

# Table of Contents

[Введение](#Top_of_index_html)

[Обработка файлов снимков сетевого трафика](#Top_of_index1_html)

[Форматы файлов исходных данных pcap, pcapng](#Top_of_chapter1_html)

[Извлечение технологических соединений из данных сетевого трафика](#Top_of_chapter2_html)

[Классификация технологических соединений](#Top_of_chapter3_html)

[Исследование возможности применения методов машинного обучения](#Top_of_index2_html)

[Технология генерации признаковых описаний виртуальных соединений](#Top_of_chapter11_html)

[Подготовка данных для обучения системы классификации](#Top_of_chapter21_html)

[Экспериментальные результаты](#Top_of_chapter31_html)

[Программная реализация системы анализа сетевого трафика](#Top_of_index3_html)

[Архитектура системы](#Top_of_chapter12_html)

[Примеры работы программы](#Top_of_chapter22_html)

[Программный интерфейс управления](#Top_of_chapter32_html)

[Выводы](#Top_of_index4_html)

Содержание

[Введение](#Top_of_index_html) 1.1

[Обработка файлов снимков сетевого трафика](#Top_of_index1_html)  1.2

[Форматы файлов исходных данных pcap, pcapng](#Top_of_chapter1_html) 1.2.1

[Извлечение технологических соединений из данных сетевого трафика](#Top_of_chapter2_html) 1.2.2

[Классификация технологических соединений](#Top_of_chapter3_html) 1.2.3

[Исследование возможности применения методов машинного обучения](#Top_of_index2_html)  1.3

[Технология генерации признаковых описаний виртуальных соединений](#Top_of_chapter11_html) 1.3.1

[Подготовка данных для обучения системы классификации](#Top_of_chapter21_html) 1.3.2

[Экспериментальные результаты](#Top_of_chapter31_html) 1.3.3

[Программная реализация системы анализа сетевого трафика](#Top_of_index3_html) 1.4

[Архитектура системы](#Top_of_chapter12_html) 1.4.1

[Примеры работы программы](#Top_of_chapter22_html) 1.4.2

[Программный интерфейс управления](#Top_of_chapter32_html) 1.4.3

[Выводы](#Top_of_index4_html) 1.5

# Введение

## Описание программного продукта

Becoming a super hero is a fairly straight forward process:

$ give me super-powers

Super-powers are granted randomly so please submit an issue if you're not happy with yours.

Once you're strong enough, save the world:

// Ain't no code for that yet, sorry  
echo 'You got to trust me on this, I saved the world'

# Форматы файлов исходных данных Pcap, PcapNg

Одними из самых распространенных форматов хранения полных данных сетевого трафика являются формат Pcap (англ. Packet CAPture) и его обновленная версия PcapNg (Pcap Next generation). Оба формата представляют собой бинарные файлы, состоящие из глобального заголовка, позволяющего его идентифицировать и записей для каждого захваченного пакета (рис.).

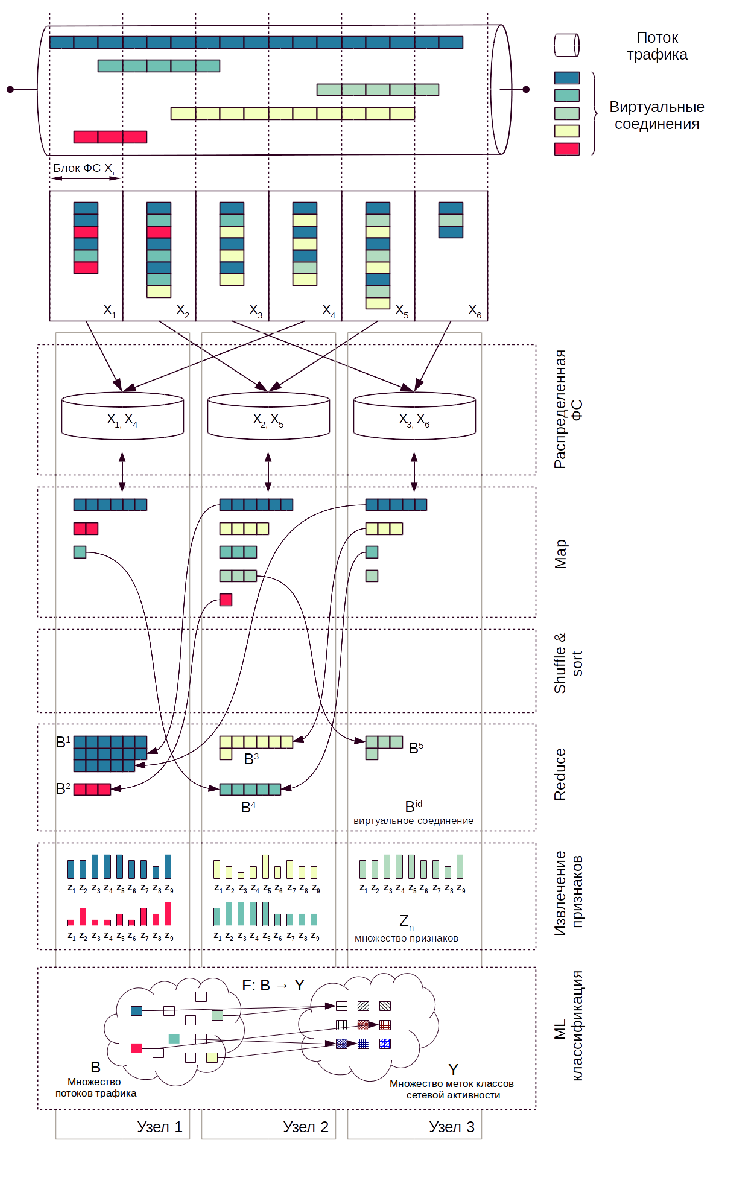
pcap_file

Структура файла снимка сетевого трафика.

Далее будут рассмотрены структура и поля заголоков для обоих типов файлов, и их различия.

## Формат пакетной записи формата Pcap

## Формат пакетной записи формата PcapNg



# Введение

## Описание программного продукта

Becoming a super hero is a fairly straight forward process:

$ give me super-powers

Super-powers are granted randomly so please submit an issue if you're not happy with yours.

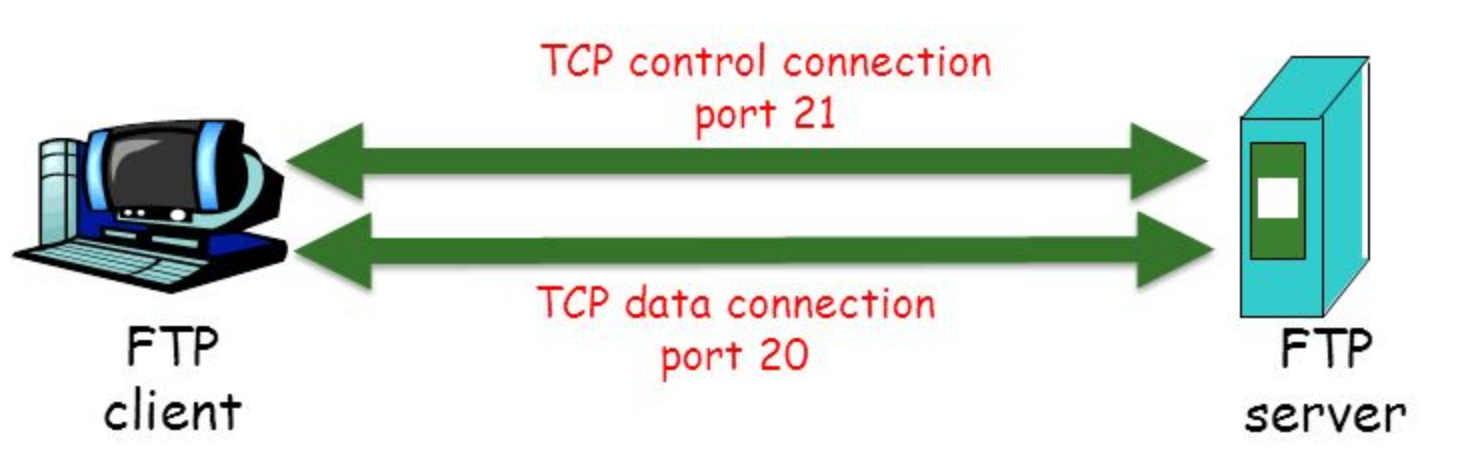
Once you're strong enough, save the world:

// Ain't no code for that yet, sorry  
echo 'You got to trust me on this, I saved the world'

# Технология генерации признаковых описаний виртуальных соединений

Виртуальное соединение (ВС) – это логически упорядоченный обмен сообщениями между узлами сети. Трафик компьютерной сети – это совокупность виртуальных соединений.

Виртуальные соединения классифицируются на два уровня – технологические виртуальные соединения (ТВС) и информационные виртуальные соединения (ИВС).

Для реализации механизмов классификации сетевого трафика, исходные данные пакетов передаваемых данных декомпозируются в форму ИВС и ТВС. ТВС можно определить как потоки пакетов, формируемые сетевыми приложениями в рамках информационного взаимодействия. ТВС может быть представлено в виде счетного подмножества декартова произведения множества пакетов и временных меток. Такое представление характеризуется конечным набором параметров, характеризующих субъект и объект доступа, а также действие в форме потока пакетов между ними в рамках межсетевого взаимодействия. Параметрами в простейшем случае являются идентификаторы субъекта и объекта, такие как адреса, порты, и другие характеристики сетевых протоколов. Для оперативной классификации трафика, наряду с моделью ТВС, используется модель ИВС для описания взаимодействия между объектом и субъектом на уровне прикладных сервисов. Модель ИВС представляет собой совокупность ТВС, используемых одним прикладным приложением в процессе его функфционирования. Примером ИВС как совокупности ТВС является работа с FTP сервером (рис.). Протокол FTP построен на модели отдельных потоков управления и передачи данных между клиентом и сервером. В этом случае можно говорить об одном ИВС, состоящим из двух ТВС. 

Технологические соединения при работе с ftp

## Признаковое описание технологических виртуальных соединений на основе статистических данных потока сетевого трафика

Для

## Признаковое описание информационныъ виртуальных соединений с использованием временных характеристик потока данных сетевого трафика

# Подготовка данных для обучения системы классификации

## Изоляция трафика приложения с ипользованием Linux Namespaces

С появлением таких утилит, как LXC и Docker стало возможным очень легко изолировать процессы в Linux системе в их собственное системное окружение. Это дало возможность запускать целый набор приложений на одной Linux машине и быть уверенным в том, что они не будут мешать друг другу, без необходимости использования виртуальных машин.

Ключевая функциональность ядра, которая позволяет добиться такой изоляции – Linux Namespaces, появилась в Linux начиная с версии 2.4.19 в 2002 году (CLONE\_NEWNS), после чего с последующими обновлениями ядра добавлялись новые:

1. UTS namespaces (CLONE\_NEWUTS, Linux 2.6.19)
2. IPC namespaces (CLONE\_NEWIPC, Linux 2.6.19)
3. PID namespaces (CLONE\_NEWPID, Linux 2.6.24)
4. Network namespaces (CLONE\_NEWNET, Linux 2.6.29)
5. User namespaces (CLONE\_NEWUSER, Linux 3.8)
6. cgroup namespace (CLONE\_NEWCGROUP, Linux 4.0)

Аналогично тому, как chroot позволяет процессам видеть только определённую директорию как корень файловой системы – механизмы Linux namespaces позволяют выполнять операции изоляции в других механизмах операционной системы, такими как дерево процессов, сетевые интерфейсы, точки монтирования, IPC и так далее.

Для изоляции сетевого трафика приложения нас интересуют в первую очередь Network Namespaces. Network namespaces позволяют в рамках одной машины в каждом netns иметь собственные, изолированные от других:

* набор таблиц маршрутизации;
* arp-таблицу;
* правила iptables;
* устройства.

Создаем namespace:

ip netns add R0

В новом изолированном пространстве, отсутвуют сетевые интерфейсы

ip netns exec R0 ip a

1: lo: mtu 65536 qdisc noop state DOWN group default qlen 1000 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

Создаем пару виртуальных интерфейсов:

ip link add veth  
-a  
   
type  
 veth peer name veth-b

Тип создаваемого интерфейса - «Virtual Ethernet»

Перенесем интерфейс veth-a из initial network namespace в namespace R0:

ip link   
set  
 veth  
-a  
 netns   
test

Настроим ip-адреса на созданных виртуальных интерфейсах:

ip netns exec test ifconfig veth-a up 10.0.0.1 netmask 255.255.255.0

ifconfig veth-b up 10.0.0.254 netmask 255.255.255.0

Настроим шлюз по-умолчанию для изолированного интерфейса:

ip netns exec test route add default gw 10.0.0.254 dev veth-a

Активируем ip\_forward

echo 1 > /proc/sys/net/ipv4/ip\_forward

С помощью iptables настроим NAT для интерфейса.

iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o eth0 -j SNAT --to-source 192.168.88.80

Теперь в созданом netspace можно запустить процесс, трафик которого нужно записать, и программу захвата трафика, например wireshark

ip netns exec test Telegram  
ip netns exec test wireshark

## Программа захвата трафика приложения

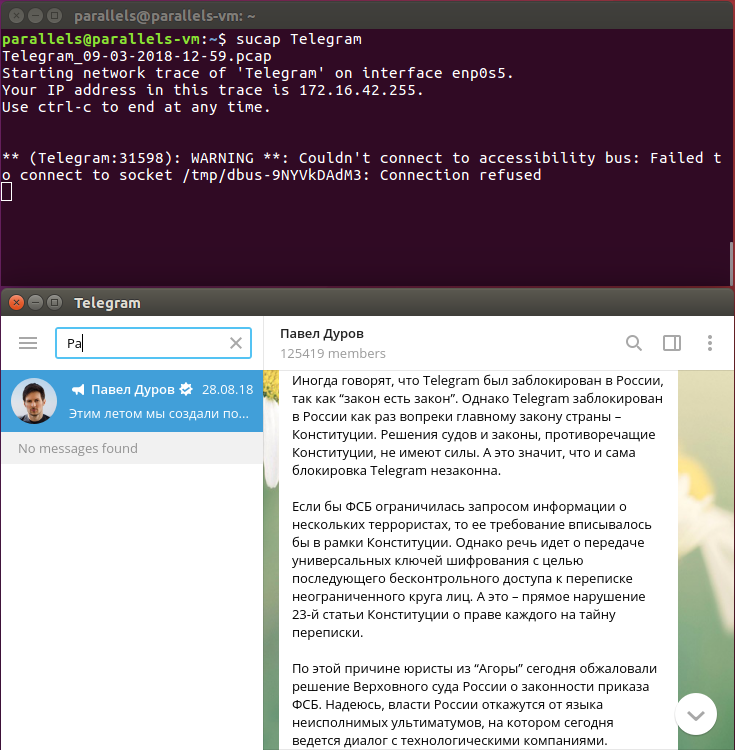


Рис.: Запуск мессенжера телеграм в утилите захвата сетевого трафика

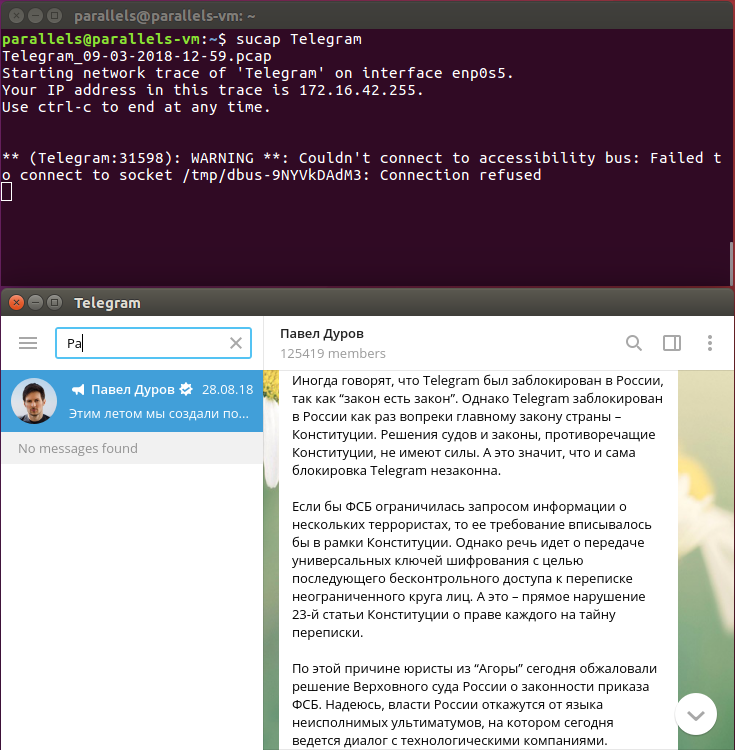


Рис.: Выделенный сетевой интерфейс для приложения Telegram

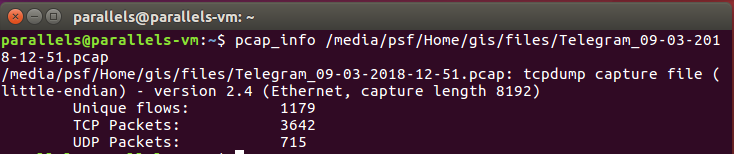
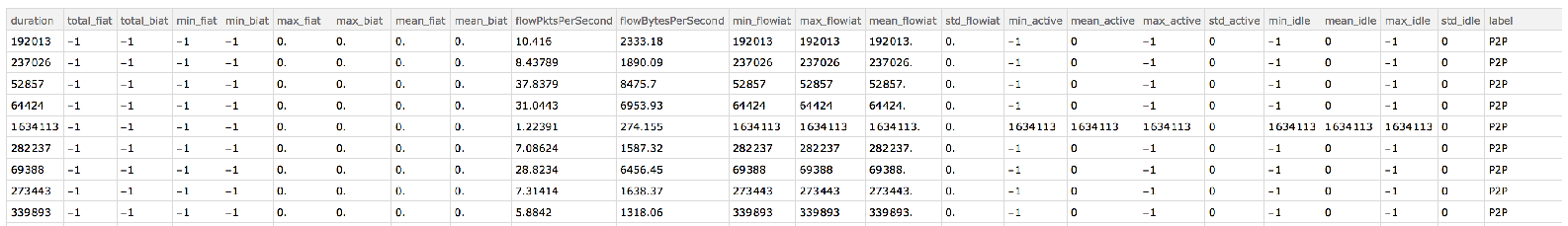


Рис.: Файл снимка сетевого трафика приложения Telegram

Экспериментальные результаты:



# Введение

## Описание программного продукта

Becoming a super hero is a fairly straight forward process:

$ give me super-powers

Super-powers are granted randomly so please submit an issue if you're not happy with yours.

Once you're strong enough, save the world:

// Ain't no code for that yet, sorry  
echo 'You got to trust me on this, I saved the world'

# Aрхитектура системы

# Программный интерфейс управления версии v1 системы Tractor

## /files

### Available endpoints

* [/files/list](#_files_list)
* [/files/process](#_files_process)
* [/file/{job\_id}/status](#Top_of_chapter32_html)
* [/file/{job\_id}/details](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics](#Top_of_chapter32_html)\* [/file/{job\_id}/statistics/labels](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics/ip](#Top_of_chapter32_html)
* [/file/{job\_id}/flows/labels](#Top_of_chapter32_html)
* [/labels/list](#_labels_list)

### /files/list

Collection of available files in repository

### GET:

Get the files collection

### Response code: 200

### application/json (application/json)

[  
 {  
 "name": "some file name",  
 "extension": "pcap",  
 "time\_start": 1530476322,  
 "time\_end": 1530878322,  
 "status": {  
 "state": "finished",  
 "started\_at": 1530878325,  
 "finished\_at": 1530878333,  
 "job\_id": 12345  
 },  
 "size": 102  
 },  
 {  
 "name": "some file name 2",  
 "extension": "pcapng",  
 "time\_start": 1530476322,  
 "time\_end": 1530878322,  
 "status": {  
 "state": "new"  
 },  
 "size": 36  
 }  
]

##### List of File:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| name | string | file name | true |  |
| extension | string | file extension | true |  |
| time\_start | integer | start timestamp period of file | true |  |
| time\_end | integer | end timedtamp period of file | true |  |
| status | object | status of file | true |  |
| size | integer | filesize in kB | true |  |
| flows\_count | integer | Count of flows | false |

status:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| state | string | Value of status | true |  |
| started\_at | integer | Start time of processing | false |  |
| finished\_at | integer | End time of processing | false |  |
| job\_id | integer | uniqie job id | false |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /files/process

Process of available files in repository

### POST:

Start process of file

### FileName (application/json)

{  
 "file\_name": "some\_file\_name\_with\_extension.pcap"  
}

##### FileName:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| file\_name | string | file name with extension | true |

### Response code: 200

### FileStatus (application/json)

{  
 "state": "in\_progress",  
 "started\_at": 1530878325,  
 "job\_id": 12345  
}

##### FileStatus:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| state | string | Value of status | true |  |
| started\_at | integer | Start time of processing | false |  |
| finished\_at | integer | End time of processing | false |  |
| job\_id | integer | uniqie job id | false |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

## /file

### Available endpoints

* [/files/list](#_files_list)
* [/files/process](#_files_process)
* [/file/{job\_id}/status](#Top_of_chapter32_html)
* [/file/{job\_id}/details](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics](#Top_of_chapter32_html)\* [/file/{job\_id}/statistics/labels](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics/ip](#Top_of_chapter32_html)
* [/file/{job\_id}/flows/labels](#Top_of_chapter32_html)
* [/labels/list](#_labels_list)

### /file/{job\_id}/status

Status of file

### GET:

Get status of file

### Response code: 200

### FileStatus (application/json)

{  
 "state": "in\_progress",  
 "started\_at": 1530878325,  
 "job\_id": 12345  
}

##### FileStatus:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| state | string | Value of status | true |  |
| started\_at | integer | Start time of processing | false |  |
| finished\_at | integer | End time of processing | false |  |
| job\_id | integer | uniqie job id | false |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /file/{job\_id}/details

### GET:

Get file details by job id

### Response code: 200

### File (application/json)

{  
 "name": "some file name",  
 "extension": "pcap",  
 "time\_start": 1530476322,  
 "time\_end": 1530878322,  
 "status": {  
 "state": "finished",  
 "started\_at": 1530878325,  
 "finished\_at": 1530878333,  
 "job\_id": 12345  
 },  
 "size": 102,  
 "flows\_count":30000  
}

##### File:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| name | string | file name | true |  |
| extension | string | file extension | true |  |
| time\_start | integer | start timestamp period of file | true |  |
| time\_end | integer | end timedtamp period of file | true |  |
| status | object | status of file | true |  |
| size | integer | filesize in kB | true |  |
| flows\_count | integer | Count of flows | false |

status:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| state | string | Value of status | true |  |
| started\_at | integer | Start time of processing | false |  |
| finished\_at | integer | End time of processing | false |  |
| job\_id | integer | uniqie job id | false |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /file/{job\_id}/statistics

### GET:

Get general statistic of processed file

### Response code: 200

### application/json (application/json)

[  
 {  
 "title": "protocols",  
 "content": [  
 {  
 "label": "ftp",  
 "count": 10  
 },  
 {  
 "label": "udp",  
 "count": 16  
 }  
 ]  
 },  
 {  
 "title": "clients",  
 "content": [  
 {  
 "label": "ip",  
 "count": 10  
 }  
 ]  
 }  
]

##### List of StatisticItem:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| title | string | Title of statistic item | true |  |
| content | items array | Iterable array of content items | true |

items:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| label | string |  | true |  |
| count | integer |  | true |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /file/{job\_id}/statistics/labels

### GET:

Get statistics based by Labels

###### Query parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Examples |
| label | string |  | true |

### Response code: 200

### application/json (application/json)

[  
 {  
 "title": "protocols",  
 "content": [  
 {  
 "label": "ftp",  
 "count": 10  
 },  
 {  
 "label": "udp",  
 "count": 16  
 }  
 ]  
 },  
 {  
 "title": "clients",  
 "content": [  
 {  
 "label": "ip",  
 "count": 10  
 }  
 ]  
 }  
]

##### List of StatisticItem:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| title | string | Title of statistic item | true |  |
| content | items array | Iterable array of content items | true |

items:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| label | string |  | true |  |
| count | integer |  | true |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /file/{job\_id}/statistics/ip

### GET:

Get statistics based by IP

###### Query parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Examples |
| ip | string |  | true |

### Response code: 200

### DetailsIp (application/json)

{  
 "country": "USA",  
 "city": "NewYork",  
 "user\_agent": "Macintosh/Mozilla1.0",  
 "domain\_name": "example.com",  
 "statistics": [  
 {  
 "title": "protocols",  
 "content": [  
 {  
 "label": "ftp",  
 "count": 10  
 }  
 ]  
 },  
 {  
 "title": "cients",  
 "content": [  
 {  
 "label": "ip",  
 "count": 10  
 }  
 ]  
 }  
 ]  
}

##### DetailsIp:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| country | string | County | true |  |
| city | string | City | true |  |
| user\_agent | string | User Agent | false |  |
| domain\_name | string | Domain name | false |  |
| statistics | items array | Iterable array of statistics items | true |

items:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| title | string | Title of statistic item | true |  |
| content | items array | Iterable array of content items | true |

items:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| label | string |  | true |  |
| count | integer |  | true |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### /file/{job\_id}/flows/labels

### GET:

Get statistics based by Labels

###### Query parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Examples |
| label | string |  | true |

### Response code: 200

### application/json (application/json)

[  
 {  
 "client\_ip": "192.168.88.80",  
 "client\_port": 34567,  
 "server\_ip": "8.8.8.8",  
 "server\_port": 80,  
 "time\_start": 12345678,  
 "time\_end": 12345679,  
 "comment": "some comment"  
 },  
 {  
 "client\_ip": "192.167.88.82",  
 "client\_port": 34,  
 "server\_ip": "1.1.1.1",  
 "server\_port": 9090,  
 "time\_start": 12345678,  
 "time\_end": 12345679,  
 "comment": "some comment"  
 }  
]

##### List of DetailsLabelItem:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| client\_ip | string | Client IP | true |  |
| client\_port | integer | Client Port | true |  |
| server\_ip | string | Server IP | true |  |
| server\_port | integer | Server Port | true |  |
| time\_start | integer | Session start time | true |  |
| time\_end | integer | Session end time | true |  |
| comment | string | Comment | true |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

## /labels

### Available endpoints

* [/files/list](#_files_list)
* [/files/process](#_files_process)
* [/file/{job\_id}/status](#Top_of_chapter32_html)
* [/file/{job\_id}/details](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics](#Top_of_chapter32_html)\* [/file/{job\_id}/statistics/labels](#Top_of_chapter32_html)
* [/file/{job\_id}/statistics/ip](#Top_of_chapter32_html)
* [/file/{job\_id}/flows/labels](#Top_of_chapter32_html)
* [/labels/list](#_labels_list)

### /labels/list

### GET:

Get labels collection

### Response code: 200

### application/json (application/json)

[  
 {  
 "group\_name": "social",  
 "labels": [  
 "facebook",  
 "vk"  
 ]  
 },  
 {  
 "group\_name": "peer2peer",  
 "labels": [  
 "bittorrent",  
 "skype"  
 ]  
 },  
 {  
 "group\_name": "peer2peer",  
 "labels": [  
 "vpn",  
 "sokcs5"  
 ]  
 },  
 {  
 "group\_name": "other",  
 "labels": [  
 "ssh",  
 "telnet"  
 ]  
 }  
]

##### List of Label:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| group\_name | string | Group name of certain labels | true |  |
| labels | array | array of labels | true |

### Response code: 404

### Error (application/json)

{  
 "status": 404,  
 "code": 404,  
 "message": "Not found"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |

### Response code: 500

### Error (application/json)

{  
 "status": 500,  
 "code": 500,  
 "message": "Internal server error"  
}

##### Error:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Required | Pattern |
| status | integer |  | true |  |
| code | integer |  | true |  |
| message | string |  | true |