов.

Отчеты о поведении файлов получаются либо с помощью функции GET /files/{id}/behavior, либо с помощью анализа поведения в песочнице . Они суммируют наблюдаемое поведение во время выполнения или открытия файла. Обратите внимание, что некоторые из этих действий могут быть инициированы дочерними элементами рассматриваемого файла.

Объект file_behaviour содержит следующие атрибуты:

3.2.1 DnsLookup

DNS-запросы.

- hostname <string> - имя хоста DNS-запроса;
- resolved_ips <string array> - все разрешенные IP-адреса могут быть пустыми на NX домене.

3.2.2 DroppedFile

Сброшенные файлы - это файлы, специально созданные и записанные во время анализа поведения. Это может быть результатом загрузки содержимого из интернета и записи его в файл, распаковки файла, сброса некоторого содержимого в файл и т. д.

- path <string> - полный путь к файлу, включая имя файла;
- sha256 <string> - SHA-256 хэш файла.

3.2.3 BehaviourTag

Поведение в Sandbox было помечено сложной операцией:

- DETECT_DEBUG_ENVIRONMENT
- DIRECT_CPU_CLOCK_ACCESS
- LONG_SLEEPS
- SELF_DELETE - файл удаляется сам по себе при выполнении.
- HOSTS_MODIFIER - файл local hosts изменен.
- INSTALLS_BROWSER_EXTENSION - устанавливает BHO, расширение Chrome и т. д.
- PASSWORD_DIALOG - отображается какая-то подсказка для ввода пароля.
- SUDO - повышает привилегии до администратора.
- PERSISTENCE - использует механизмы устойчивости, чтобы пережить перезагрузку.
- SENDS_SMS
- CHECKS_GPS
- FTP_COMMUNICATION
- SSH_COMMUNICATION
- TELNET_COMMUNICATION
- SMTP_COMMUNICATION
- MYSQL_COMMUNICAION
- IRC_COMMUNICATION
- SUSPICIOUS_DNS - возможен DGA (алгоритм генерации домена).
- SUSPICIOUS_UDP - большое количество различных UDP-соединений, это часто помогает выявить P2P.
- BIG_UPSTREAM - большой исходящий сетевой трафик.
- TUNNELING - наблюдается туннелирование сети, например, VPN.
- CRYPTO - использует API, связанные с криптографией.
- TELEPHONY - использует API, связанные с телефонией.
- RUNTIME_MODULES - динамически загружает библиотеки DLL или дополнительные компоненты.
- REFLECTION - выполняет отображение вызовов.

3.2.4 FileCopy

Объект, описывающий копирование или перемещение файла:

- source <string> - полный путь к исходному файлу.
- destination <string> - полный путь к файлу назначения.

3.2.5 HttpConversation

HTTP-вызовы.

- RequestMethod - один из:
  - GET
  - HEAD
  - POST
  - PUT
  - DELETE
  - TRACE
  - OPTIONS
  - CONNECT
  - PATCH
- url - полное имя хоста и путь к указанному URL-адресу.
- request_headers ключи и значения:
  - key - например Content-Type;
  - value - например image/jpeg;
- response_headers - ключи и значения заголовков ответов.
- response_status_code - код состояния ответа, например 200.
- response_body_filetype
- response_body_first_ten_bytes

3.2.6 IpTraffic

IP-трафик:

- destination_ip <string> - IP-адрес.
- destination_port <integer> - номер порта.
- transport_layer_protocol - один из:
  - ICMP
  - IGMP
  - TCP
  - UDP

  - ESP
  - AH
  - L2TP
  - SCTP

3.2.7 PermissionCheck

Записывает запрос, чтобы узнать, имеет ли данный компонент/пакет/процесс/служба определенное разрешение.

- permission <string> - например: android.permission.INTERNET.
- owner <string> - имя приложения, которому было предоставлено проверяемое разрешение.

3.2.8 Process

- process_id <string> - ID процесса.
- name <string> - имя процесса.
- time_offset <integer> - начало наблюдения. Секунды с момента начала исполнения.
- children <Process array> - массив этого объекта Process. Позволяет построить дерево процессов.

3.2.9 Sms

Отправлено SMS сообщение.

- destination <string> - номер телефона, на который отправляется SMS.
- body <string> - текст сообщения.

3.2.10 VerdictTag

Вердикты для пометки образца поведения в песочнице:

- CLEAN - чистый, занесенный в белый список или незамеченный.
- MALWARE - должно быть определено как вредоносное ПО
- GREYWARE - PUA, PUP (возможно, нежелательная программа).
- RANSOM - вымогатель или криптор.
- PHISHING - пытается обмануть пользователя, чтобы получить его учетные данные.
- BANKER - банковский троян.
- ADWARE - отображает нежелательную рекламу.
- EXPLOIT - содержит или запускает эксплойт.
- EVADER - содержит логику, позволяющую уклониться от анализа.
- RAT - троян для удаленного доступа, может прослушивать входящие соединения.
- TROJAN - троян или бот.
- SPREADER распространяется на USB, других накопителях, по сети и т. д.

3.3 Домены (domains)

Наряду с URL-адресами VirusTotal хранит информацию о сетевых местоположениях, таких как домены и IP-адреса. В этом разделе будет рассмотрена информация, предоставляемая объектами типа domain.

Объекты типа domain представляют собой информацию о домене или FQDN, и могут быть получены путем поиска уже существующего домена по его идентификатору, по его связи с другими объектами или по другим значениям при поиске в службах VT Enterprise services.

Помните, что в отличие от отчетов о файлах и URL-адресах, сетевое расположение (такое как домены и IP-адреса) не записывает вердикты партнеров для рассматриваемого ресурса. Вместо этого эти отчеты включают всю недавнюю активность, которую VirusTotal наблюдал для ресурса, а также контекстную информацию о нем. Эта информация включает в себя:

- id - для идентификации используется доменное имя или FQDN.
- Categories - сопоставление, которое связывает службы классификации с категорией, которую они назначают домену. К таким службам относятся, в частности: Alexa, BitDefender, TrendMicro, Websense ThreatSeeker и т. д.
- creation_date - дата, когда домен был впервые включен в набор данных VirusTotal.
- last_update_date - дата последнего обновления информации о домене.
- registrar - компания, которая зарегистрировала домен.
- reputation - оценка домена, рассчитанная по голосам сообщества VirusTotal.
- total_votes - невзвешенное количество голосов от сообщества, разделенное на “harmless” и “maliciousus”.
- whois - информация “Whois”, возвращенная с соответствующего whois-сервера.
- whois_date - дата последнего обновления записи whois в VirusTotal.

Note: Репутация каждого домена определяется сообществом Virustotal (в которое входят зарегистрированные пользователи). Пользователи, голосующие за домены, в свою очередь, сами имеют репутацию, при этом оценка сообщества включает в себя все голоса, с учетом репутациеи пользователей, которые проголосовали за тот или иной домен. Отрицательные (красные) оценки указывают на злонамеренность, в то время как положительные (зеленые) оценки отражают безвредность. Чем больше абсолютное число, тем больше вы можете доверять данной оценке. Вы можете прочитать больше об этом в этой статье сообщества.

Объект типа “Domain”

```
{
  "data": {
    "type": "domain"
    "id": "<DOMAIN>",
    "links": {
      "self": "https://virustotal.com/api/v3/domains/<DOMAIN>"
    },
    "attributes": {
      "categories": {
        "<SERVICE>": "<string>"
      },
```

(continues on next page)

(continued from previous page)

```
    "creation_date": <int:timestamp>,
    "last_update_date": <int:timestamp>,
    "registrar": "<string>",
    "reputation": <int>,
    "total_votes": {
      "harmless": <int>,
      "malicious": <int>
    },
    "whois": "<string>",
    "whois_date": <int:timestamp>
  },
 }
}
```

3.3.1 communicating_files

Отношение communicating_files перечислит все файлы, которые генерируют какой-либо трафик для данного домена в какой-то момент выполнения этих файлов. Это отношение может быть получено с помощью API функции relationships. Ответ содержит поле:

data список объектов типа “File” (см. Файлы (files)). Это представление будет содержать раздел attributes файла.

/domains/{domain}/communicating_files

```
{
 "data": [
   <FILE_OBJECT>,
   <FILE_OBJECT>,
   ...
 ],
 "links": {
   "next": <string>,
   "self": <string>
 },
 "meta": {
   "cursor": <string>
 }
}
```

3.3.2 downloaded_files

Отношение downloaded_files возвращает список файлов, которые были доступны с URL-адреса в данном домене или поддомене в определенный момент. Это отношение может быть получено с помощью API функции GET /domains/{domain}/{relationship}. Ответ содержит поле:

data список объектов типа “File” (см. Файлы (files)). Это представление будет содержать раздел attributes файла.

/domains/{domain}/communicating_files

```
{
  "data": [
    <FILE_OBJECT>,
    <FILE_OBJECT>,
    ...
  ],
  "links": {
    "next": <string>,
    "self": <string>
  },
  "meta": {
    "cursor": <string>
  }
}
```

3.3.3 graphs

Отношение graphs возвращает список графиков, содержащих данный домен. Это отношение может быть получено с помощью API функции GET /domains/{domain}/{relationship}. Ответ содержит поле:

data список объектов типа “Graph”. Это представление будет содержать раздел attributes графика.

/domains/{domain}/graph

```
{
  "data": [
    <GRAPH_OBJECT>,
    ...
  ],
  "links": {
    "self": <url>
    }
}
```

3.3.4 referrer_files

Отношение referrer_files возвращает список файлов, содержащих данный домен в своих строках. Это отношение может быть получено с помощью API функции GET /domains/{domain}/{relationship}. Ответ содержит поле:

data список объектов типа “File” (см. Файлы (files)). Это представление будет содержать раздел attributes файла.

/domains/{domain}/referrer_files

```
{
  "data": [
    <FILE_OBJECT>,
    <FILE_OBJECT>,
```

(continues on next page)

(continued from previous page)

```
    ...
  ],
  "links": {
    "next": <string>,
    "self": <string>
  },
  "meta": {
    "cursor": <string>
  }
}
```

3.3.5 resolutions

Отношение resolutions возвращает список прошлых и текущих разрешений IP-адресов для данного домена или поддомена. Это отношение может быть получено с помощью API функции GET /domains/{domain}/{relationship}. Ответ содержит поле:

data список объектов типа "Resolution". Это представление будет содержать раздел attributes объекта.

/domains/{domain}/resolutions

```
{
  "data": [
    <RESOLUTION_OBJECT>,
    <RESOLUTION_OBJECT>,
    ...
  ],
  "links": {
    "next": <string>,
    "self": <string>
  },
  "meta": {
    "cursor": <string>
  }
}
```

Объект "Resolutions" (см. Resolution object) включает в себя следующую информацию:

- id - объединение IP-адреса и домена.
- date - метка времени (дата), когда был сделан запрос на разрешение.
- host_name - домен или поддомен, запрошенный у резолвера.
- ip_address - IP-адрес, на который указывал домен в заданную дату.
- resolver - DNS-сервер, на который был отправлен запрос на разрешение.

Resolution object

```
{
  "type": "resolution",
  "id": <string>,
  "attributes": {
```

(continues on next page)

(continued from previous page)

```
            "date": <timestamp>,
            "host_name": <string>,
            "ip_address": <string>,
            "resolver": <string>
        },
 "links": {
        "self": <string>
  }
}
```

3.3.6 siblings

С помощью отношения sibling можно получить список поддоменов на том же уровне, что и данный поддомен для домена, вместе с информацией о них. Это отношение может быть получено с помощью API функции GET /domains/{domain}/{relationship}. Ответ содержит поле:

data список объектов типа ” Domain” (см. Домены (domains)). Это представление будет содержать раздел attributes объекта.

/domains/{domain}/siblings

```
{
  "data": [
    <DOMAIN_OBJECT>,
    <DOMAIN_OBJECT>,
    ...
  ],
  "links": {
    "next": <string>,
    "self": <string>
  },
  "meta": {
    "cursor": <string>
  }
}
```

3.4 IP-адреса (IP addresses)

IPv4-адреса - это сетевые адреса, о которых VirusTotal также хранит информацию. Ниже приводится описание полей, хранящихся в объектах типа IP addresses.

IP-адреса, также как домены и сетевые местоположения связаны с объектами файлов и URL-адресов во многих отношениях. Именно поэтому их можно получить по связи с другими объектами, а также при поиске в службах VT Enterprise или просто путем поиска уже существующего IP-адреса.

Обратите внимание, что в качестве объектов домена, представления IP-адресов не записывают вердикты партнеров для рассматриваемого ресурса. Вместо этого, отчеты включают в себя всю недавнюю активность, которую VirusTotal видел для ресурса, а также контекстную информацию о нем. Эти детали включают в себя:

- id - идентификатор объекта в виде строки с IPv4-адресом;
- as_owner - владелец объекта Autonomous System, которому принадлежит IP-адрес;

- asn - номер Autonomous System, которому принадлежит IP-адрес;
- continent - континент, на котором размещен IP (код континента по ISO-3166);
- country - страна, в которой размещен IP (код страны по ISO-3166);
- network - диапазон IPv4 сети, к которому принадлежит IP-адрес;
- regional_internet_registry - RIR (один из пяти региональных регистраторов: AFRINIC, ARIN, APNIC, LACNIC или RIPE NCC);
- reputation - оценка домена, рассчитанная исходя из результатов голосования сообщества VirusTotal;
- total_votes - Unweighted number of total votes from the community, divided in “harmless” and “malicious”.

Note: Репутация каждого домена определяется сообществом Virustotal (в которое входят зарегистрированные пользователи). Пользователи, голосующие за домены, в свою очередь, сами имеют репутацию, при этом оценка сообщества включает в себя все голоса, с учетом репутациеи пользователей, которые проголосовали за тот или иной домен. Отрицательные (красные) оценки указывают на злонамеренность, в то время как положительные (зеленые) оценки отражают безвредность. Чем больше абсолютное число, тем больше вы можете доверять данной оценке. Вы можете прочитать больше об этом в этой статье сообщества.

Объект типа “IP-addresses”

```
{
  "data": {
    "type": "ip_address"
    "id": "<ipv4>",
    "links": {
      "self": "https://virustotal.com/api/v3/ip_addresses/<ipv4>"
    },
    "attributes": {
      "as_owner": "<string>",
      "asn": <int>,
      "continent": "<string>",
      "country": "<string>",
      "network": "<ipv4_range>",
      "regional_internet_registry": "<string>",
      "reputation": <int>,
      "total_votes": {
        "harmless": <int>,
        "malicious": <int>
      }
    }
  }
}
```

3.5 URL (URLs)

Информация об URL-адресах.

URL-адреса не только представляют информацию сами по себе, но и могут давать контекстную информацию о файлах и других элементах на VirusTotal.

Различные вызовы URL-адресов могут возвращать различные объекты, связанные с URL-адресами:

- data - корневая структура отчета:
  - categories - категория;
  - first_submission_date - дата первого представления этого URL-адреса в VirusTotal;
  - last_analysis_date - время последнего сканирования URL-адреса;
  - last_analysis_results - результат сканирования URL-адресов. Словарь с именем сканера в качестве ключа и словарь с примечаниями / результатом сканирования в качестве значения:
    - category - нормализованный результат сканирования:
      - "harmless" - сайт не является вредоносным;
      - "undetected" - сканер не имеет никакого мнения об этом сайте;
      - "suspicious" - сканер считает сайт подозрительным;
      - "malicious" - сканер считает сайт вредоносным;
    - engine_name - полное наименование сервиса, сканировавшего URL (имя антивирусного “движка”);
    - engine_update - значение обновления антивирусного “движка”, в случае, если эти данные доступны;
    - engine_version - версия антивирусного “движка”, в случае, если эти данные доступны;
    - method - способ анализа URL, предоставляемого сервисом (например "blacklist");
    - result - необработанное значение, возвращаемое сканером URL-адресов ("clean", "malicious", "suspicious", "phishing"). Данное значение может варьироваться от сканера к сканеру, поэтому для нормализации требуется поле "category";

  last_analysis_stats - общее количество результатов сканирования этого URL-адреса;
  - harmless - количество сообщений о безвредности URL-адреса;
  - malicious - количество сообщений о вредоносности URL-адреса;
  - suspicious - количество сообщений о подозрительности URL-адреса;
  - timeout - количество таймаутов при сканировании URL-адреса;
  - undetected - количество сообщений о необнаружении каких-либо признаков вредоносности URL-адреса;
  - last_final_url - окончание перенаправления исходного URL (при перенаправлении);
  - last_http_response_code - HTTP код последнего ответа;
  - last_http_response_content_length - длина полученного содержимого (в байтах);
  - last_http_response_content_sha256 - SHA256 хэш полученного контента;

  - last_http_response_headers - словарь из заголовков и их значений последнего HTTP-ответа;
  - last_modification_date - дата последней модификации;
  - last_submission_date - время последней отправки URL-адреса на анализ;
  - reputation - значение голосов от сообщества VirusTotal;
  - tags - тэги;
  - times_submitted - количество проверок URL-адреса;
  - total_votes - словарь с количеством положительных ("harmless") и отрицательных ("malicious") голосов, полученных от сообщества VirusTotal;
  - url - исходный URL для сканирования;
- id - идентификатор для этого конкретного отчета об URL-адресе;
- links - содержит "self", со ссылкой на сам отчет;
- type - значение - "url", тип этого ответа.

Объект типа “URL”

```
{
  "data": {
    "attributes": {
      "categories": {dict},
      "first_submission_date": <int:timestamp>,
      "last_analysis_date": <int:timestamp>,
      "last_analysis_results": {
        "<str:scanner name>": {
          "category": "<string>",
          "engine_name": "<string>",
          "engine_update": null,
          "engine_version": null,
          "method": "<string>",
          "result": "<string>"
          }, ...
        },
      "last_analysis_stats": {
        "harmless": <int>,
        "malicious": <int>,
        "suspicious": <int>,
        "timeout": <int>,
        "undetected": <int>
        },
      "last_final_url": "<string>",
      "last_http_response_code": <int>,
      "last_http_response_content_length": <int>,
      "last_http_response_content_sha256": "<string>",
      "last_http_response_headers": {"<string>": "<string>", ... },
      "last_modification_date": <int:timestamp>,
      "last_submission_date": <int:timestamp>,
      "reputation": <int>,
      "tags": [<strings>],
      "times_submitted": <int>,
      "total_votes": {"harmless": <int>, "malicious": <int>},
```

(continues on next page)

(continued from previous page)

```
    "url": "<string>"
    },
  "id": "<string>",
  "links": {"self": "<string>"}
  "type": "url"
 }
}
```

3.6 Комментарии (comments)

Комментарии, размещенные сообществом о файлах, URL-адресах, IP-адресах, доменах и графиках.

Пользователи сообщества VirusTotal могут добавить информацию в отчет объекта, добавив комментарий. Детали комментария:

- attributes:
  - date - дата публикации комментария в формате UTC;
  - html - необработанный HTML-текст комментария;
  - tags - тэг комментария (размещенный в тексте комментария с использованием #);
  - text - текст комментария;
  - votes - количество голосов по категориям (abuse, negative, positive);
- id - идентификатор комментария;
- links - содержит "self", со ссылкой на сам отчет;
- type - тип ответа (значение "comment");
- relationships - по умолчанию не возвращается. Должен быть запрошен с помощью параметра запроса relationships и типа отношения:
  - author - информация о пользователе, опубликовавшего комментарий:
    - data:
      - id - идентификатор пользователя;
      - type - тип пользователя (значение user);
    - links:
      - self - ссылка на автора комментария;
      - related - ссылка на отношение комментарий-автор;
  - item - информация об элементе, о котором был размещен комментарий:
    - data:
      - id - идентификатор элемента;
      - type - тип элемента, может быть file, url, ip_address, domain или graph;
      - links:
      - self - ссылка на комментируемый элемент;
      - related - ссылка на отношение комментарий-элемент.

Объект типа “comment”

```
{
  "attributes": {
    "date": <int:timestamp>,
    "html": "<string>",
    "tags": [<strings>],
    "text": "<string>",
    "votes" {
      "abuse": <int>,
      "negative": <int>,
      "positive": <int>
    }
  },
  "id": "<string>",
  "links": {
    "self": "<string>"
  },
  "type": "<string>",
  "relationships":
    "author": {
      "data": {
        "id": "<string>",
        "type": "<string>"
      },
      "links": {
        "related": "<string>",
        "self": "<string>"
      }
    },
  "item": {
    "data": {
      "id": "<string>",
      "type": "<string>"
    },
    "links": {
      "related": "<string>",
      "self": "<string>"
    }
  }
}
```

3.7 Представления (submissions)

Информация о представлениях.

- attributes - содержит "date", с датой, когда был представлен ресурс;
- id - идентификатор представленного ресурса;
- links - содержит "self", со ссылкой на сам отчет;
- type - значение "submission", то есть тип объекта.

Объект типа “submission”

```
{
  "attributes": {"date": <int:timestamp>},
  "id": "<string>",
  "links": {"self": "<string>"},
  "type": "submission"
}
```

3.8 Скриншоты (screenshots)

Скриншоты - это снимки экрана, полученные во время выполнения файла в изолированной машине анализа поведения (“песочнице”). Этот объект содержит атрибуты, определяющие, где и когда был создан снимок экрана:

- sandbox_name - наименование песочницы, в которой был выполнен файл;
- date - время создания скриншота (как метка времени Unix);
- link - URL-адрес, указывающий на изображение;
- analysed_file_sha256 - отношение, указывающее на файловый объект, который был выполнен.

JSON

```
{
  "data": {
    "type": "screenshot",
    "id": "<SCREENSHOT_NAME>",
    "attributes" : {
      "sandbox_name": "<string>",
      "date": "<unix_timestamp>",
      "link": "<string>",
      "analysed_file_sha256": <object>
    }
  }
}
```

3.9 Голоса (votes)

- attributes - данные о конкретном голосовании:
  - date - дата окончания голосования;
  - value - вес, который дает этот голос (положительный или отрицательный) для Community Score;
  - verdict - "malicious" или "harmless";
- id - идентификатор ресурса, по которому проводилось голосование;
- links - содержит "self", со ссылкой на само голосование;
- type - значение "vote", то есть тип объекта.

Объект типа “vote”

```
{
  "attributes": {"date": <int:timestamp>,
                 "value": <int>,
                 "verdict": "<string>"},
  "id": "<string>",
  "links": {"self": "<string>"},
  "type": "vote"
}
```

ГЛАВА 4

Основные функции VirusTotal API

4.1 Files (Функции для работы с файлами)

Файлы являются одним из наиболее важных типов объектов в API VirusTotal. У нас есть огромный набор данных из более чем 2 миллиардов файлов, которые были проанализированы VirusTotal на протяжении многих лет. В этом разделе описываются функции API для анализа новых файлов и получения информации о любом файле в нашем наборе данных.

4.1.1 POST /files

Загрузка и анализ файла.

https://www.virustotal.com/api/v3/files

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/files \
  --header 'x-apikey: <your API key>' \
  --form file=@/path/to/file
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files"
headers = {"x-apikey" : "<ключ доступа к API>"}
with open("<путь к файлу>", "rb") as file:
```

(continues on next page)

(continued from previous page)

```
files = {"file": ("<путь к файлу>", file)}
response = requests.post(api_url, headers=headers, files=files)
```

Параметры запроса

- file - файл для сканирования.

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Файлы могут быть загружены в VirusTotal путем отправки POST-запросов, закодированных как multipart/form-data, в функцию https://www.virustotal.com/api/v3/files. Каждый POST-запрос должен иметь поле с именем file, содержащее файл для анализа. Общий размер полезной нагрузки не может превышать 32 МБ. Для загрузки больших файлов см. GET /files/upload_url.

Результат, возвращаемый этой функцией, является дескриптором объекта для нового анализа. Идентификатор, содержащийся в дескрипторе, можно использовать с функцией GET /analyses/{id} для получения информации о результатах анализа этого файла.

Для анализа файла, который ранее уже был загружен в VirusTotal, можно использовать POST /files/{id}/analyse.

Пример ответа

```
{
  "data": {
    "type": "analysis",
    "id": "NjY0MjRlOTFjMDIyYTkyNWM0NjU2NWQzYWNlMzFmZmI6MTQ3NTA0ODI3Nw=="
  }
}
```

4.1.2 GET /files/upload_url

Получение URL для загрузки файла больше 32 МБ.

GET https://www.virustotal.com/api/v3/files/upload_url

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/upload_url \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/upload_url"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Для загрузки файлов размером менее 32 МБ вы можете просто использовать функцию POST /files, но для файлов большего размера необъодимо сначала получить специальный URL загрузки, а затем отправить POST-запрос на этот URL. Этот POST-запрос должен иметь тот же формат, что и для функции POST /files. Каждый полученный URL можно использовать только один раз.

Note: Файлы размером более 200 МБ. Обратите внимание, что файлы размером более 200 МБ, как правило, представляют собой пакеты какого-либо вида (сжатые файлы, ISO-образы и т. д.) в этих случаях имеет смысл загрузить внутренние файлы отдельно по нескольким причинам:

- Движки некоторых антивирусов, как правило, имеют проблемы с производительностью при сканировании больших файлов (из-за больших тайм-аутов, некоторые из них могут даже не сканировать их);
- Движки некоторых антивирусов не могут проверять определенные типы файлов, в то время как они смогут проверить внутренние файлы, если они будут отправлены;
- При сканировании большого пакета вы теряете контекст, в котором конкретный внутренний файл вызывает обнаружение.

Пример ответа

```
{
  "data": "http://www.virustotal.com/_ah/upload/AMmfu6b-_DXUeFe36Sb3b0F4B8mH9Nb-CHbRoUNVOPwG/
↪"
}
```

4.1.3 GET /files/{id}

Получение информации о файле.

GET https://www.virustotal.com/api/v3/files/{id}

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id} \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Пример ответа

```
{
  "type": "file",
  "id": "8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85",
  "links": {
    "self": "https://www.virustotal.com/api/v3/files/
↪8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85"
  },
  "data": {
    "attributes": {
      "first_seen_itw_date": 1075654056,
      "first_submission_date": 1170892383,
      "last_analysis_date": 1502355193,
      "last_analysis_results": {
        "AVG": {
          "category": "undetected",
          "engine_name": "AVG",
          "engine_update": "20170810",
          "engine_version": "8.0.1489.320",
          "method": "blacklist",
          "result": null
        }
          ...
      },
      "last_analysis_stats": {
        "harmless": 0,
        "malicious": 0,
        "suspicious": 0,
        "timeout": 0,
        "type-unsupported": 8,
        "undetected": 59
      },
      "last_submission_date": 1502355193,
      "magic": "data",
      "md5": "76cdb2bad9582d23c1f6f4d868218d6c",
      "names": [
```

(continues on next page)

(continued from previous page)


```
      "zipnew.dat",
      "327916-1502345099.zip",
      "ac3plug.zip",
      "IMG_6937.zip",
      "DOC952.zip",
      "20170801486960.zip"
    ],
    "nsrl_info": {
      "filenames": [
        "WINDOWS DIALUP.ZIP",
        "kemsetup.ZIP",
        "Data_Linux.zip",
        "2003.zip",
        "_6A271FB199E041FC82F4D282E68B01D6"
      ],
      "products": [
        "Master Hacker Internet Terrorism (Core Publishing Inc.)",
        "Read Rabbits Math Ages 6-9 (Smart Saver)",
        "Neverwinter Nights Gold (Atari)",
        "Limited Edition Print Workshop 2004 (ValuSoft)",
        "Crysis (Electronic Arts Inc.)"
      ]
    },
    "reputation": -889,
    "sha1": "b04f3ee8f5e43fa3b162981b50bb72fe1acabb33",
    "sha256": "8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85",
    "size": 22,
    "ssdeep": "3:pjt/l:Nt",
    "tags": [
      "software-collection",
      "nsrl",
      "attachment",
      "trusted",
      "via-tor"
    ],
    "times_submitted": 26471,
    "total_votes": {
      "harmless": 639,
      "malicious": 958
    },
    "trid": [
      {
        "file_type": "ZIP compressed archive (empty)",
        "probability": 100
      }
    ],
    "trusted_verdict": {
      "filename": "lprn_spotlightstory_015.zip",
      "link": "https://dl.google.com/dl/spotlight/test/lprn_spotlightstory/9/lprn_spotlightstory_015.zip",
      "organization": "Google",
      "verdict": "goodware"
    },
    "type_description": "unknown",
    }
  }
 }
}
```

4.1.4 POST /files/{id}/analyse

Повторный анализ файла в VirusTotal/ .. warning:: Эта функция API может привести к отказу в обслуживании инфраструктуры сканирования в случае неправильного использования. Пожалуйста, свяжитесь с нами, если вы собираетесь сканировать более 50 тысяч файлов в день.

https://www.virustotal.com/api/v3/files/{id}/analyse

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/files/{id}/analyse \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/analyse"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.post(api_url, headers=headers)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Файлы, которые уже были загружены в VirusTotal, можно повторно проанализировать, не загружая их снова, используя эту функцию. Ответом является дескриптор объекта для нового анализа, как и в функции POST /files. Идентификатор, содержащийся в дескрипторе, можно использовать с функцией GET /analyses/{id} для получения информации о результатах анализа.

Пример ответа

```
{
  "data": {
    "type": "analysis",
    "id": "NjY0MjRlOTFjMDIyYTkyNWM0NjU2NWQzYWNlMzFmZmI6MTQ3NTA0ODI3Nw=="
  }
}
```

4.1.5 GET /files/{id}/comments

Получение комментариев для файла

https://www.virustotal.com/api/v3/files/{id}/comments

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id}/comments \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

4.1.6 POST /files/{id}/comments

Добавление комментария для файла.

POST https://www.virustotal.com/api/v3/files/{id}/comments

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/files/{id}/comments \
  --header 'x-apikey: <your API key>' \
  --data '{"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
comments = {"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}
response = requests.post(api_url, headers=headers, json=comments)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string);
- data - комментарий (json).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

С помощью этой функции вы можете опубликовать комментарий для данного файла. Тело POST-запроса должно быть JSON-представлением комментария. Обратите внимание, что вам не нужно указывать идентификатор объекта, так как он автоматически генерируется для новых комментариев.

Любое слово, начинающееся с # в тексте вашего комментария, будет считаться тегом и добавляться в атрибут тега комментария.

Пример запроса

```
{
  "data": {
    "type": "comment",
    "attributes": {
       "text": "Lorem #ipsum dolor sit ..."
    }
  }
}
```

Пример ответа

```
{
  "data": {
    "type": "comment",
    "id": "<comment's ID>",
    "links": {
      "self": "https://www.virustotal.com/api/v3/comments/<comment's ID>"
    },
    "attributes": {
      "date": 1521725475,
      "tags": ["ipsum"],
      "html": "Lorem #ipsum dolor sit ...",
      "text": "Lorem #ipsum dolor sit ...",
      "votes": {
        "abuse": 0,
        "negative": 0,
        "positive": 0
      }
    }
  }
}
```

4.1.7 GET /files/{id}/votes

Получение результатов голосования для файла

GET https://www.virustotal.com/api/v3/files/{id}/votes

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id}/votes \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

4.1.8 POST /files/{id}/votes

Добавление голоса для файла.

POST https://www.virustotal.com/api/v3/files/{id}/comments

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/files/{id}/votes \
  --header 'x-apikey: <your API key>' \
  --data '{"data": {"type": "vote", "attributes": {"verdict": "malicious"}}}''
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/votes"
headers = {"x-apikey" : "<ключ доступа к API>"}
votes = {"data": {"type": "vote", "attributes": {"verdict": "malicious"}}}
response = requests.post(api_url, headers=headers, json=votes)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string);
- data - голос (json).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

С помощью этой функции вы можете опубликовать свой голос за данный файл. Тело для запроса POST должно быть JSON-представлением объекта голосования. Обратите внимание, однако, что вам не нужно указывать идентификатор объекта, так как они автоматически генерируются для новых голосов.

Атрибут verdict должен быть либо harmless, либо malicious.

Пример ответа

```
{
  "data": {
    "type": "vote",
    "attributes": {
        "verdict": "harmless"
    }
  }
}
```

4.1.9 GET /files/{id}/download_url

Получение URL для загрузки файла.

Note: Требуются особые привилегии. Эта функция доступна только для пользователей со специальными привилегиями.

GET https://www.virustotal.com/api/v3/files/{id}/download_url

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id}/download_url \
 --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/download_url"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Эта функция возвращает подписанный URL, с которого можно загрузить указанный файл. Получение URL считается загрузкой файла в квоте, даже если вы на самом деле не загружаете файл. URL можно использовать для загрузки файла несколько раз, не потребляя никакой квоты. Срок действия URL истекает через 1 час.

Пример ответа

```
{
  "data": "https://vtsamples.commondatastorage.googleapis.com/275a..fd0f?GoogleAccessId=758681729565-
↪rc7fcckv235v1@developer.gserviceaccount.com&Expires=1524733537&Signature=GRs9WLy...oHA%3D"
}
```

4.1.10 GET /files/{id}/download

Загрузка файла.

Note: Требуются особые привилегии. Эта функция доступна только для пользователей со специальными привилегиями.

GET https://www.virustotal.com/api/v3/files/{id}/download

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id}/download \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/download"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Эта функция похожа на GET /files/{id}/download_url, но она перенаправляет вас на URL загрузки файла. URL загрузки, на который вы перенаправлены, может быть использован повторно столько раз, сколько вы хотите в течение 1 часа. После этого срока действие URL истекает и он больше не может быть использован.

4.1.11 GET /files/{id}/{relationship}

Получение объектов, связанных с файлом.

https://www.virustotal.com/api/v3/files/{id}/{relationship}

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/files/{id}/{relationship} \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/files/{id}/{relationship}"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - SHA-256, SHA-1 или MD5 идентификатор файла (string);
- relationship - наименование отношения (см. таблицу ниже);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Объекты типа file имеют ряд отношений с другими файлами и объектами. Как уже упоминалось в разделе “Отношения”, эти связанные объекты можно получить, отправив GET-запросы на URL, соответствующий нужному отношению.

Некоторые отношения доступны только тем пользователям, которые имеют доступ к VirusTotal Intelligence.

Отношения, поддерживаемые объектами файла:

Отношения	Описание	Доступность
analyses	Объект analyses для файла	Только для пользователей Intelligence
behaviours	Отчеты о поведении для файла	Все пользователи
bundled_files	Файлы, собранные в одном файле	Все пользователи
carbonblack_children	Файлы, полученные из файла Carbon Black	Только для пользователей Intelligence
carbonblack_parents	Файлы Carbon Black, из которых был получен файл	Только для пользователей Intelligence
comments	Комментарии к файлу	Все пользователи
compressed_parents	Сжатые файлы, содержащие этот файл	Все пользователи
contacted_domains	Домены, с которыми связан файл	Все пользователи
contacted_ips	IP-адреса, с которыми связан файл	Все пользователи
contacted_urls	URL, с которыми связан файл	Все пользователи
email_parents	Файлы электронной почты, содержащие этот файл	Только для пользователей Intelligence
embedded_domains	Имена доменов, содержащиеся в файле	Только для пользователей Intelligence
embedded_ips	IP-адреса, содержащиеся в файле	Только для пользователей Intelligence
execution_parents	Файлы, которые запустили файл	Все пользователи
graphs	Графики, включающие файл	Все пользователи
itw_urls	URL “in the wild”, откуда был загружен файл	Все пользователи
overlay_parents	Файлы, содержащие файл в виде оверлея	Все пользователи
pcap_parents	Файлы PCAP, содержащие этот файл	Все пользователи
pe_resource_parents	PE-файлы, содержащие файл в качестве ресурса	Все пользователи
similar_files	Файлы, похожие на данный файл	Только для пользователей Intelligence
submissions	Представления файла	Только для пользователей Intelligence
screenshots	Скриншоты, связанные с песочницей, в которой выполнялся файл	Все пользователи
votes	Результаты голосования для файла	Все пользователи

4.1.12 GET /file_behaviours/{sandbox_id}/pcap

GET https://www.virustotal.com/api/v3/file_behaviours/{sandbox_id}/pcap

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/file_behaviours/{sandbox_id}/pcap \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/file_behaviours/{sandbox_id}/pcap"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- sandbox_id - идентификатор, полученный из функции GET /files/{id}/{relationship}, с параметром relationship равным behaviours.

Заголовок запроса

- x-apikey - ключ доступа к API (string).

4.2 URLs (Функции для работы с URL-адресами)

VirusTotal анализирует не только файлы, но и URL-адреса. В этом разделе описаны функции API для анализа URL-адресов и получения информации о них.

4.2.1 Идентификатор URL-адреса

Всякий раз, когда мы говорим об идентификаторе URL-адреса в этой документации, мы имеем в виду последовательность символов, которые однозначно идентифицируют конкретный URL. Эти идентификаторы могут принимать две формы:

- SHA-256 хэш от строки канонического URL-адреса;
- Строка, полученная в результате кодирования URL-адреса в base64 (без заполнения символами "=").

Все идентификаторы URL-адресов, возвращаемые API VirusTotal, находятся в первой форме, и если у вас есть один из этих идентификаторов, вы можете использовать его в последующих вызовах API, которым требуется идентификатор URL-адреса. Однако создание таких идентификаторов самостоятельно может быть затруднено из-за алгоритма канонизации, который должен быть применен к URL-адресу перед вычислением SHA-256 хэша. Канонизация гарантирует, что два URL-адреса, отличающиеся только незначительными аспектами, например некоторыми экранированными символами, имеют один и тот же идентификатор. По этой причине мы предлагаем возможность идентификации URL-адреса путем кодирования его в base64 и использования результирующей строки в качестве идентификатора. В таких случаях URL-адрес не нужно канонизировать, это делается на стороне сервера VirusTotal.

Обратите внимание, что мы используем неупакованную кодировку base64, как определено в разделе 3.2 RFC 4648, что означает, что полученные идентификаторы URL-адресов не должны быть дополнены символами "=", как это обычно происходит с данными, закодированными в base64.

Вот один из примеров того, как сгенерировать идентификатор URL-адреса:

```
import base64
  ...
url_id = base64.urlsafe_b64encode("<строка с url-адресом>").strip("=")
```

4.2.2 POST /urls

Анализ URL-адреса.

POST https://www.virustotal.com/api/v3/urls

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/urls \
  --header 'x-apikey: <your API key>' \
  --form url='<url>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls"
    headers = {"x-apikey" : "<ключ доступа к API>"}
    data = {'url': url}
response = requests.post(api_url, headers=headers, data=data)
```

Параметры запроса

- url - URL-адрес, который должен быть проанализирован.

Заголовок запроса

- x-apikey - ключ доступа к API (string).

URL-адреса могут быть отправлены в VirusTotal путем отправки POST-запросов. Каждый POST-запрос должен иметь поле с именем url, содержащие URL-адрес, который должен быть проанализирован.

Результатом, возвращаемым этой функцией, является дескриптор объекта для нового анализа. Идентификатор, содержащийся в дескрипторе, можно использовать с функцией GET /analyses/{id} для получения информации о результатах анализа.

Для анализа URL-адреса, ранее отправленного в VirusTotal, можно использовать POST /urls/{id}/analyse.

- id - идентификатор для последующего использования с другими вызовами;
- type - значение analysis.

Структура ответа

```
{
  "data": {"id": "<string>", "type": "analysis"}
}
```

4.2.3 GET /urls/{id}

Получение информации об URL-адресе.

GET https://www.virustotal.com/api/v3/urls/{id}

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/urls/{id} \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls{id}"
    headers = {"x-apikey" : "<ключ доступа к API>"}
    response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - идентификатор URL-адреса.

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе “Идентификатор URL-адреса”.

Структура ответа

```
{
  "data": <URL OBJECT>
}
```

4.2.4 POST /urls/{id}/analyse

Анализ URL-адреса.

POST https://www.virustotal.com/api/v3/urls/{id}/analyse

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/urls/{id}/analyse \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/analyse"
    headers = {"x-apikey" : "<ключ доступа к API>"}
    response = requests.post(api_url, headers=headers)
```

Параметры запроса

- id - идентификатор URL-адреса.

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе "Идентификатор URL-адреса".

Структура ответа

```
{
  "data": {"id": "<string>", "type": "analysis"}
}
```

4.2.5 GET /urls/{id}/comments

Получение комментариев для URL

GET https://www.virustotal.com/api/v3/urls/{id}/comments

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/urls/{id}/comments \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- id - идентификатор URL (string);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе “Идентификатор URL-адреса”.

- data - список объектов типа “комментарий” (comments);
- links - содержит "self" со ссылкой на эту группу комментариев и "next", со ссылкой на следующую группу;
- cursor - содержит символ курсора, используемый для доступа к следующей группе комментариев.

Структура ответа

```
{
  "data": [<COMMENT OBJECTS>],
  "links": {"next": "<string>", "self": "<string>"},
  "meta": {"cursor": "<string>"},
}
```

4.2.6 POST /files/{id}/comments

Добавление комментария для URL-адреса.

POST https://www.virustotal.com/api/v3/urls/{id}/comments

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/urls/{id}/comments \
  --header 'x-apikey: <your API key>' \
  --data '{"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
comments = {"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}
response = requests.post(api_url, headers=headers, json=comments)
```

Параметры запроса

- id - идентификатор URL (string);
- data - комментарий (json).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

С помощью этой функции вы можете опубликовать комментарий для данного URL-адреса. Тело POST-запроса должно быть JSON-представлением комментария. Обратите внимание, что вам не нужно указывать идентификатор объекта, так как он автоматически генерируется для новых комментариев.

Любое слово, начинающееся с # в тексте вашего комментария, будет считаться тегом и добавляться в атрибут тега комментария.

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе "Идентификатор URL-адреса".

Структура ответа

```
{
  "data": <COMMENT OBJECT>
}
```

4.2.7 GET /urls/{id}/votes

Получение результатов голосования для URL-адреса.

GET https://www.virustotal.com/api/v3/urls/{id}/votes

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/urls/{id}/votes \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/votes"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- id - идентификатор URL (string);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе “Идентификатор URL-адреса”.

- data - список объектов типа “голос” (votes);
- links - содержит "self ", со ссылкой на саму группу голосов.

Структура ответа

```
{
  "data": [<VOTE OBJECTS>],
  "links": {"self": "<string>"}
}
```

4.2.8 POST /urls/{id}/votes

Добавление голоса для URL-адреса.

POST https://www.virustotal.com/api/v3/urls/{id}/votes

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/urls/{id}/votes \
  --header 'x-apikey: <your API key>' \
  --data '{"data": {"type": "vote", "attributes": {"verdict": "malicious"}}}'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/votes"
headers = {"x-apikey" : "<ключ доступа к API>"}
votes = {"data": {"type": "vote", "attributes": {"verdict": "malicious"}}}
response = requests.post(api_url, headers=headers, json=votes)
```

Параметры запроса

- id - идентификатор URL (string);
- data - голос (json).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе “Идентификатор URL-адреса”.

С помощью этой функции вы можете опубликовать свой голос за данный URL-адрес. Тело для запроса POST должно быть JSON-представлением объекта голосования. Обратите внимание, однако, что вам не нужно указывать идентификатор объекта, так как они автоматически генерируются для новых голосов.

Атрибут verdict должен быть либо harmless, либо malicious.

Пример ответа

```
{
  "data": <VOTE OBJECT>
}
```

4.2.9 GET /urls/{id}/network_location

Получение домена или IP-адреса для URL-адреса.

GET https://www.virustotal.com/api/v3/urls/{id}/network_location

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/urls/{id}/network_location \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/network_location"
headers = {"x-apikey" : "<ключ доступа к API>"}
response = requests.get(api_url, headers=headers)
```

Параметры запроса

- id - идентификатор URL (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе "Идентификатор URL-адреса".

- data - водержит объект типа "домен" (domain) или объект типа "IP-адрес" (IP addresses), в зависимости от идентификатора запроса;
- links - содержит "self", со ссылкой на отчет о конкретном местоположении.

Пример ответа

```
{
  "data": <DOMAIN OBJECT> or <IP OBJECT>,
  "links": {"self": "<string>"}
}
```

4.2.10 GET /urls/{id}/{relationship}

Получение объектов, связанных с URL-адресом.

GET https://www.virustotal.com/api/v3/urls/{id}/{relationship}

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/urls/{id}/{relationship} \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/urls/{id}/{relationship}"
headers = {"x-apikey" : "<ключ доступа к API>"}
    query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- id - идентификатор URL (string).
- relationship - наименование отношения (см. таблицу ниже);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Hint: Дополнительные сведения о создании допустимого идентификатора URL-адреса см. в разделе “Идентификатор URL-адреса”.

Объекты URL имеют ряд связей с другими URL-адресами и объектами. Как уже упоминалось в разделе “Отношения”, эти связанные объекты можно получить, отправив GET-запросы на соответствующий URL-адрес.

Некоторые отношения доступны только пользователям, имеющим доступ к VirusTotal Intelligence.

Отношения, поддерживаемые объектами URL:

Отношения	Описание	Доступность
analyses	Анализ URL-адреса	Только для пользователей Intelligence
downloaded_files	Файлы, загруженные с URL-адреса	Только для пользователей Intelligence
graphs	Графики, включающие URL-адрес	Все пользователи
last_serving_ip_address	Последний IP-адрес, который обслуживал URL	Все пользователи
redirecting_urls	URL-адреса, которые перенаправляются на данный URL-адрес	Только для пользователей Intelligence
submissions	Представления для URL-адреса	Только для пользователей Intelligence

- analyses - возвращает объект типа “url”;
- downloaded_files - возвращает список объектов типа “file”;
- graphs - возвращает объект типа “graph”;
- last_serving_ip_address - возвращает объект типа “ip”;
- redirecting_urls - возвращает список объектов типа “url”;
- submissions - возвращает список объектов типа “submission”.

4.3 Domains (Функции для работы с доменами)

4.3.1 GET /domains/{domain}

Получение информации об Internet-домене.

https://www.virustotal.com/api/v3/domains/{domain}

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/domains/{domain} \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/domains/{domain}"
    headers = {"x-apikey" : "<ключ доступа к API>"}
    response = requests.get(api_url, headers=headers)
```

Параметры запроса

- domain - имя домена (string).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

Пример ответа

```
{
  "data": {
    "type": "domain",
    "id": "virustotal.com",
    "links": {
```

(continues on next page)

(continued from previous page)


```
    "self": "https://virustotal.com/api/v3/domains/virustotal.com"
   },
   "attributes": {
    "categories": {
     "Alexa": "services",
     "BitDefender": "computersandsoftware",
     "TrendMicro": "computers internet",
     "Websense ThreatSeeker": "computer security"
    },
    "creation_date": 1032308169,
    "last_update_date": 1389199030,
    "registrar": "MarkMonitor Inc.",
    "reputation": 13,
    "total_votes": {
     "harmless": 2,
     "malicious": 0
    },
     "whois": "Domain Name: VIRUSTOTAL.COM\r\n   Registry Domain ID: ...",
    "whois_date": 1560599498
   }
  }
}
```


4.3.2 GET /domains/{domain}/comments

Получение комментариев для Internet-домена.

https://www.virustotal.com/api/v3/domains/{domain}/comments

cURL

```
curl --request GET \
  --url https://www.virustotal.com/api/v3/domains/{domain}/comments \
  --header 'x-apikey: <your API key>'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/domains/{domain}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
query = {"limit": "<limit)>", "cursor": "<cursor>"}
response = requests.get(api_url, headers=headers, params=query)
```

Параметры запроса

- domain - имя домена (string);
- limit - максимальное число комментариев в ответе (int_32, необязательный параметр);
- cursor - курсор продолжения (string, необязательный параметр).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

4.3.3 POST /domains/{domain}/comments

Добавление комментария для Internet-домена.

https://www.virustotal.com/api/v3/domains/{domain}/comments

cURL

```
curl --request POST \
  --url https://www.virustotal.com/api/v3/domains/{domain}/comments \
  --header 'x-apikey: <your API key>' \
  --data '{"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}'
```

Python

```
import requests
    ...
api_url = "https://www.virustotal.com/api/v3/domains/{domain}/comments"
headers = {"x-apikey" : "<ключ доступа к API>"}
comments = {"data": {"type": "comment", "attributes": {"text": "Lorem ipsum dolor sit ..."}}}
response = requests.post(api_url, headers=headers, json=comments)
```

Параметры запроса

- domain - имя домена (string);
- data - комментарий (json).

Заголовок запроса

- x-apikey - ключ доступа к API (string).

С помощью этой функции вы можете опубликовать комментарий для данного домена. Тело POST-запроса должно быть JSON-представлением объекта “comment”. Обратите внимание, однако, что вам не нужно указывать идентификатор для объекта, так как они автоматически генерируются для новых комментариев.

Любое слово, начинающееся с # в тексте вашего комментария, будет считаться тегом и будет добавлено в атрибут тега комментария.

Пример запроса

```
{
  "data": {
    "type": "comment",
    "attributes": {
       "text": "Lorem #ipsum dolor sit ..."
    }
  }
}
```

Пример ответа

```
{
  "data": {
    "type": "comment",
    "id": "<comment's ID>",
    "links": {
      "self": "https://www.virustotal.com/api/v3/comments/<comment's ID>"
    },
    "attributes": {
      "date": 1521725475,
      "tags": ["ipsum"],
      "html": "Lorem #ipsum dolor sit ...",
      "text": "Lorem #ipsum dolor sit ...",
      "votes": {
        "abuse": 0,
        "negative": 0,
        "positive": 0
      }
    }
  }
}
```

4.3.4 GET /domains/{domain}/{relationship}

Получение объектов, связанных с Internet-доменом.

GET https://www.virustotal.com/api/v3/domains/{domain}/{relationship}

4.3.5 GET /domains/{domain}/votes

GET https://www.virustotal.com/api/v3/domains/{domain}/votes

4.3.6 POST /domains/{domain}/votes

Добавить голос за имя хоста или домена.

POST https://www.virustotal.com/api/v3/domains/{domain}/votes

4.4 IP addresses (Функции для работы с IP-адресами)

4.4.1 GET /ip_addresses/{ip}

Получение информации о IP-адресе.

GET https://www.virustotal.com/api/v3/ip_addresses/{ip}

4.4.2 GET /ip_addresses/{ip}/comments

Получение комментариев для IP-адресе.

GET https://www.virustotal.com/api/v3/ip_addresses/{ip}/comments

4.4.3 POST /ip_addresses/{ip}/comments

Добавление комментария для IP_адреса.

POST https://www.virustotal.com/api/v3/ip_addresses/{ip}/comments \

4.4.4 GET /ip_addresses/{ip}/{relationship}

Получение объектов, связанных с IP-адресом.

GET https://www.virustotal.com/api/v3/ip_addresses/{ip}/{relationship}

4.4.5 POST /ip_addresses/{ip}/votes

Добавление голоса для IP-адреса.

POST www.virustotal.com/api/v3/ip_addresses/{ip}/votes

4.4.6 GET /ip_addresses/{ip}/votes

Получение результатов голосования по IP-адресу.

GET https://www.virustotal.com/api/v3/ip_addresses/{ip}/votes

4.5 Analyses (функции для анализа объектов)

ГЛАВА 5

Функции VT Enterprise

ГЛАВА 6

Функции VT Monitor

- genindex

Index

R

S

T

U

V