OTLP HTTP exporters read unbounded HTTP response bodies

Moderate **dashpole** published **GHSA-w8rr-5gcm-pp58** 7 hours ago

Package

go.opentelemetry.io/otel/exporters/otlp/otlplog/otlploghttp (Go)

Affected versions	Patched versions
< v0.19.0	v0.19.0

go.opentelemetry.io/otel/exporters/otlp/otlpmetric/otlpmetrichttp (Go)

< v1.43.0	v1.43.0

go.opentelemetry.io/otel/exporters/otlp/otlptrace/otlptracehttp (Go)

< v1.43.0	v1.43.0

Description

overview:
this report shows that the otlp HTTP exporters (traces/metrics/logs) read the full HTTP response body into an in-memory `bytes.Buffer` without a size cap.

this is exploitable for memory exhaustion when the configured collector endpoint is attacker-controlled (or a network attacker can mitm the exporter connection).

severity

HIGH

not claiming: this is a remote dos against every default deployment.
claiming: if the exporter sends traces to an untrusted collector endpoint (or over a network segment where mitm is realistic), that endpoint can crash the process via a large response body.

callsite (pinned):

- exporters/otlp/otlptrace/otlptracehttp/client.go:199

- exporters/otlp/otlptrace/otlptracehttp/client.go:230
- exporters/otlp/otlpmetric/otlpmetrichttp/client.go:170
- exporters/otlp/otlpmetric/otlpmetrichttp/client.go:201
- exporters/otlp/otlplog/otlploghttp/client.go:190
- exporters/otlp/otlplog/otlploghttp/client.go:221

permalinks (pinned):

- **opentelemetry-go/exporters/otlp/otlptrace/otlptracehttp/client.go**
  Line 199 in `248da95`

```
199     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

- **opentelemetry-go/exporters/otlp/otlptrace/otlptracehttp/client.go**
  Line 230 in `248da95`

```
230     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

- **opentelemetry-go/exporters/otlp/otlpmetric/otlpmetrichttp/client.go**
  Line 170 in `248da95`

```
170     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

- **opentelemetry-go/exporters/otlp/otlpmetric/otlpmetrichttp/client.go**
  Line 201 in `248da95`

```
201     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

- **opentelemetry-go/exporters/otlp/otlplog/otlploghttp/client.go**
  Line 190 in `248da95`

```
190     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

- **opentelemetry-go/exporters/otlp/otlplog/otlploghttp/client.go**
  Line 221 in `248da95`

```
221     if _, err := io.Copy(&respData, resp.Body); err != nil {
```

root cause:
each exporter client reads `resp.Body` using `io.Copy(&respData, resp.Body)` into a `bytes.Buffer` on both success and error paths, with no upper bound.

impact:
a malicious collector can force large transient heap allocations during export (peak memory scales with attacker-chosen response size) and can potentially crash the instrumented process (oom).

affected component:

- go.opentelemetry.io/otel/exporters/otlp/otlptrace/otlptracehttp
- go.opentelemetry.io/otel/exporters/otlp/otlpmetric/otlpmetrichttp
- go.opentelemetry.io/otel/exporters/otlp/otlplog/otlploghttp

repro (local-only):

```
unzip poc.zip -d poc
cd poc
make canonical resp_bytes=33554432 chunk_delay_ms=0
```

expected output contains:

```
[CALLSITE_HIT]: otlptracehttp.UploadTraces::io.Copy(resp.Body)
[PROOF_MARKER]: resp_bytes=33554432 peak_alloc_bytes=118050512
```

control (same env, patched target):

```
unzip poc.zip -d poc
cd poc
make control resp_bytes=33554432 chunk_delay_ms=0
```

expected control output contains:

```
[CALLSITE_HIT]: otlptracehttp.UploadTraces::io.Copy(resp.Body)
[NC_MARKER]: resp_bytes=33554432 peak_alloc_bytes=512232
```

attachments: poc.zip (attached)

PR_DESCRIPTION.md

attack_scenario.md

poc.zip

Fixed in: #8108

Severity

Moderate 5.3 / 10

CVSS v3 base metrics

Attack vector	Adjacent
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Unchanged
Confidentiality	None

Integrity	None
Availability	High

Learn more about base metrics

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

CVE ID

CVE-2026-39882

Weaknesses

▶ CWE-789

Credits

1seal Reporter

pellared Remediation developer