CLOUDFLARE

DNS Analytics API BETA

Dimensions

Dimensions can be used to break down the data by given attributes.
In API requests, dimensions are set in the dimensions parameter. If you need to list multiple dimensions, separate them with commas.

Dimension	Name	Example	Plan Availability	Notes
responseCode	Response Code	NOERROR	Free, Pro, Business, Enterprise	Response codes defined by IANA (always uppercase).
ipVersion	IP Version	6	Free, Pro, Business, Enterprise	Shows distribution of queries over IPv4 vs IPv6. (IPv4 is 4, IPv6 is 6)
tcp	TCP	1	Free, Pro, Business, Enterprise	Shows distribution of queries over TCP vs UDP. (UDP is 0, TCP is 1)
dayOfWeek	Day Of Week	1	Free, Pro, Business, Enterprise	Break down by day of week, (Monday is 1, and Sunday is 7).
coloName	Colo Name	SJC	Pro, Business, Enterprise	PoP airport code.
queryType	Query Type	AAAA	Business, Enterprise	Types defined by IANA, unknown types are empty.
queryName	Query Name	example.com	Enterprise	

Metrics

A metric is a numerical value that is based on an attribute of the data, for example a query count.
In API requests, metrics are set in the metrics parameter. If you need to list multiple metrics, separate them with commas.

Metric	Name	Example	Plan Availability	Unit

queryCount	Query count	1000	Free, Pro, Business, Enterprise	Count
responseTimeAvg	Average response time	1.0	Business, Enterprise	Time in milliseconds.
responseTimeMedian	Median response time	1.0	Business, Enterprise	Time in milliseconds.
responseTime90th	90th percentile response time	1.0	Business, Enterprise	Time in milliseconds.
responseTime99th	99th percentile response time	1.0	Business, Enterprise	Time in milliseconds.

GET Table

Retrieves a list of summarised aggregate metrics over a given time period.
permission needed: #analytics:read

```
GET /zones/:identifier/dns_analytics/report
```

Optional parameters

Name	Description	Constraints
metrics *array*	Array of metrics ["queries", "avgResponseTime"]	
dimensions *array*	Array of dimensions ["responseCode", "queryName"]	
sort *array*	Array of dimensions to sort by, each dimension may be prefixed by - (descending) or + (ascending) ["+responseCode", "-queryName"]	
filters *string*	Segmentation filter in 'attribute operator value' format "queryName=='example.com'"	Note that filters don't work over metrics, so you can't i.e. find time range when queryCount<1000, but they work over dimensions, so you can do queryName==example.com.
since *string*	Start date and time of requesting data period in the ISO8601 format "2016-11-11T12:00:00Z"	
until *string*	End date and time of requesting data period in the ISO8601 format	

	"2016-11-11T13:00:00Z"	
limit *integer*	Limit number of returned metrics, default is 100 100	

cURL (example)

```
$ curl -X GET
"https://api.cloudflare.com/client/v4/zones/9a7806061c88ada191ed06f989cc3dac/dns_analytics/report?dimensions=responseCode,queryName&metrics=queryCount,responseTimeAvg&sort=+responseCode,-queryName&filters=responseCode==NOERROR&since=2016-11-11T12:00:00Z&until=2016-11-11T13:00:00Z&limit=100" \
-H "X-Auth-Email: user@example.com" \
-H "X-Auth-Key: API-KEY" \
-H "Content-Type: application/json"
```

Response (example)


```
{
 "success": true,
 "errors": [],
 "messages": [],
 "result": {
 "data": {
 "dimensions": [
 {
 "name": "NODATA"
 }
 ],
 "metrics": [
 1.5,
 2
 ]
 },
 "totals": {
 "queryCount": 1000,
 "responseTimeAvg": 1
 },
```

```
"min": {
"queryCount": 1000,
"responseTimeAvg": 1
},
"max": {
"queryCount": 1000,
"responseTimeAvg": 1
}
},
"query": {
"dimensions": [
"responseCode",
"queryName"
],
"metrics": [
"queryCount",
"responseTimeAvg"
],
"sort": [
"+responseCode",
"-queryName"
],
"filters": "responseCode==NOERROR",
"since": "2016-11-11T12:00:00Z",
"until": "2016-11-11T13:00:00Z",
"limit": 100
}
}
```


GET Interval

Retrieves a list of aggregate metrics grouped by time interval.
permission needed: #analytics:read.

GET /zones/:identifier/dns_analytics/report/bytime

***Optional* parameters**

Name	Description	Constraints

metrics *array*	Array of metrics ["queries", "avgResponseTime"]	
dimensions *array*	Array of dimensions ["responseCode", "queryName"]	
sort *array*	Array of dimensions to sort by, each dimension may be prefixed by - (descending) or + (ascending) ["+responseCode", "-queryName"]	
filters *string*	Segmentation filter in 'attribute operator value' format "queryName=='example.com'"	Note that filters don't work over metrics, so you can't i.e. find time range when queryCount<1000, but they work over dimensions, so you can do queryName==example.com.
since *string*	Start date and time of requesting data period in the ISO8601 format "2016-11-11T12:00:00Z"	
until *string*	End date and time of requesting data period in the ISO8601 format "2016-11-11T13:00:00Z"	
limit *integer*	Limit number of returned metrics, default is 100 100	
time_delta *string*	Unit of time to group data by "hour"	valid values: all, auto, year, quarter, month, week, day, hour, dekaminute, minute

cURL (example)

```
$ curl -X GET
"https://api.cloudflare.com/client/v4/zones/9a7806061c88ada191ed06f989cc3dac/dns_analytics/report/bytime?dimensions=responseCode,queryName&metrics=queryCount,responseTimeAvg&sort=+responseCode,-queryName&filters=responseCode==NOERROR&since=2016-11-11T12:00:00Z&until=2016-11-11T13:00:00Z&limit=100&time_delta=hour" \
-H "X-Auth-Email: user@example.com" \
-H "X-Auth-Key: API-KEY" \
-H "Content-Type: application/json"
```

Response (example)

```
{
 "success": true,
 "errors": [],
 "messages": [],
 "result": {
 "data": {
 "dimensions": [
 {
 "name": "NODATA"
 }
 ],
 "metrics": [
 [
 1.5,
 2
 ],
 [
 2.3,
 1.5
 ]
 ]
 },
 "totals": {
 "queryCount": 1000,
 "responseTimeAvg": 1
 },
 "min": {
 "queryCount": 1000,
 "responseTimeAvg": 1
 },
 "max": {
 "queryCount": 1000,
 "responseTimeAvg": 1
 }
 },
 "query": {
 "dimensions": [
 "responseCode",
```

```
 "queryName"
 ],
 "metrics": [
 "queryCount",
 "responseTimeAvg"
 ],
 "sort": [
 "+responseCode",
 "-queryName"
 ],
 "filters": "responseCode==NOERROR",
 "since": "2016-11-11T12:00:00Z",
 "until": "2016-11-11T13:00:00Z",
 "limit": 100,
 "time_delta": "hour"
 }
}
```

Example API Queries

Here are some curl's you can do against the DNS analytics API to get you started.

Query count by DNS record

```
$ curl -X GET
"https://api.cloudflare.com/client/v4/zones/####/dns_analytics/report?metrics=queryCount&dimensions=queryName,queryType" \
-H "X-Auth-Email: user@example.com" \
-H "X-Auth-Key: API-KEY" \
-H "Content-Type: application/json"
```

Response code returned per DNS record (search for NXDOMAIN for queries causing errors)

```
$ curl -X GET
"https://api.cloudflare.com/client/v4/zones/####/dns_analytics/report?metrics=queryCount&dimensions=queryName,queryType,responseCode" \
-H "X-Auth-Email: user@example.com" \
-H "X-Auth-Key: API-KEY" \
-H "Content-Type: application/json"
```

Response codes returned per colo

```
$ curl -X GET
"https://api.cloudflare.com/client/v4/zones/####/dns_analytics/report?metrics=queryCount&dimensions=queryName,queryType,coloName,responseCode" \
-H "X-Auth-Email: user@example.com" \
-H "X-Auth-Key: API-KEY" \
-H "Content-Type: application/json"
```

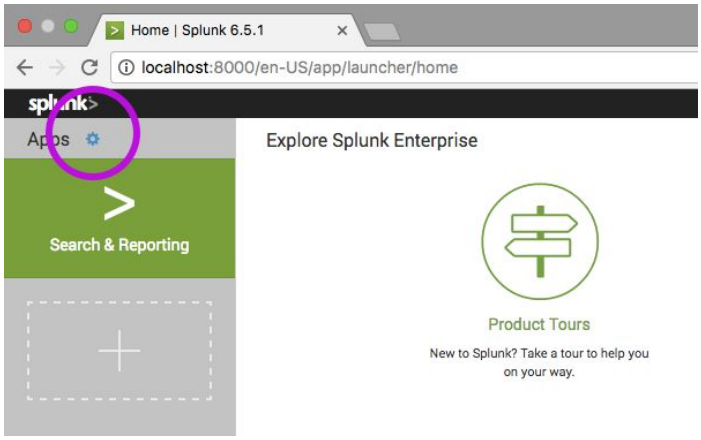
Top queries causing errors

```
$ curl -s -H 'X-Auth-Key:####' -H 'X-Auth-Email:####@####.com'
'https://api.cloudflare.com/client/v4/zones/####/dns_analytics/report?metrics=queryCount&dimensions=queryName,queryType,responseCode' | jq -r '.result.data|.[]|.|[.dimensions,
.metrics]|map(.[]|.|tostring)|.|join(" ")' | grep -v NOERROR | sort -rn -k 4 | head
```

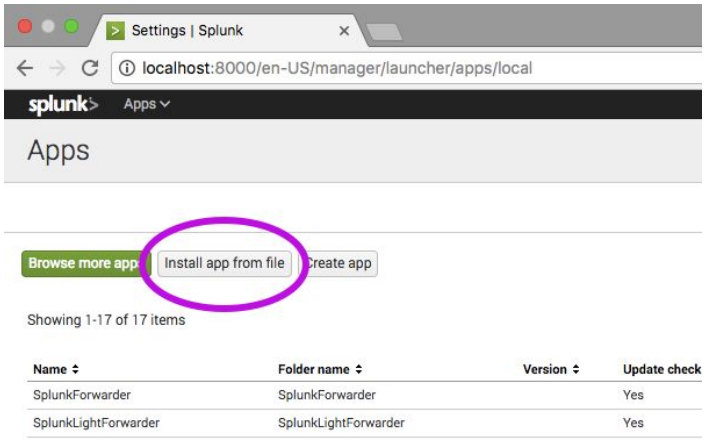
CLOUDFLARE

How to push data from the DNS Analytics API into Splunk

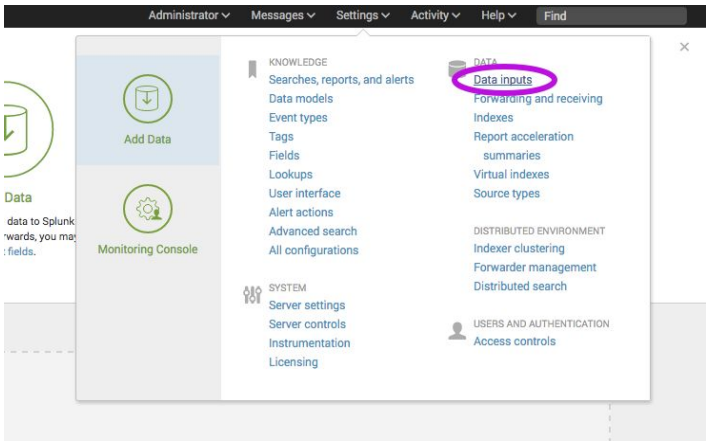
1. To pull data in Splunk from a REST API, you need to download the REST API Modular Input add on for splunk. Download it here: https://splunkbase.splunk.com/app/1546/
2. Now you need to import that app into Splunk. In Splunk, click on the gear in the top left corner.



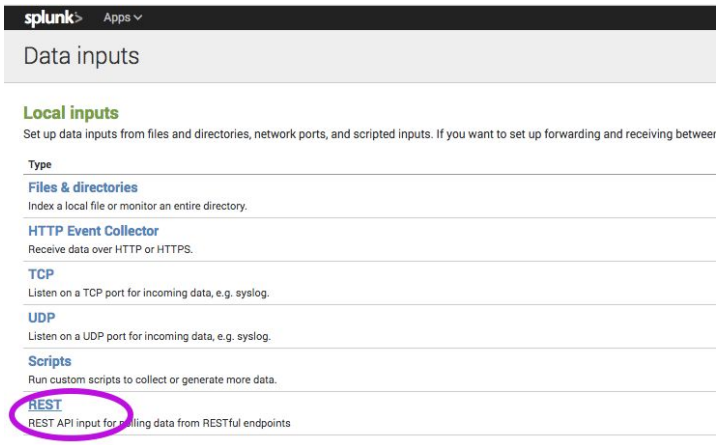
3. Then click 'Install app from file'



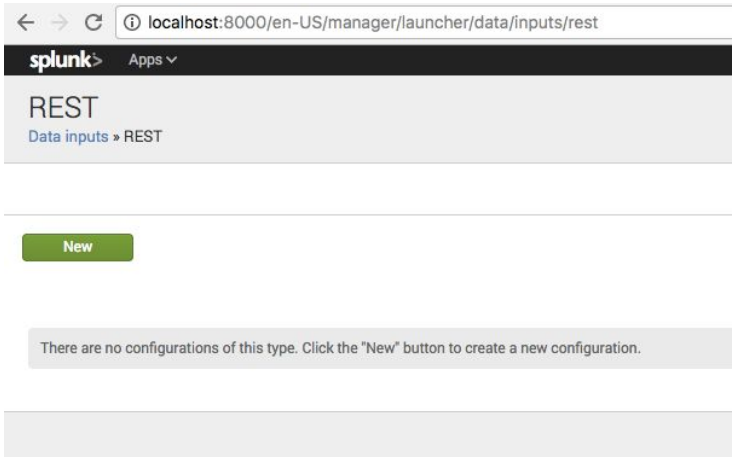
4. Upload the file.
5. You will see it is installed. It's called 'rest_ta'.
6. Now in the top menu, to setup the DNS analytics rest API as a data input, go to the top nav bar, hover on Settings, and click on Data Input



7. Under Local Inputs, click on REST.



8. Click 'New'



9. Add the DNS analytics API.
   Endpoint URL: https://api.cloudflare.com/client/v4/zones/####/dns_analytics/report
   HTTP Method: GET
   Authentication Type: None
   HTTP Header Properties: X-Auth-Key=####,X-Auth-Email=##@##.com
   URL Arguments:
   metrics=queryCount,dimensions=queryName,dimensions=queryType,dimensions=responseCode,dimensions=coloName
   Response type: json
   Response handler: leave blank
   Response handler arguments: leave blank
   Response filter pattern: leave blank
   HTTP Proxy Address: leave blank
   HTTPS Proxy Address: leave blank
   Request Timeout: leave blank
   Backoff time: leave blank
   Polling interval: leave blank
   Sequential stagger time: leave blank
   Delimiter: leave blank
   Set sourcetype: manual
   Source type: json_no_timestamp
10. You will now be able to see the DNS analytics data in Splunk. Go to the Splunk homepage and click on Search & Reporting, and you will see data from the DNS analytics API you just added.