# Cache middleware default key generator ignores query string, causing response mix-up across distinct query parameters

Moderate **ReneWerner87** published **GHSA-35hp-hqmv-8qg8** last week

**Package**

**github.com/gofiber/fiber/v3** (Go)

| Affected versions | Patched versions |
| --- | --- |
| <=3.1.0 | >3.1.0 |

## Description

### Summary

Fiber cache middleware's default key generator uses only `c.Path()` and does not include the query string.
As a result, requests like `/?id=1` and `/?id=2` can map to the same cache key and share the same cached response.

This can cause response mix-up (cache poisoning-like behavior) for endpoints where response content depends on query parameters.

### Details

Default configuration in cache middleware:

- `KeyGenerator: func(c fiber.Ctx) string { return utils.CopyString(c.Path()) }`

References:

- https://github.com/gofiber/fiber/blob/main/middleware/cache/config.go#L90-L92
- https://github.com/gofiber/fiber/blob/main/middleware/cache/cache_test.go#L599-L621

The existing test demonstrates that when handler output depends on query parameter `id`, a second request with a different query still returns the first cached response (cache hit), confirming query is not part of the default cache key.

## PoC

Minimal PoC:

```
package main

import (
    "log"

    "github.com/gofiber/fiber/v3"
    "github.com/gofiber/fiber/v3/middleware/cache"
)

func main() {
    app := fiber.New()
    app.Use(cache.New()) // default config

    app.Get("/", func(c fiber.Ctx) error {
        return c.SendString(c.Query("id", "1"))
    })

    log.Fatal(app.Listen(":3000"))
}
```

Reproduction:

1. `GET /?id=1`
   - Cache miss
   - Response body: `1`
2. `GET /?id=2`
   - Cache hit
   - Response body: `1` (expected `2`)

Local verification command used:

```
go test ./middleware/cache -run Test_Cache_WithNoCacheRequestDirective -count=1
```

Observed result: test passes, confirming this is current behavior.

## Impact

- Responses that should vary by query parameters can be mixed between requests.

- In real deployments, this may leak or corrupt user/tenant-specific content if query parameters influence context or data selection.
- This is deployment-dependent but security-relevant, and not safe-by-default for query-variant responses.

## Suggested remediation

- Change default cache key generation to include path + normalized query string (or canonicalized original URL).
- Keep ability for custom key generators.
- Add explicit documentation warning that path-only keying is unsafe for query-dependent responses.

### Severity

Moderate 6.5 / 10

**CVSS v3 base metrics**

| | |
|---|---|
| Attack vector | Network |
| Attack complexity | Low |
| Privileges required | None |
| User interaction | None |
| Scope | Unchanged |
| Confidentiality | Low |
| Integrity | Low |
| Availability | None |

Learn more about base metrics

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N

### CVE ID

CVE-2026-30246

### Weaknesses

- CWE-200
- CWE-524

### Credits

 **xeloxa** Reporter

 **gaby** Remediation developer

**ReneWerner87**

Remediation reviewer