

# GraphQL Shield

*CWE-aware defense in depth for GraphQL APIs in Go*

14

CWEs

7

attacks

3

layers

1

line

GraphQLConf  
2026

hosted by



GraphQL  
Foundation



**Ravi Sastry Kadali**

AI/ML Engineer | Autonomous Vehicles  
@General Motors

Go ecosystem contributor

Open Source · Security

<https://www.linkedin.com/in/ravisastryk/>

<https://github.com/ravisastryk/graphqlshield>

## THE PROBLEM

Every GraphQL request  
looks like this to your WAF:

```
POST /graphql HTTP/1.1
Content-Type: application/json

{ "query": "..."
```

**WAF**

blind here

*But hiding inside that opaque JSON body:*

**CWE-89**

**SQL Injection**

```
filter: "1' OR '1'='1"
```

**CWE-400**

**Depth-of-50 DoS**

```
user.posts.comments...
```

**CWE-200**

**Introspection leak**

```
{ __schema { types {..} } }
```

**CWE-79**

**XSS in mutation**

```
body: "<script>..."
```

# 14 vulnerability classes across 3 exploitation layers

## Static analysis

**CWE-400**

Resource exhaustion (depth DoS)

**CWE-200**

Sensitive data exposure (introspection)

**CWE-200**

Sensitive field exposure

## Runtime sanitization

**CWE-89**

SQL Injection

**CWE-79**

Cross-Site Scripting

**CWE-78**

Command Injection

**CWE-22**

Path Traversal

**CWE-943**

NoSQL Injection

**CWE-601**

Open Redirect

## Resolver audit (CI)

**CWE-327**

Broken cryptographic algorithm

**CWE-328**

Weak hash (MD5, SHA-1)

**CWE-798**

Hardcoded credentials

**CWE-321**

Hardcoded cryptographic key

**CWE-338**

Insecure pseudo-random number

**CWE-862**

Missing authorization check

Three layers. Zero config for the runtime checks.

## Layer 1 — Static controls

stdlib · no deps

- Reject if query depth > MaxDepth(n) → CWE-400
- Block \_\_schema / \_\_type introspection → CWE-200
- Block named sensitive fields (password, ssn, ...) → CWE-200

▼ pass

## Layer 2 — Runtime variable sanitization

go-safeinput

- SQL injection patterns → CWE-89
- XSS / HTML injection → CWE-79 · Command injection → CWE-78
- Path traversal → CWE-22 · NoSQL operator injection → CWE-943

▼ pass

## Layer 3 — Resolver source audit (run in CI)

cryptoguard-go

- MD5 / SHA-1 / DES / RC4 usage → CWE-327/328
- Hardcoded passwords, secrets, API keys → CWE-798/321
- math/rand instead of crypto/rand → CWE-338 · missing AuthZ → CWE-862

✓ gqlgen resolver executes

# The entire integration is one new line.

## BEFORE

```
gqlHandler := handler.NewDefaultServer(schema)

http.Handle("/graphql", gqlHandler)
```

```
$ go get github.com/ravisastryk/graphqlshield@latest
```

Requires Go 1.26+

## AFTER

```
gqlHandler := handler.NewDefaultServer(schema)

s := shield.New(
    shield.WithMaxDepth(5),
    shield.WithBlockIntrospection(),
    shield.WithBlockSensitiveFields(
        "password", "ssn"),
)

http.Handle("/graphql", s.Wrap(gqlHandler))
```

← *this is the only new line*



# Demo

*8 vectors · real gqngen server · sub-millisecond response*

## 1 SQL Injection

CWE-89

400

## 2 XSS payload

CWE-79

400

## 3 Command injection

CWE-78

400

## 4 Path traversal

CWE-22

400

## 5 NoSQL injection

CWE-943

400

## 6 Depth DoS (depth 12)

CWE-400

400

## 7 Introspection leak

CWE-200

403

## 8 Legitimate register

clean

200

```
-zsh
-zsh #1 -zsh #2 -zsh #3 -zsh #4 -zsh #5 -zsh #6
-x -zsh
Sastry/src/github/ravisastryk/graphqlshield on main [?] via v1.27-devel_799a87862b
> cd examples/gqlgen-demo && go mod tidy && go run server.go

-x -zsh
Sastry/src/github/ravisastryk/graphqlshield on main [?] via v1.27-devel_799a87862b
> []
```

# The Go toolchain has a GraphQL-shaped hole.

Tool	Depth DoS	Injection	Introspect.	Secrets	GraphQL-aware
GraphQL Go(graphql-go)	✗	✗	✗	✗	✓
gqlgen	✓	✗	✗	✗	✓
gosec + go-safeinput	✗	✓	✗	✓	✗
WAF / API Gateway	✗	~	✗	✗	✗
<b>GraphQLShield ✨</b>	✓	✓	✓	✓	✓

~ = partial / HTTP-layer only

Credits: GraphQL · gqlgen · go-safeinput · cryptoguard-go · gosec · Go community ❤️

GET STARTED

# One command. Full coverage.

```
$ go get github.com/ravisastryk/graphqlshield@latest
```



Star the repo

[github.com/ravisastryk/graphqlshield](https://github.com/ravisastryk/graphqlshield)



Read the docs

[pkg.go.dev/  
github.com/ravisastryk/graphqlshield](https://pkg.go.dev/github.com/ravisastryk/graphqlshield)



Run attacks.sh

See 7 attack vectors blocked in < 1s



Contribute

PRs welcome!!!  
new CWE rules especially

*Standing on the shoulders of giants:*

GraphQL

gqlgen

go-safeinput

cryptoguard-go

gosec

Go community ❤️

github.com/99designs/gqlgen/pull/4021

99designs / gqlgen

Code Issues 357 Pull requests 31 Agents Discussions Actions Projects Security and quality Insights

## federation: apply OBJECT-level directives to entity resolvers #4021

Merged StevenACoffman merged 8 commits into 99designs:master from ravisastryk:fix/federation-entity-directives on Feb 16

Conversation 14 Commits 8 Checks 18 Files changed 21 +10,156 -1,488

ravisastryk commented on Feb 9 · edited

Fixes #3910: Federation entity resolvers now correctly execute OBJECT-level directives.

When using Apollo Federation, OBJECT-level directives (example:- @auth , @guard ) applied to federated entities were not being executed on entity resolvers. Regular queries worked fine, but `__entities` queries bypassed these directives entirely.

Previously, directives declared on a type such as:

```
type Person @key(fields: "id") @guard(name: "PersonGuard")
```

were enforced for `Query.getPerson` but silently skipped when the Apollo Router resolved the entity through federation's `__entities` query.

### High Level Changes

- Added `ImplDirectives` field to `Entity` struct
- Populated directives during code generation
  - Filters out federation-internal directives (`@key`, `@requires`, etc.)
  - Preserves user-defined OBJECT-level directives
- Applied directive chains in entity resolver template
  - Wraps entity resolver calls with directive middleware
  - Maintains same behavior as regular query resolvers

Reviewers: StevenACoffman

Assignees: No one assigned

Labels: None yet

Projects: None yet

Milestone: No milestone

Development: Successfully merging this pull request may close these issues.

Directives not applied to entity resolvers in Feder...

Notifications: [Customize](#)

“

Your GraphQL API deserves  
a shield, not wishful thinking. ”

---

```
go get github.com/ravisastryk/graphqlshield@latest
```

---

