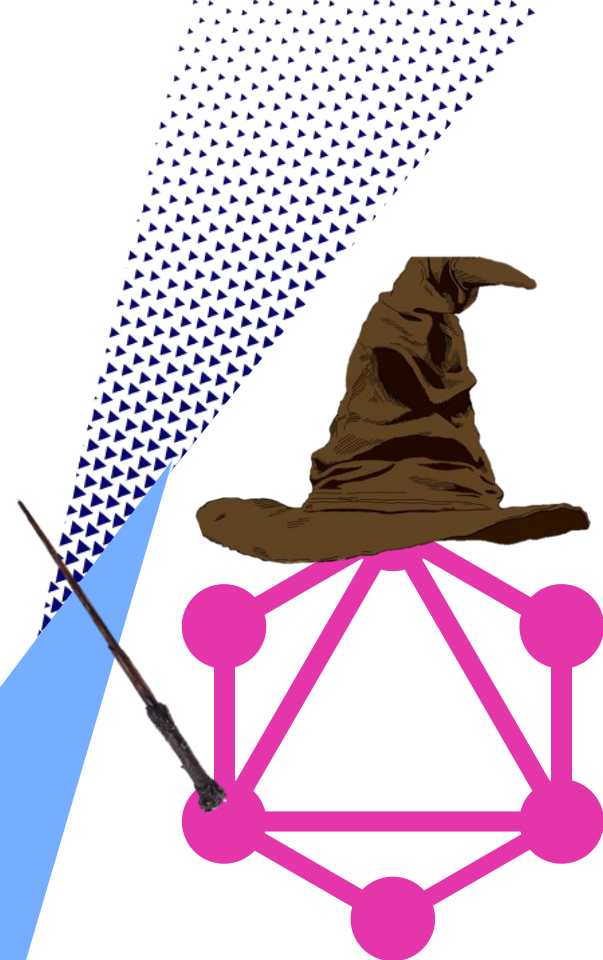




Cyber Security + Customer Experience

Fantastic GraphQL Bugs and Where To Find Them



Thank You to Our Sponsors and Hosts!



OWASP
**NEW
ZEALAND**
owasp.org.nz



QUANTUM
SECURITY



CyberCX

DATACOM



snyk



Auth0

Checkmarx



HCL AppScan

kordia



**LATERAL
SECURITY**



**MICRO
FOCUS**



Pulse Security
www.pulsesecurity.co.nz



RedShield



Flux

SEQA
Information Security



Cobalt



LACEWORK



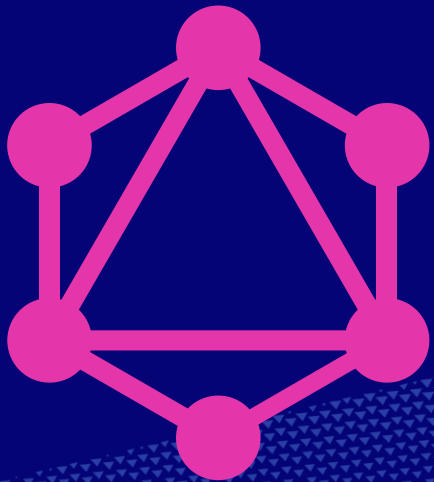
SecureFlag

Without them, OWASP New Zealand Day couldn't happen

whoami

- ▶ Security Consultant @ CyberCX
- ▶ Been working in the industry for 4 years

What is GraphQL



- ▶ GraphQL is a query language for APIs
- ▶ Allows you to query data as well as perform CRUD operations via a single endpoint
- ▶ Schema based
- ▶ GraphQL is designed for efficiency, you retrieve only what you need, nothing more nothing less

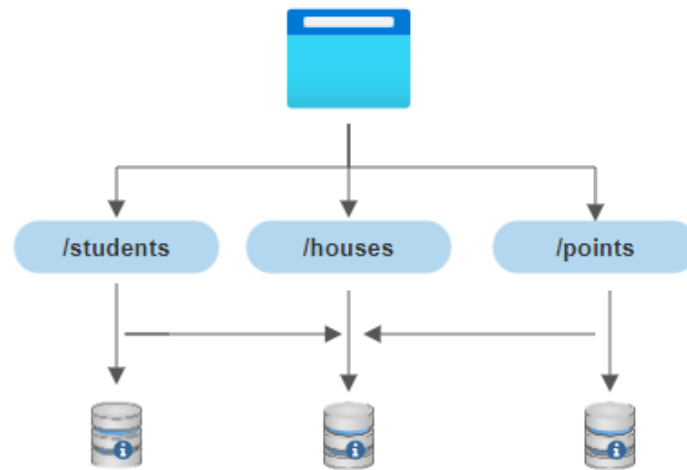
```
query getBurger {  
  burger {  
    bun  
    patty  
    bun  
    lettuce  
  }  
}
```



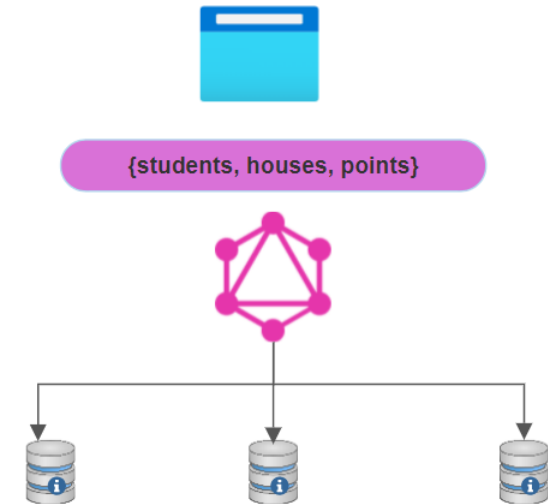
<https://apievangelist.com/2018/06/29/graphql-and-rest-differences-explained-with-burgers/>

REST vs GraphQL

REST



GraphQL



Queries and Mutations

▷ Query = GET

```
1 query {  
2   allPosts {  
3     edges {  
4       node {  
5         title  
6         body  
7         users  
8         { username }  
9       }  
10    }  
11  }  
12 }
```

```
{  
  "data": {  
    "allPosts": {  
      "edges": [  
        {  
          "node": {  
            "title": "Yer a wizard Harry!",  
            "body": "Yer gonna fly a broom, learn spells n shit. And yer gonna enjoy it",  
            "users": {  
              "username": "iliekdragons"  
            }  
          }  
        },  
        {  
          "node": {  
            "title": "HARRY DID YA PUT YA NAME IN DA GOBLET OF FIYAAH",  
            "body": "said Dumbledore calmly",  
            "users": {  
              "username": "dumbledore"  
            }  
          }  
        },  
        {  
          "node": {  
            "title": "Pottah!",  
            "body": "My father will hear about this",  
            "users": {  
              "username": "draco malfoy"  
            }  
          }  
        }  
      ]  
    }  
  }  
}
```

▷ Mutation = POST, DELETE

```
1 mutation {  
2   userMutation(id:1, password:"AlasEarWax!") {  
3     id  
4     password  
5     updated_at  
6   }  
7 }
```

```
{  
  "data": {  
    "userMutation": {  
      "id": 1,  
      "password": "AlasEarWax!",  
      "updated_at": "2022-03-02 23:02:22"  
    }  
  }  
}
```

Accio Vulnerability

▶ IDOR

▶ SQLi

▶ Broken Access Control

▶ DoS

Tools?

- ▶ Burp Plugins – inQL and GraphQL Raider – helps formatting GraphQL in Burp
- ▶ GraphQL Voyager – Mapping out the schema
- ▶ Insomnia[.]rest – API Client
- ▶ GraphQL 😏

Introspection

- Allows you to retrieve the API schema

```
1 query MyQuery {  
2   __schema {  
3     types {  
4       name  
5       fields {  
6         name  
7       }  
8     }  
9   }  
10 }  
11 }
```

```
{  
  "data": {  
    "__schema": {  
      "types": [  
        {  
          "name": "Query",  
          "fields": [  
            {  
              "name": "node"  
            },  
            {  
              "name": "allPosts"  
            },  
            {  
              "name": "allUsers"  
            }  
          ]  
        },  
        {  
          "name": "Node",  
          "fields": [  
            {  
              "name": "id"  
            }  
          ]  
        }  
      ]  
    }  
  }  
}
```

Look for:

- ▷ /v1/explorer
- ▷ /v1/graphql
- ▷ /graph
- ▷ /graphql
- ▷ /graphql/console/
- ▷ /graphql.php
- ▷ /graphql
- ▷ /graphql.php



WHAT IF I TOLD YOU

ITS NOT A BUG, ITS A FEATURE

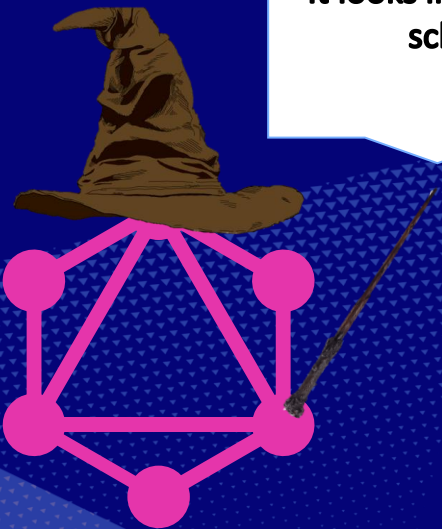
Introspection

▷ Introspection Query

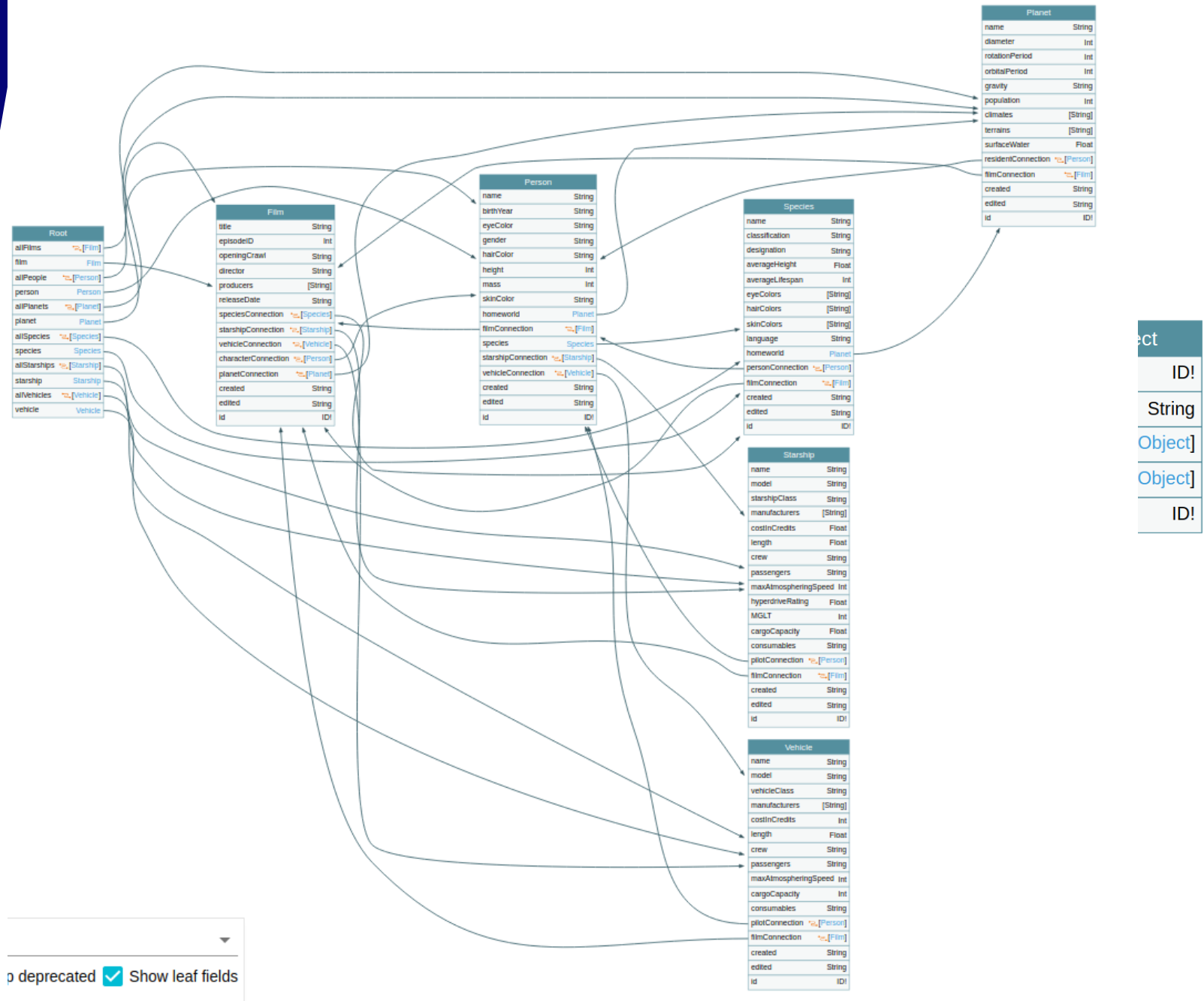
```
{__schema{queryType{name}mutationType{name}subscriptionType{name}types{...FullType}directives{name
description locations args{...InputValue}}}}fragment FullType on __Type{kind name description
fields(includeDeprecated:true){name description args{...InputValue}type{...TypeRef}isDeprecated
deprecationReason}inputFields{...InputValue}interfaces{...TypeRef}enumV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

It looks like you want request the
schema for this API.
Here you go 😊

```
{
  "data": {
    "__schema": {
      "queryType": {
        "name": "Query"
      },
      "mutationType": null,
      "subscriptionType": null,
      "types": [
        {
          "kind": "OBJECT",
          "name": "Query",
          "description": null,
          "fields": [
            {
              "name": "node",
              "description": "The ID of the object",
              "args": [
                {
                  "name": "id",
                  "description": null,
                  "type": {
                    "kind": "NON_NULL",
                    "name": null,
                    "ofType": {
                      "kind": "SCALAR",
                      "name": "ID",
                      "ofType": null
                    }
                  }
                }
              ]
            }
          ]
        }
      ]
    }
  }
}
```



So where are the graphs?



DoS

- ▶ As queries can be very complex - you can nest them and call them recursively
- ▶ E.g In a blog, you will have Users and Posts. A user can have multiple posts and each post will have a user.
- ▶ We can call this recursively.

```
Query  Variables  Injection Points
1 {
2   allUsers {
3     edges {
4       node {
5         username
6         posts {
7           edges {
8             node {
9               title
10              authorId
11              users {
12                username
13                posts {
14                  edges {
15                    node {
16                      title
17                      body
18                      users {
19                        username
20                        uuid
21                        username
22                        uuid
23                        posts {
24                          edges {
25                            node {
26                              title
27                              body
28                              users {
29                                username
30                                posts {
31                                  edges {
32                                    node {
33                                      title
34                                      body
```

DoS

```
{
  "data":{
    "allPosts":{
      "edges":[
        {
          "node":{
            "title":"Yer a wizard Harry!",
            "body":"Yer gonna fly a broom, learn spells n shit. And yer gonna enjoy it",
            "users":{
              "username":"iliekdiragons"
            }
          }
        },
        {
          "node":{
            "title":"HARRY DID YA PUT YA NAME IN DA GOBLET OF FIYAHH",
            "body":"said Dumbledore calmly",
            "users":{
              "username":"dumbledore"
            }
          }
        },
        {
          "node":{
            "title":"Pottah!",
            "body":"My father will hear about this",
            "users":{
              "username":"draco malfoy"
            }
          }
        }
      ]
    }
  }
}
```

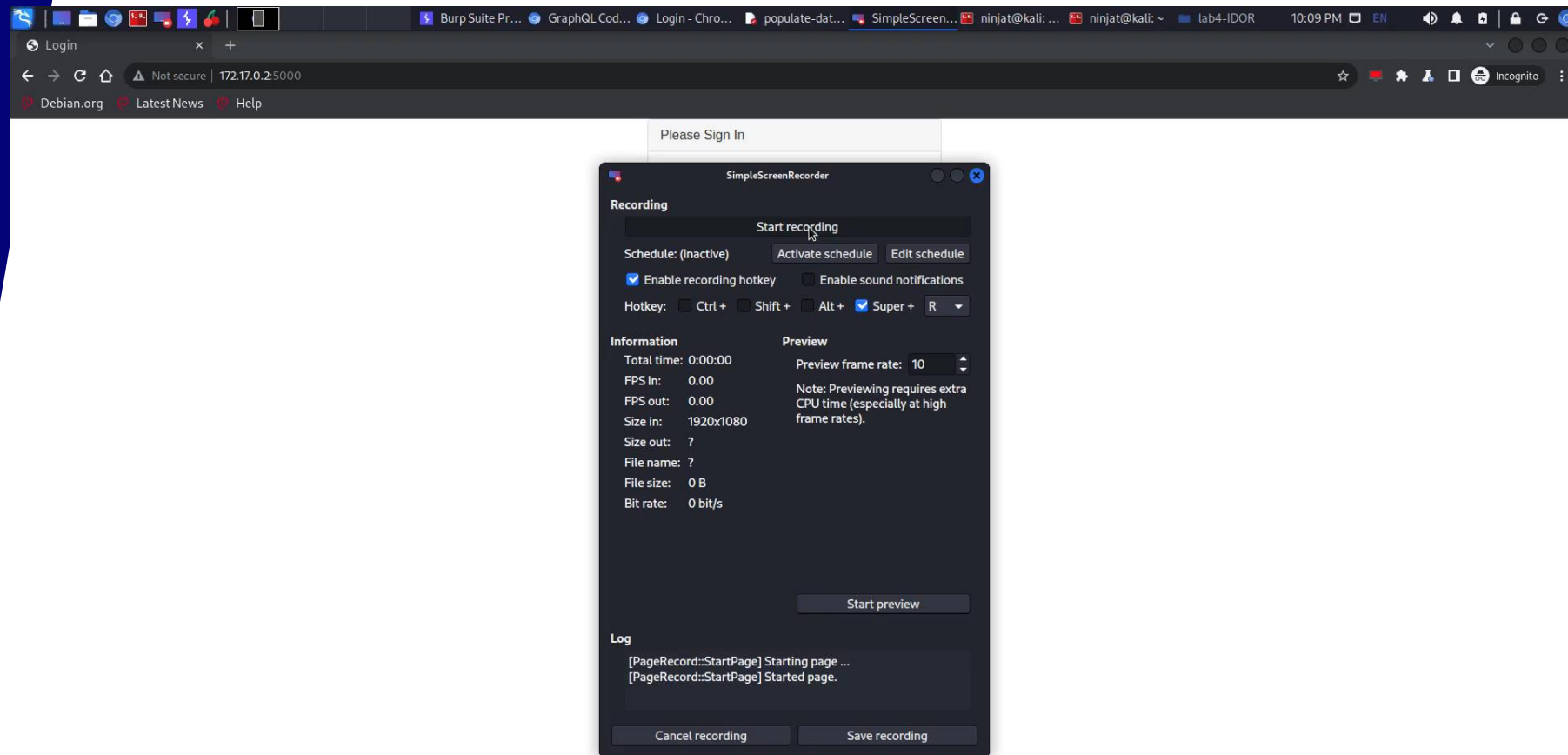
```
7 {  
  "data":{  
    "allUsers":{  
      "edges":[  
        {  
          "node":{  
            "username":"iliekdragons",  
            "posts":{  
              "edges":[  
                {  
                  "node":{  
                    "title":"Yer a wizard Harry!",  
                    "authorId":1,  
                    "users":{  
                      "username":"iliekdragons",  
                      "posts":{  
                        "edges":[  
                          {  
                            "node":{  
                              "title":"Yer a wizard Harry!",  
                              "body":  
                                "Yer gonna fly a broom, learn spells n shit. And yer gonna enjoy it  
                                ",  
                              "users":{  
                                "username":"iliekdragons",  
                                "uuid":"1",  
                                "posts":{  
                                  "edges":[  
                                    {  
                                      "node":{  
                                        "title":"Yer a wizard Harry!",  
                                        "body":  
                                          "Yer gonna fly a broom, learn spells n shit. And yer gonn  
                                          a enjoy it",  
                                        "users":{  
                                          "username":"iliekdragons",  
                                          "posts":{  
                                            "edges":[  
                                              {
```

0 matches
7,462 bytes | 127 millis

DoS Mitigation?

- ▶ Limit Maximum Query Depth
 - ▶ Throttling Based on Server Time or Query Complexity
 - ▶ Audit your query before production
-
- ▶ <https://www.npmjs.com/package/graphql-validation-complexity>
 - ▶ <https://github.com/4Catalyzer/graphql-validation-complexity>
 - ▶ <https://github.com/slicknode/graphql-query-complexity>

IDOR



IDOR Mitigation?

- ▶ Use GUIDs instead of IDs
 - Increases complexity
- ▶ Implement Role Based Access Controls (RBACs)
 - Validate that the requested is authorised to perform the action

def(dev)eu

def(dev)eu GraphQL Blog

Create New Post

A Twelve Year Old Destroyed My Diary And

voldy

Lorum Ipsum AVADA KEDAVARA

Using the Nokia 1100 as a Horcrux???

voldy

In this post I will be attempting to...

Ngarles vs Wackspruts

loonylovegood

You're just as sane as I am

SimpleScreenRecorder

Recording

Start recording

Schedule: (inactive)

Activate schedule

Edit schedule

☒ Enable recording hotkey ☐ Enable sound notifications

Hotkey: ☐ Ctrl + ☐ Shift + ☐ Alt + ☒ Super +

R

Information

Total time: 0:00:00

FPS in: 0.00

FPS out: 0.00

Size in: 1920x1080

Size out: ?

File name: ?

File size: 0 B

Bit rate: 0 bit/s

Preview

Preview frame rate: 10

Note: Previewing requires extra CPU time (especially at high frame rates).

Start preview

Log

[PageRecord::StartPage] Starting page ...

[PageRecord::StartPage] Started page.

Cancel recording

Save recording

Injection Mitigation?

- ▶ Input Validation
 - Whitelist approach
- ▶ Use parameterised queries

Examples

- ▶ Testing a crowdfunding style app
- ▶ Access controls weren't applied to GraphQL endpoint
- ▶ Allowed **unauthed** payments to funding requests, remove funding from one project to another and resetting user passwords
- ▶ Create your own project -> Move funding to your own project -> \$\$\$\$



Q&A

