



Introduction to APIs

REST and GraphQL

Module Overview

This module will cover the following:

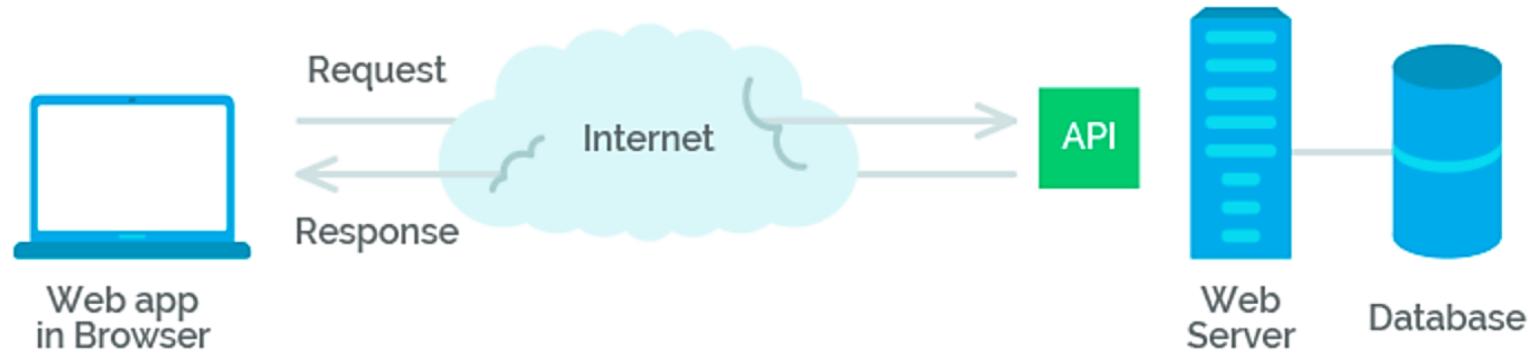
- Getting Started with REST APIs
- REST Requests
- REST Responses
- REST Authentication
- GraphQL



Getting Started with REST APIs

What is an API?

- APIs (application programming interface) are a set of rules and mechanisms by which one application or component interacts with the others



What is REST?

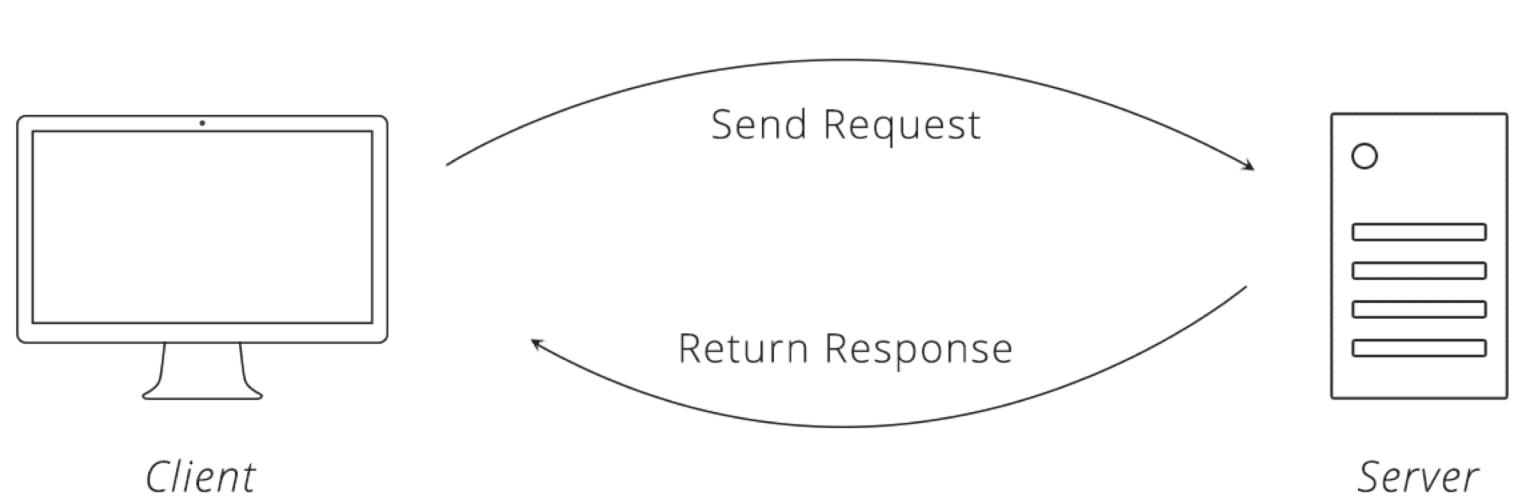
REST stands for **Representational State Transfer**

- Data presentation for a client in the format that is convenient for it to consume.
- REST is not a standard or protocol
 - An approach to writing an API.
 - Architectural style for writing an API.



REST Communication

- REST communication centers around a concept called the **Request-Response Cycle**.



API Documentation

- Real-time visibility into all available endpoints
- Includes parameter, query, path, and response details
- Professionally written and maintained with pseudo-code and code snippet examples
- Explains the **why** and the **how**

SLA Domains

Rubrik clusters provide automated data management and protection through SLA Domains. An SLA Domain defines the protection policies for their assigned snappables (virtual machines, file systems, and applications).

To provide policy based management and protection of a snappable, add the snappable to an SLA Domain, or to multiple

Retrieving SLA Domains

Before assigning snappables to SLA Domains, get a list of the SLA Domains that exist on a Rubrik cluster. For a new Rubrik cluster, only the default SLA Domains. When custom SLA Domains are added to the Rubrik cluster, the list is modified to include them.

Example: Retrieving SLA Domains from a Rubrik cluster

Send a GET request to `/sla_domain`.

```
curl -X GET "https://$cluster_address/api/v1/sla_domain"
```

The Rubrik REST API server returns a `ListResponse` object of all SLA Domains. At a minimum, the `ListResponse` contains the following SLA Domains: Gold, Silver, and Bronze.

```
{
  "data": [
    {
      "id": "$gold_sla_id",
      "name": "Gold"
    },
    ...
  ]
},
```



API Explorer

- Live access to API through industry standard interface familiar to developers
- Documents all available REST endpoints
- Facilitates quick integration development

/cluster : Cluster configuration and health

Show/Hide | List Operations | Expand Operations

Get cluster details

GET /cluster/{id}

Implementation Notes
Retrieve public information about the Rubrik cluster

Response Class (Status 200)
Information about the cluster

Model Example Value

```
{  
  "id": "string",  
  "version": "string",  
  "apiVersion": "string"  
}
```

Response Content Type application/json ▾

Parameters

Parameter	Value	Description	Parameter Type	Data Type
id	me	ID of the Rubrik cluster or me for self.	path	string

Try it out! Hide Response

Curl

```
curl -X GET --header 'Accept: application/json' 'https://172.21.8.51/api/v1/cluster/me'
```

Request URL

```
https://172.21.8.51/api/v1/cluster/me
```

Response Body

```
{  
  "id": "89fc0d86-6f1c-4652-aefa-37b7ba0e6229",  
  "version": "4.0.4-568",  
  "apiVersion": "1"  
}
```



cURL

- cURL is an open source command line utility called cURL
- Most API documentation is written with reference to cURL, for example:

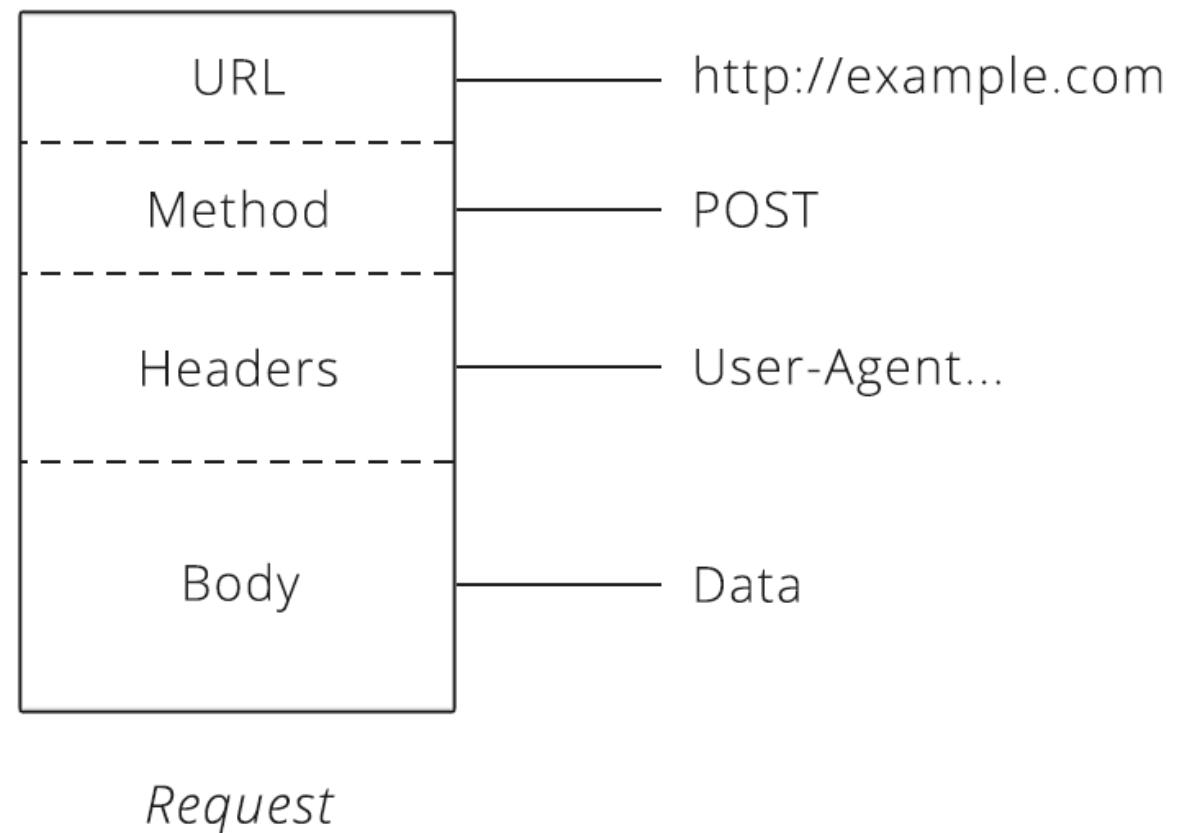
```
curl -k -u admin:pass -X GET  
"https://$cluster_address/api/v1/cluster/me"
```



REST Requests

Anatomy of a Request

- A request is made up of four things:
 - Endpoint
 - Method
 - Headers
 - Body (or data)



Endpoint

- **The endpoint** (or route) is the URL you request for. It follows this structure:
 - root-endpoint/[path](#)
- The **root-endpoint** is the starting point of the API you're requesting
 - <https://api.github.com>
 - [https://\\$Rubrik_cluster_IP/api/](https://$Rubrik_cluster_IP/api/)
- The **path** determines the resource you are requesting
 - <https://api.github.com/orgs/{org}>
 - [https://\\$Rubrik_cluster_IP/api/v1/cluster/{id}/api_version](https://$Rubrik_cluster_IP/api/v1/cluster/{id}/api_version)



Method

REST relies heavily on HTTP.

Each operation uses its own HTTP method:

- GET – getting
- POST – creation
- PUT – full object update (modification)
- PATCH – partial object update (modification)
- DELETE – removal



Headers and Body

- **Headers** provide meta-information about a request
- The request **body** contains the data the client wants to send the server



Request Example

 Rubrik REST API Explorer Authorize

Rubrik REST API

/cluster : Cluster configuration and health Show/Hide | List Operations | Expand Operations

GET /cluster/{id} Endpoint Get cluster details

Method Method

Implementation Notes
retrieve public information about the Rubrik cluster

Response Class (Status 200) i
Information about the cluster

Model Example Value

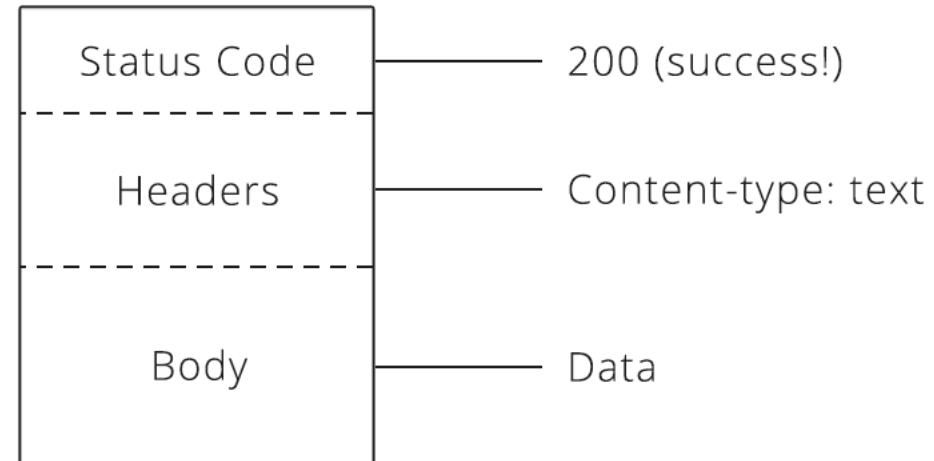
```
{  
  "id": "string",  
  "version": "string",  
  "apiVersion": "string"  
}
```

REST Response

Anatomy of a Response

APIs respond with:

- Status Code
- Headers
- Body or Data



Response



HTTP Status

- 1xx - Informational
- 2xx - Success
- 3xx - Redirection (client needs to take additional action)
- 4xx - Client error
- 5xx - Server error



Data Formats

- Data is returned in a conveniently formatted header and body using JSON or XML



key

value

```
{ "crust": "original" }
```



Response Example

```
Curl
curl -X GET --header 'Accept: application/json' --header 'Authorization: Basic ZmlsaXAudmVybG95QHJ1YnJpay5kZW1v0lJ1YnJp
Request URL
https://172.17.28.32/api/v1/cluster/me
URL
Response Body
{
  "id": "4a5d5fc5-a764-4c9c-8908-9d56e4221fd6",
  "version": "4.0.5-607",
  "apiVersion": "1"
}
Endpoint
JSON
Response Code
Code
200
```

The diagram illustrates a response example from a curl command. It is organized into several sections:

- Curl:** Shows the full curl command with headers for JSON accept and basic authentication.
- Request URL:** Shows the URL `https://172.17.28.32/api/v1/cluster/me`.
- Response Body:** Shows the JSON response:

```
{
  "id": "4a5d5fc5-a764-4c9c-8908-9d56e4221fd6",
  "version": "4.0.5-607",
  "apiVersion": "1"
}
```
- Response Code:** Shows the status code `200`.

Red arrows with labels point to specific parts of the response:

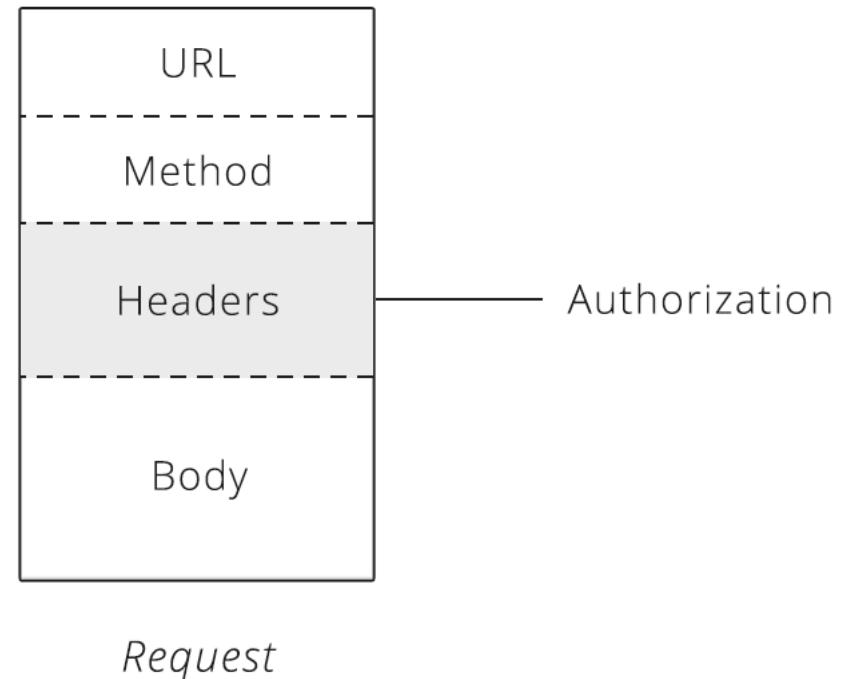
- A red arrow labeled **URL** points to the Request URL.
- A red arrow labeled **Endpoint** points to the Response Body.
- A red arrow labeled **JSON** points to the Response Body.
- A red arrow labeled **Code** points to the Response Code.



REST Authentication

Anatomy of Authentication

- There are two main ways to authenticate:
 - Basic authentication (username and password)
 - Token-based
 - Includes oAuth, which lets you to authentication with social media networks such as GitHub, Google, Twitter, Facebook, etc.



Authenticating with cURL

- To perform a basic authentication with cURL, you can use the `-u` option, followed by your username and password. For example:
- `curl -x POST -u "username:password" https://api.github.com/user/repos`



Authenticating using API Explorer

The screenshot shows the Rubrik REST API Explorer interface. At the top, there's a dark header bar with the Rubrik logo and the text "Rubrik REST API Explorer". On the right side of the header is a green "Authorize" button. Below the header, the main content area has a title "Rubrik REST API" and a sub-section title "/cluster : Cluster config". Under this, there's a "GET /cluster/{id}" button, "Implementation Notes" (describing how to retrieve public information about a cluster), and a "Response Class (Status 200)" section (describing information about the cluster). To the right of these sections, a large callout box titled "Available authorizations" provides instructions for authentication. It lists two methods: "Basic authentication - authorized" (with a note about a specific username) and "Api key authorization" (with a note about an "api_key" header). Red arrows point from the explanatory text on the right to each method. The background of the callout box is white, and it has a semi-transparent yellow overlay at the bottom.

Available authorizations

Basic authentication - authorized
username: filip.verloy@rubrik.demo

Logout

Either use Basic authentication with a username and password

Or use

Api key authorization

name: Authorization
in: header
value:

Authorize

Cancel

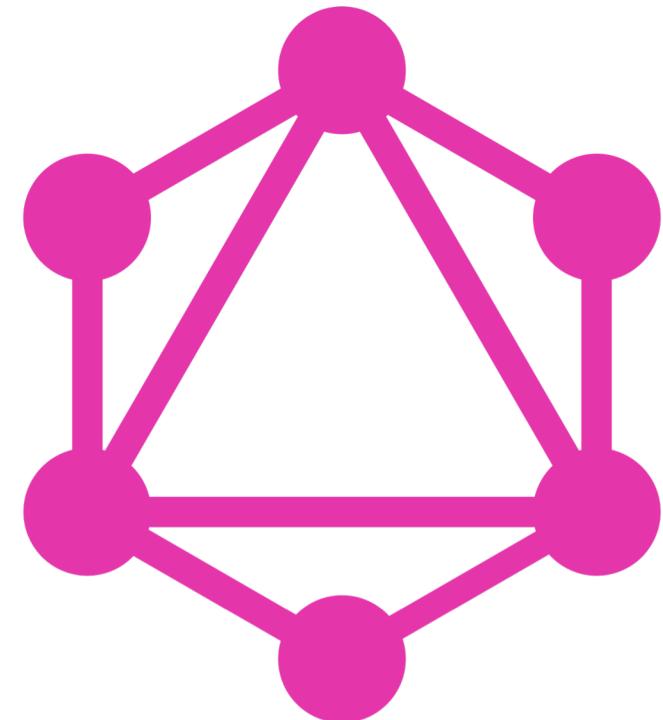
Operations | Expand Operations

Get cluster details

GraphQL

Why GraphQL?

- GraphQL is an API query and manipulation language built by Facebook in 2012, and publicly released in 2015.
- It uses HTTP as its transport method and leverages POST calls to provide increased efficiency around GET requests to REST APIs.
 - Request information you want via payload
 - Receive only the information you requested back



GraphQL Advantages

- Advantages of GraphQL over REST API:
 - Complex queries return spuriously-related properties of objects without having to issue multiple requests
 - Return only the data needed, so response times and processing are reduced





build

Building the Future of Data Management