Debug Report

`example-crash` — 2026-02-28

delve-helper

Contents

1 Debug Report — example-crash — 2026-02-28

1.1 Hypothesis

Suspected location: `pipeline.go:11`

Expected: nil readings skipped without dereference

Actual: panic: nil pointer dereference when r is nil

1.2 Debugging Trace

#	Action	Location	Reasoning
1	set	`pipeline.go:11`	validate entry point to observe r when nil
1	hit	`pipeline.go:11 (bp 1)`	observed r=0 (nil); condition uses OR so r.Sensor will be evaluated
1	verify	`pipeline.go:11 (post-fix)`	nil skipped; go run exits 0, 4 readings

1.3 Breakpoints & Evidence

1.3.1 pipeline.go:11

Source context:

```
9  // validate reports whether a reading should be processed.
10 func validate(r *Reading) bool {
11     return r != nil || r.Sensor != ""
12 }
13
```

Args:

```
r = 0
```

Locals:

```
r = 0, ~r0 = false
```

Stack:

```
#0 main.validate pipeline.go:11
#1 main.process pipeline.go:18
#2 main.main main.go:15
```

Print `r`:

```
0
```

Observation: validate called with r=nil; line 11 evaluates r != nil || r.Sensor != “” so r.Sensor will be dereferenced and panic

Root Cause

pipeline.go:11 — validate uses r != nil || r.Sensor != “”. When r is nil, the second operand r.Sensor is evaluated and causes nil pointer dereference. Must require both non-nil and non-empty: AND.

Fix Applied

Changed validate condition from OR to AND: r != nil && r.Sensor != “”. Nil readings are no longer dereferenced.

1.4 Post-fix Verification

Confirmed: validate uses AND; nil entries skipped. go run → processed 4 readings: [23.4 25.1 22.8 24].