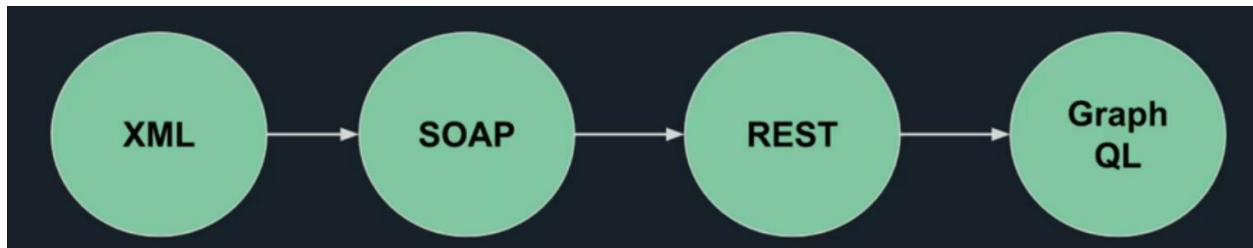


API: It is exchange data between endpoint it can be

- Backend server
- Mobile application
- Desktop application
- Frontend application

Data store in the same place in backend server, all client connect to backend server so if you have mobile application (android , Linux , iOS , .....)

API standards



First, company using xml to exchange data query data from server to another server. It is called standard xml is RPC

**Execute function in another on server it is xml request** along with required and server to execute function and return the result to original server but it was complicated all of security issue and it wasn't standard and every company have a structure on xml.

So, they develop SOAP it is call and standard for API so be **using** xml as format

Yahoo, Ebay using SOAP.

REST came to fix issue with SOAP. It is so complicated and NOT Flexible and performance and security is issue.

So, REST depend on HTTP protocol and method to exchange data

Example:

If you want to query data you using GET and query data using POST request it is simple and standard for developer.

- REST using (JSON or XML )

GraphQL : It is query and handling data used special structure data.

## XML - SOAP

- Envelope
  - HEADER
  - /Header
  - Body
  - /Body
- /Envelope

## SOAP Example

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="
http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="
http://www.w3.org/1999/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Header>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns1:sayHelloTo xmlns:ns1="Hello"
      SOAP-ENV:encodingStyle="
http://schemas.xmlsoap.org/soap/encoding/">
      <name xsi:type="xsd:string">John</name>
    </ns1:sayHelloTo
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Header: Contain general information about the Message

Body: contain the message itself.

The data you want to Query and parameter you want to pass.

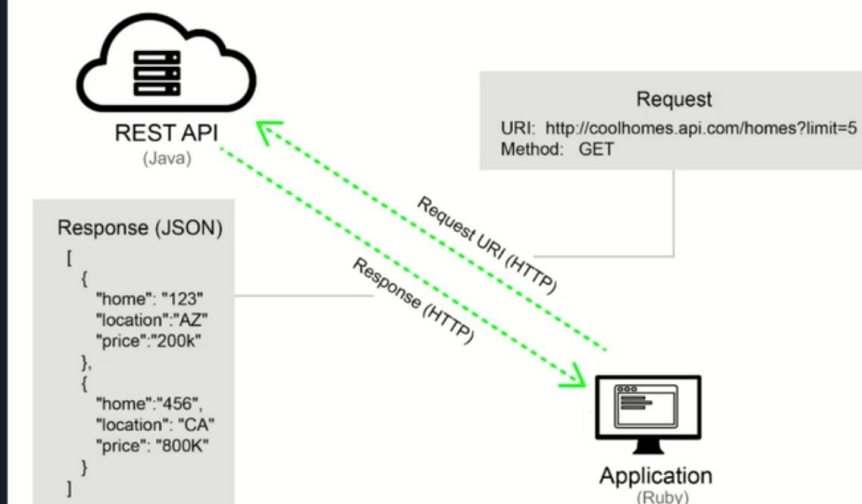
So you can see

It consists of SOAP-ENV

## REST

>> HTTP REQUEST  
<< HTTP RESPONSE

### REST API model

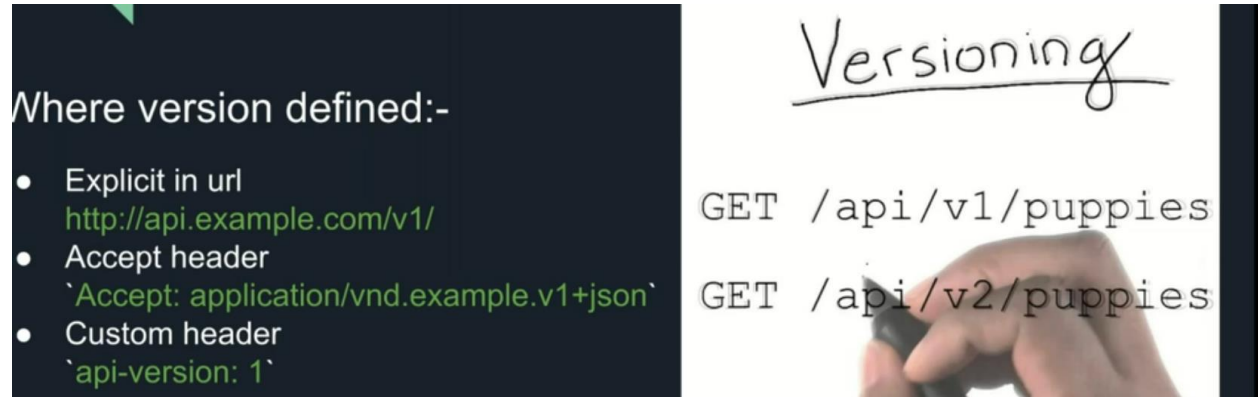


It is normal HTTP request and it handle HTTP response. In this example ruby

## API versioning

Where version defined:

- 1- Explicit in URL: the developer expose the version in the URL
- 2- Accept Header: it is header the request tell API server that you accept this type of data content structure (EX. Application)
- 3- Custom Header it is API



Imagine you are developer and develop you own API

- Add new folder to database “ EX-change login email”

In traditional way

- The backend sever you handle new data structure.
- BUT How the mobile application connect to up data to mobile application the user who updated application using API with no problem but the old user didn't updated mobile application yet have problem or even exception.

( So API version came to fix this issue )

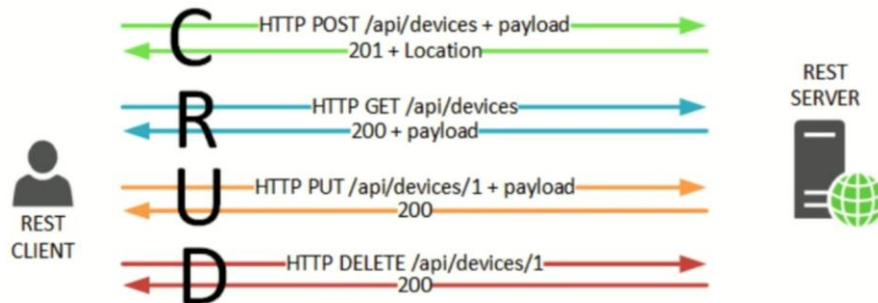
API version meaning have version API example

If you released version one your API and you want to change the data you really version two of the API and release new update to the mobile application so the customer`s who updated the mobile use all application related version and use v2 to API the old customer will use old version mobile application without problem.

## REST

It is used HTTP request to exchange data between client and server so for example this is a simple crud operation

CRUD stand for Create Read Update Delete



So, if client want to create entity the backend server it send HTTP ( POST request ) to the endpoint with the data you have to create and the backend server return the result for example status success [ 200 ]

To Read Data:- client will be send HTTP (GET-request) he will receive data as u want to read.

If client wants to update data he will send the HTTP (PUT-request) and the backend server process the request and update the data and return the result.

if client wants to delete entity, he will send HTTP (DELETE-request) to delete request to the API backend server.

\*\*\*\*\* it is the standard \*\*\*\*\*

Note:-but it is not case of all the API

Some developer which is not familiar with REST or just don't want to follow the standard it is used

for all operation GET or POST (use for delete entity ) so, be carful don't expect the developer to follow the standard.

but any way according to the standard you must using API for each HTTP method according to the operation he want to do

HTTP method

- GET /HEAD => read resource
- POST => Create Resource
- PUT/PATCH => Update resource
- Delete => Delete resource

**It is most common Method.**

other HTTP Methods = [ CONNECT – OPTION – TRACE ]

As well REST it will contain HTTP request and HTTP response

So, lets have look at the HTTP headers send with (request) first

There is many send headers with request and with the response

if you want to do big a order search in google find great resources int the header but will focus couple of

- 1- header Authentication (ex: Authorization :Bearer cn389ncoiwuencr) following the TOKEN

**(The Authentication header user to authenticate or authorize the user)**

- 2- Header Accept(ex:Accept:text/html)

**( The client who make request to backend server need to send need to Say: what is the data structure he accept so for example: if u have mobile application and backend server it is accepted the application/json/ in this case he tell the backend server he only accept only json data so the backend server he will send data in this format but again : not it is standard the developer ignore it the client send request but the response return for txt ? !!! )**

- 3- header Cookie Header (ex Cookie:\_ga=GA1.215755156156.322561)

( The client can send the cookies to the backend server to favorite Authentication or tracker

- Now Lets See what it is HTTP Response return

- 1- Authentication

- WWW-Authenticate(ex: WWW-Authenticate:Basic)

**(It u send request and the server no authentication server support header and tell the client to authenticate he using Basic Authentication)**

```
{ "API" : "SECURITY" }
```

## 2- Content Negotiation

- Content-type (ex:Content-Type:text/html)

**(what type of content he return in the request body for example: the server returned HTML so the content type will be text/html )**

## 3- Cookie

- Set-Cookie (ex:Cookie:csrf\_token:P213121235sadsadq)

( if the server want to set the cookie on the client he return header called set-cookie with (**key** and **value**) of the cookie,

Note: It is important header to look for

### 1- HTTP access control (CORS)

- Access-Control-Allow-Origin(ex:Access-Control-Allow-Origin:\*)
- Access-Control-Allow-Methods(ex:Access-Control-Allow-Methods:GET,OPTION)

#### HTTP access control (CORS)

Access-Control-Allow-Origin ( ex: **Access-Control-Allow-Origin: \*** )

Access-Control-Allow-Methods ( ex: **Access-Control-Allow-Methods: GET, OPTIONS** )

#### Custom Headers

X-Custom-Headers ( ex: **X-Custom-Headers: Any** )

**(This header origin for mean he will request for a resource for another domain or subdomain this header actually for browser to control a cros AJAX request from frontend to backend)**

- Frist, it control which domain or subdomain it is allowed to access the URL and process this request
- Second, which control what method allow in CORS request
- Lastly, The Developer can return Custom Header

### 2- Custom Header

- X-Custom-Header(ex:X-Custom-Header:Any )

```
{ "API" : "SECURITY" }
```

Any HTTP response it is need to have the status code

```
200 OK
201 Created

301 Moved Permanently

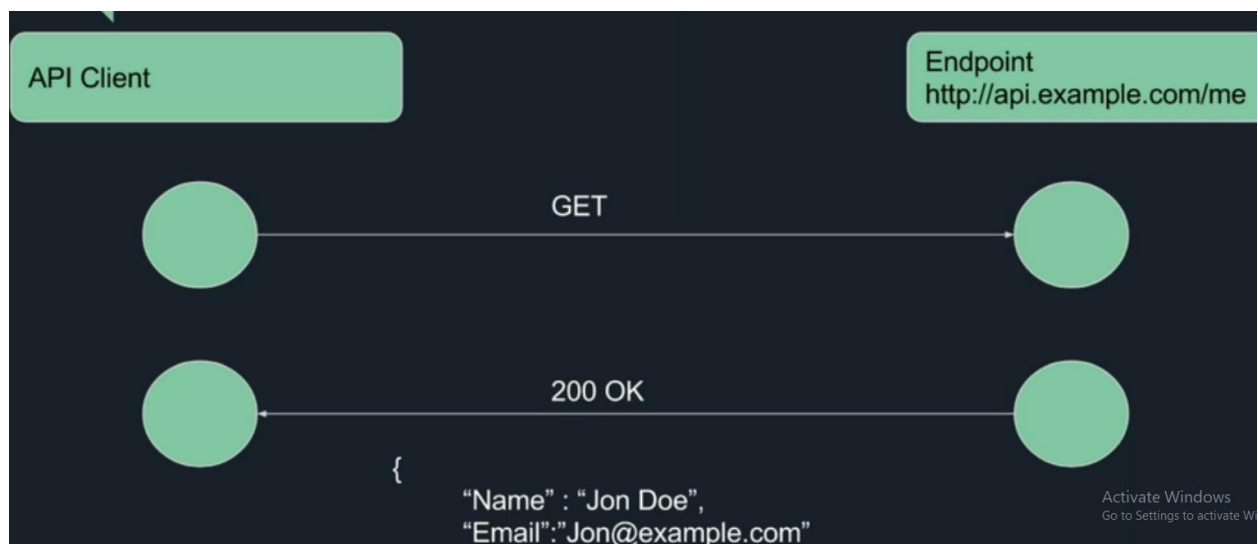
400 Bad Request
401 Unauthorized
403 Forbidden
404 Not Found
405 Method Not Allowed

500 Internal Server Error
```

301 for redirect

Again, it is standard but developer can return any status code he want.

### Example for query data



Let's have query data he want to query data for authenticate the user client send GET request to the endpoint and the End-point process the request and return 200 along with data for JSON Format

```
{ "API" : "SECURITY" }
```

Example e-commerce website to add product in shopping card

The API client send the POST request to API end-point with data



The endpoint will process the request and return 201 to create with status success.



what do you want to know ?

1- Where is the API endpoints?

(First, how client connect to the API Where is the backend server domain or subdomain and u need to)

2- How Developer handle versioning?

(where it is explicit in URL or in header or something else )

3- What is the programming Language use ?

( To know exploit the language self or something effected this programming language and you want to know ... )

4- What is backend data storage used ?

( SQL - Database - MySQL - SQL Server - no SQL - database liTe - mongo DB u need to know what is the cache API by using vulnerable cache or custom or domin cache that we help u to understand more how the API handle the request )

5- How client authenticate to use the API?

( It is very important to now or understand how API is authenticated .. because most of API vulnerabilities authentication follow it self )

1- Where is the API endpoints?

a. Public information are good

( Many website exposed API endpoint and released the developer help document to educated the developer how to use the API -\*- it is very vulnerability method to used and so Let`s try to target twitter for example here we can find that API twitter.com endpoint that client connect to with twitter backend end server Actually u can learn about by just serving documentation for the target)

3.API Fingerprinting - x The Search API — Twit: x +

https://dev.twitter.com/rest/public/search

envoy-iad server

### REST APIs

- API Rate Limits
- Rate Limits: Chart
- The Search API
- The Search API: Tweets by Place
- Working with Timelines

## How to build a query

The best way to build a query and test if it's valid and will return matched Tweets is to first try it at [twitter.com/search](https://twitter.com/search). As you get a satisfactory result set, the URL loaded in the browser will contain the proper query syntax that can be reused in the API endpoint. Here's an example:

1. We want to search for Tweets referencing @twitterapi account. First, we run the search on [twitter.com/search](https://twitter.com/search)
2. Check and copy the URL loaded. In this case, we got: <https://twitter.com/search?q=%40twitterapi>
3. Replace "https://twitter.com/search" with "https://api.twitter.com/1.1/search/tweets.json" and you will get: <https://api.twitter.com/1.1/search/tweets.json?q=%40twitterapi>
4. *Execute this URL to do the search in the API*

## 2- Subdomain Brute Force?

( It is the second step if you don't have any public information in documentation most of API endpoint it is subdomain to separate the backend server API from the main web site which mean static or another programming language, my Favorite tool Knockpy )

it is command

-----

knockpy hackerone.com

- it is search of a lot of thing
- it is search viurstool database
- search for DNS zone transfer
- it is search in search engine
- it is scanning for subdomain

links: [KnockPy](#)



```
knockpy hackerone.com
build CHANGELOG.rst dist knockpy knockpy.egg-info README.rst setup.py
-> knock git:(4.1) X knockpy hackerone.com
A detailed reference on this API endpoint can be found at GET /search/tweets

4.1.1
KnockPy build a query
This tool will build a query and test if it's valid and will return matched Tweets.
If you get a result set, the URL loaded in the browser will contain the result.

+ checking for virustotal subdomains: SKIP
+ VirusTotal API_KEY not found
+ checking for wildcard: NO
+ checking for zonetransfer: NO
+ resolving target: YES
+ scanning for subdomain...

Ip Address      Status  Type      Domain Name      Server
-----
104.16.99.52    301     host      api.hackerone.com cloudflare-nginx
104.16.100.52   301     host      api.hackerone.com cloudflare-nginx
```

### Gathering info

104.16.99.25 301 host api.hackerone.com endpoint= cloudflare.nginx (The server type of endpoint)

3- you can bug to putting proxy like burp suite he will go throw to catch endpoint

Note

- It may be more than one API more than endpoint the API so let's

Example

hackerone/souq

Site: <https://hackerone.com/souq>

he will find two API

- [api.souq.com](https://api.souq.com) : it is general purpose
- [mobileapisouq.com](https://mobileapisouq.com) : it is for mobile connection

## 1- How Developer Handle the API ?

- Public information
- Debug

### How to build a query

The best way to build a query and test if it's valid and will return matched Tweets is to first try it at [twitter.com/search](https://twitter.com/search). As you get a satisfactory result set, the URL loaded in the browser will contain the proper query syntax that can be reused in the API endpoint. Here's an example:

1. We want to search for Tweets referencing @twitterapi account. First, we run the search on [twitter.com/search](https://twitter.com/search)
2. Check and copy the URL loaded. In this case, we got: <https://twitter.com/search?q=%40twitterapi>
3. Replace "https://twitter.com/search" with "https://api.twitter.com/1.1/search/tweets.json" and you will get:  
<https://api.twitter.com/1.1/search/tweets.json?q=%40twitterapi>
4. Execute this URL to do the search in the API

Please note that the API requires that the request be authenticated (OAuth). Authentication & Authorization documentation

**(Go to the twitter API it will find explicit version the API 1.1 ( Again u can alot of public information and developer documentation )**

Another Example is Tool

it is Platform like YouTube

**curl -I https://vimeo.com**

- to grab header viemo

He will get Header request lets try the api

**curl -I https://api.vimeo.com**

he will see content-type: application/vnd.vimeo.error.json

```
→ ~ curl -I https://api.vimeo.com
HTTP/1.1 401 Authorization Required
Server: nginx
Content-Type: application/vnd.vimeo.error+json
Cache-Control: no-cache, max-age=315360000
Strict-Transport-Security: max-age=15552000; includeSubDomains; preload
WWW-Authenticate: Bearer error="invalid_token"
Expires: Tue, 14 Sep 2027 12:37:12 GMT
Via: 1.1 varnish
Fastly-Debug-Digest: 1c6479ffc327308c8d48415d89634de85ebb3b0ced300f4207004c6341ec607
Content-Length: 71
Accept-Ranges: bytes
Date: Sat, 16 Sep 2017 12:37:12 GMT
Via: 1.1 varnish
Connection: keep-alive
X-Served-By: cache-iad2145-IAD, cache-hhn1531-HHN
X-Cache: MISS, MISS
X-Cache-Hits: 0, 0
X-Timer: S1505565433.784277,V50,V136
Vary: Accept,Vimeo-Client-Id,Accept-Encoding
```

3. Replace "https://twitter.com/search" with "https://api.twitter.com/1.1/search/tweets.json" and you will get:  
<https://api.twitter.com/1.1/search/tweets.json?q=%40twitterapi>

Execute this URL to do the search in the API

The query can have operators that modify its behavior: then

Operator	Finds Tweets...
matching now	containing both "watche

if he remember how the API version can be in header itself this is good example

- error = is authentication failure

but it is indicator vemio uses the header to define which API version use.

## 2- Debug it yourself

( To find how clients tell backend server which version to use )

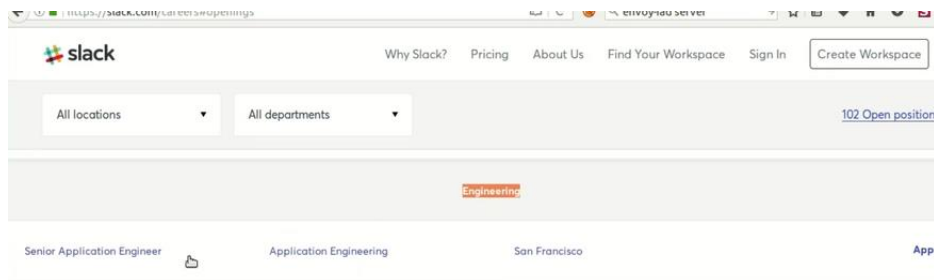
- Let's look about how to discover the programming Language?
  - o Public information (Company jobs /LinkedIn)
  - o Social Engineering
  - o Server Header (Server/X-Powered-By )

To look at the public information resource company jobs and Linked

Company jobs it refer a lot things Language used in the company so if company of jobs php developer or ruby rails developer so u can clearly Company used php r ruby

I think Slack good example :

slack.com/careers



Look at the engineering

(PHP /HACK it is version of php ) and Mysql is the backend and Linux Another Good Source it is Social Engineering LinkedIn Profile if u

another think is company engineering block and infrastructure if use most of company Social Engineering it is help information )

- **Server Headers**

Example

- **curl -I https://www.souq.com**

(This is indicate the website using php (Apache it is general service of php)

{ "API" : "SECURITY" }

```
→ - curl -I https://apl.souq.com
HTTP/1.1 404 Not Found
Content-Type: application/json
Server: Apache
Status: 404 Not Found
Accept-Charset: utf-8
Cache-Control: max-age=0
Expires: Sat, 16 Sep 2017 12:43:01 GMT
Access-Control-Allow-Headers: AUTHORIZATION
Access-Control-Allow-Methods: POST,GET
Date: Sat, 16 Sep 2017 12:43:01 GMT
Content-Length: 0
Connection: keep-alive
```

4

### Debug API: Using Proxy

- To Debug API using proxy to intercept traffic between the client and backend server.

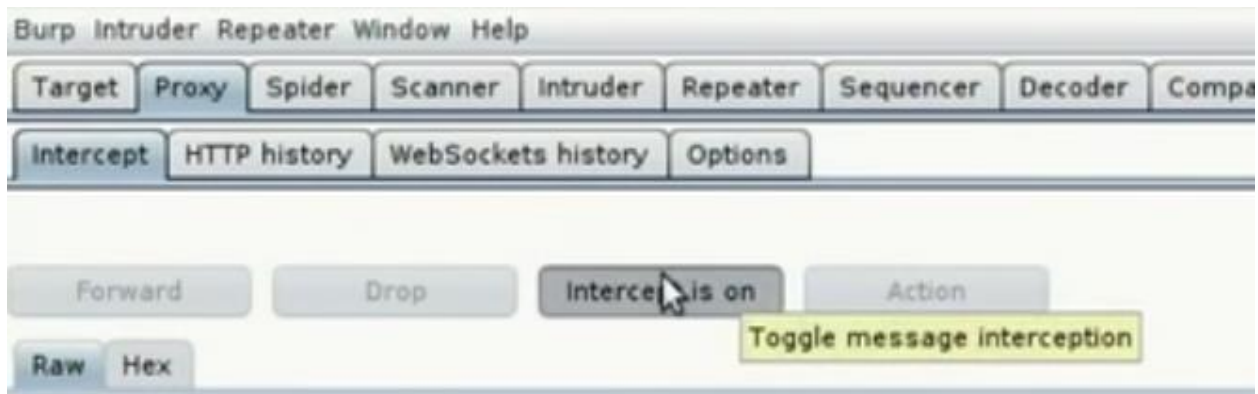
#### Tool

- 1- Burp suite
- 2- Fiddler
- 3- OWSAP proxy

#### Plug-in tool for multiple switch between proxies

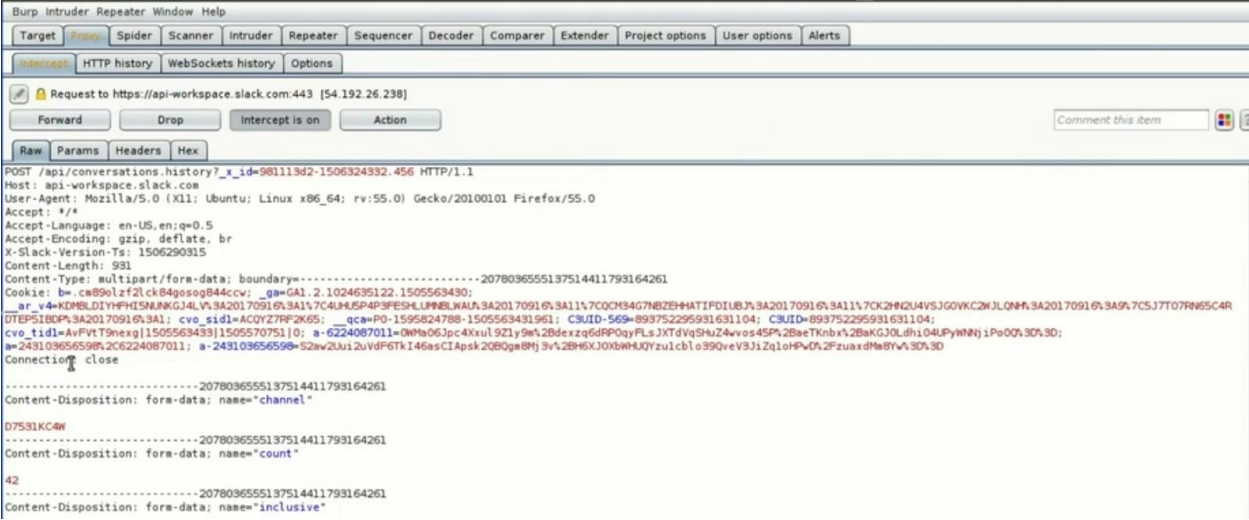
- 1- FoxyProxy

(Edit selection >> Proxy details >> Manual proxy >> “ Local ip “ >> back and switch tap from select Mode )



intercept is on >> ( intercept all traffic between the browser and any website connection.) >> click any site >>

{ “API” : “SECURITY” }



Request to : https: .....

Parameter

Raw	Params	Headers	Hex
POST request to /api/conversations.history			
Type	Name	Value	
URL	_x_id	981113d21506324332456	
Cookie	b	.cm90zld2lck84gosog844ccw	
Cookie	_ga	GA1.2.1024635122.1505563430	
Cookie	_ar_v4	KDMBLDIYHFH5HUNUKGJ4LV:20170916:1 4UHU5P4P3FESHLMNBLWUW:20170916:11 QCM34G7NBZEHHATFI0UB:20170916:11 K2HN2U4V5JGOVCK2W LQNH:20170916:11	
Cookie	cvo_sid1	ACQYZ7RF2K65	
Cookie	_qca	P0-1595824788-1505563431961	
Cookie	C3UID-569	893752295931631104	
Cookie	C3UID	893752295931631104	
Cookie	cvo_tid1	AuFVT59nag 1505563433 1505570751 0	
Cookie	a-4224087011	0WwA98Pp4XkuuZ2ly9m+dexq6dRPDqyFLsJXTdVqSHuZ4wvos45P+aeTKnxb+aKGJ0ldhi04UPyWNN PoQ0==	
Cookie	a	243103656598,6224087011	
Cookie	a-243103656598	S2aw2UuizVdF6TK46asCIapstk2QBQgm8Mj3v+H6Xj0xbWHUQYzu1cbo39QveV3jZq1oHPwD/zuaxdMrm8Yw==	
Body	channel	D7513K4CW	
Body	count	42	
Body	inclusive	true	
Body	ignore_replies	true	
Body	include_pin_count	true	
Body	token	xoxs-242358467893-243103656598-246074300419-29cb42b109	
Body	visible	1	

Name of parameter you can change any thing forward request

## Let`s modify the request in header

## Let`s play with token

Type	Name	Value
URL	_x_id	981113d2-1506324500.144
Cookie	b	.cm89olz2lck84gosog844ccw
Cookie	_ga	GAI_2.1024635122.1505563430
Cookie	_af_v4	KDMLDIYHFHISUNUNKGj4LV:20170916:1 4UHU5P4P3FESHLMUNBLWAU:20170916:11 QCM34G7NBZEHHATFIDIBJ:20170916:1
Cookie	cvo_sid1	ACQYZ7RF2K65
Cookie	_qca	P0-1595824788-1505563431961
Cookie	C3UID-569	893752295931631104
Cookie	C3UID	893752295931631104
Cookie	cvo_tid1	AvFvtT9nexpj1505563433 1505570751 0
Cookie	a-6224087011	0WMa06jpc4Xxul9Z1y9m+dexzq6dRPOqyFLsjXTdVqShuZ4wvos45P+aeTKnbx+aKGj0Ldhi04UPyWNNjPiPo0Q==
Cookie	a	243103656598,6224087011
Cookie	a-243103656598	S2awZUuiZuVdF6TK46asC1Apsk2QBQgm8Mj3v+H6XjOXbWHUQYzu1cblo39QveV3jIzq1oHPwD/zuaxdMm8Yw==
Body	channel	D73GM6R7S
Body	latest	1505570903.000043
Body	count	1
Body	ignore_replies	true
Body	include_pin_count	true
Body	token	<u>xxx-242354867493-243103656598-246074300419-29cb42b109</u>
Body	visible	

Forward after edit



{ "API" : "SECURITY" }

Original request Edited request Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 35
Connection: close
Access-Control-Allow-Origin: *
Date: Mon, 25 Sep 2017 07:28:45 GMT
Referrer-Policy: no-referrer
Server: Apache
Set-Cookie: a=6224087011; expires=Sat, 25-Sep-2027 07:28:45 GMT; Max-Age=315532800; path=/; domain=.slack.com; secure; httponly
Set-Cookie: a=243103656590; expires=Thu, 01-Jan-1970 00:00:01 GMT; Max-Age=0; path=/; domain=.slack.com; secure; httponly
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-Slack-Backend: h
X-Slack-Req-Id: df40dbda-6bcc-474e-8631-aaa517210083
X-XSS-Protection: 0
X-Cache: Miss from cloudfront
Via: 1.1 09696b72fd824c461b396d99979987a3.cloudfront.net (CloudFront)
X-Amz-Cf-Id: gz6xcXz6ZvLm4ftaPufJSP-5XbwQ05C1_XY_DcKD9wyFL0J_AcfVYQ==

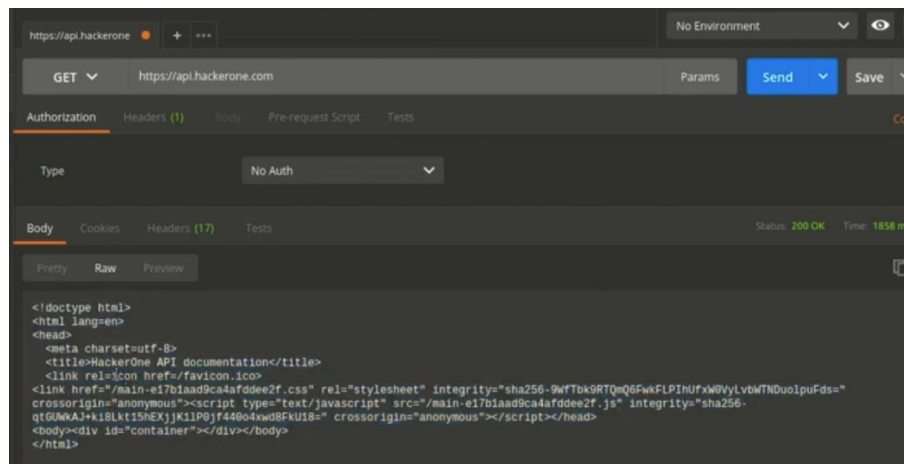
{"ok":false,"error":"invalid_auth"}
```

Send response with invalid\_auth

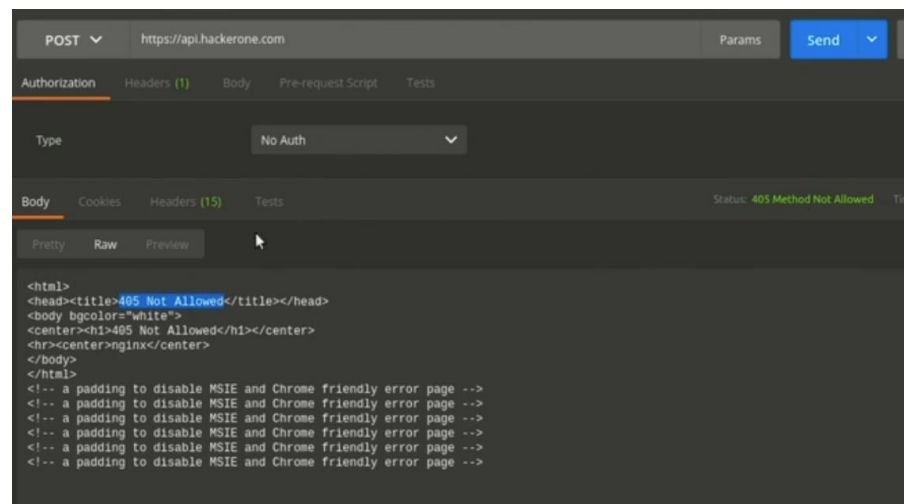
Debug API: API testing tool

Insomnia

If we change GET method to POST method ( what will happen )



POST method



Automating testing: finding implementation bugs using malformed/semi-malformed data injection in an automated fashion.

Meaning: used tools to send data to backend server or any thing to test with multiple data or malformed data or semi-malformed or huge data any an expected. To see what happen or the server react to this data to find an exception or an expected behavior for application to this URL and debug it.

Fuzzing:

- 1- Burp suite
- 2- Acunetix
- 3- Peach fuzzer
- 4- Fuzz-get ( The fuzzer try to send multiple request with characters to the server. And see what HTTP status will be return

Test for website but not for local test

Fuzz-get <http://localhost:9090/search/query>



```
HTTP200:/sear9Dch/
HTTP200:/yre7QEDHEuq/hcraes/
HTTP200:/search/queryrqWRU
HTTP200:/searGKch/query
HTTP200:/seakfMMKb17Kffrch/query
HTTP200:/search
HTTP200:/sea/query
HTTP200:/yreuq/hcraes/
HTTP200:/sMZqxAdAearch/query
HTTP200:/seaqoJ0N2huery
HTTP200:/sea
HTTP200:/search/quer
HTTP500:/search/qy
HTTP200:/search/jlq8q
HTTP200:/search/qu
HTTP200:/yreuq/hcraes/
HTTP200:/searQyvC6voVbmlzNch/query
HTTP200:/suery
HTTP200:/Ensea
```

Fuzz-get <http://localhost:9090/search/query> | grep 'HTTP500'



## Intro to Authentication

Authentication is the process to identify the client so the clients tell the backend server its id or password \*api key and the server try to make sure this information is right. And return success code

It can be

- 1- Token
- 2- Cookie session

This process only to define who is client try to connect the API backend server.

Authorization is define to permission of the connect to client so the client already connected and the authenticated but we need to check if he has authority to permission to do this operation.

Example

- 1- Authentication : login Facebook
- 2- Authorization : modify in page Facebook

Most of used API some times in the same request but in sometimes in multiple request.

( **First request to authentication and second for authorized and exchange token** )

## Authentication method

