WAL Watching

Bartłomiej Płotka Mar 5, 2026

Goal

- Ensure https://github.com/prometheus/prometheus/pull/18062 is safe for default prod Prometheus (no st-storage flag)

Painpoint: WAL Watcher

Current alloc/s diff is because we re-alloc RefSample slices a lot and RefSample is 8B larger.

github.com/prometheus/prometheus/tsdb/wlog.(*Watcher).readSegment
/go/src/github.com/prometheus/prometheus/tsdb/wlog/watcher.go

```
Total:    985.62GB     1.52TB (flat, cum)  7.99%
   503            .          .                           histogramsToSend      []record.RefHistogramSample
   504            .          .                           floatHistograms       []record.RefFloatHistogramSample
   505            .          .                           floatHistogramsToSend []record.RefFloatHistogramSample
   506            .          .                           metadata              []record.RefMetadata
   507            .          .                   )
   508            .   908.19MB                   for r.Next() && !isClosed(w.quit) {
   509            .          .                           var err error
   510            .          .                           rec := r.Record()
   511            .          .                           w.recordsReadMetric.WithLabelValues(dec.Type(rec).String()).Inc()
   512            .          .
   513            .          .                           switch dec.Type(rec) {
   514            .          .                           case record.Series:
   515            .    53.87GB                                   series, err = dec.Series(rec, series[:0])
   516            .          .                                   if err != nil {
   517            .          .                                           w.recordDecodeFailsMetric.Inc()
   518            .          .                                           return err
   519            .          .                                   }
   520            .     3.02GB                                   w.writer.StoreSeries(series, segmentNum)
   521            .          .
   522            .          .                           case record.Samples:
   523            .          .                                   // If we're not tailing a segment we can ignore any samples records we see.
   524            .          .                                   // This speeds up replay of the WAL by > 10x.
   525            .          .                                   if !tail {
   526            .          .                                           break
   527            .          .                                   }
   528            .   515.64GB                                   samples, err = dec.Samples(rec, samples[:0])
   529            .          .                                   if err != nil {
   530            .          .                                           w.recordDecodeFailsMetric.Inc()
   531            .          .                                           return err
   532            .          .                                   }
   533            .          .                                   for _, s := range samples {
   534            .          .                                           if s.T > w.startTimestamp {
   535            .          .                                                   if !w.sendSamples {
   536            .          .                                                           w.sendSamples = true
   537            .          .                                                           duration := time.Since(w.startTime)
   538            .          .                                                           w.logger.Info("Done replaying WAL", "duration", duration)
   539            .          .                                                   }
   540    985.62GB   985.62GB                                                   samplesToSend = append(samplesToSend, s)
   541            .          .                                           }
   542            .          .                                   }
   543            .          .                                   if len(samplesToSend) > 0 {
   544            .     2.95MB                                           w.writer.Append(samplesToSend)
   545            .          .                                           samplesToSend = samplesToSend[:0]
   546            .          .                                   }
   547            .          .
   548            .          .                           case record.Exemplars:
   549            .          .                                   // Skip if experimental "exemplars over remote write" is not enabled.
```

Benchmark: Prombench Segments (noST, v1 records, RefSampleV1)

Taken from prombench https://github.com/prometheus/prometheus/pull/18062 (st-storage = false)

```shell
NS="prombench-18062"
```

```
POD="prometheus-test-main-7db67bc7cb-ddvb9"
kubectl cp -n ${NS} ${POD}:wal/00000916 \
       ${SCRIPT_DIR}/testdata/main00000916 -c prometheus

POD="prometheus-test-pr-18062-89c9cd459-gxn8t"
kubectl cp -n ${NS} ${POD}:/prometheus/wal/00000916 \
       ${SCRIPT_DIR}/testdata/pr18062-00000916 -c prometheus
```

Segment Stats

```
None
=== RUN   TestInspect
------- ../../../test-infra/x.bwplotka/testdata/main/00000916 -----
SegmentSize: 134217728
seriesStores: 0 (data: 0)
metadataStores: 0 (data: 0)
sampleAppends: 49754 (data: 23323992)
exemplarAppends: 0 (data: 0)Tha
histogramAppends: 0 (data: 0)
floatHistogramsAppends: 0 (data: 0)
activeSeries from segment: 0, activeSeries calculated: 1176282, samplesPerRef: 19.828571720046725
time gap 2026-03-05 08:49:19.747 +0000 UTC - 2026-03-05 08:54:41.131 +0000 UTC 5m21.384s
EmptyRecordsCount: 0, SampleDecodeSizeMax: 67093, SampleDecodeSizeMin: 5046, SampleDecodeSizeSum:
256327800, SampleDecodeCount: 49754, SamplesCount: 23323992, AvgSizePerRecord: 5151.903365,
AvgSamplesPerRecord: 468.786268
------- ../../../test-infra/x.bwplotka/testdata/pr18062/00000916 -----
SegmentSize: 134217728
seriesStores: 0 (data: 0)
metadataStores: 0 (data: 0)
sampleAppends: 50011 (data: 23447393)
exemplarAppends: 0 (data: 0)
histogramAppends: 0 (data: 0)
floatHistogramsAppends: 0 (data: 0)
activeSeries from segment: 0, activeSeries calculated: 1176282, samplesPerRef: 19.933479386745695
time gap 2026-03-05 09:45:43.666 +0000 UTC - 2026-03-05 09:50:51.316 +0000 UTC 5m7.65s
EmptyRecordsCount: 0, SampleDecodeSizeMax: 67334, SampleDecodeSizeMin: 5046, SampleDecodeSizeSum:
257598539, SampleDecodeCount: 50011, SamplesCount: 23447393, AvgSizePerRecord: 5150.837596,
AvgSamplesPerRecord: 468.844714
--- PASS: TestInspect (3.61s)
```

Reading full segment (BenchmarkWatcher_ReadSegment)

```
None
goos: darwin
goarch: arm64
pkg: github.com/prometheus/prometheus/tsdb/wlog
cpu: Apple M4 Pro
                                                                        |    main     |
                                                                        |   sec/op    |
```

```
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2         291.2m ± 4%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2     321.2m ± 3%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2       291.7m ± 3%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2   323.1m ± 6%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2       329.7m ± 2%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2   336.4m ± 1%
geomean                                                                   315.0m

                                                                        |     main     |
                                                                        | readBytes/op |
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2         134.2M ± 0%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2     134.2M ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2       134.2M ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2   134.2M ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2       45.84M ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2   45.84M ± 0%
geomean                                                                   93.82M

                                                                        |    main     |
                                                                        |  reads/op   |
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2         1.000 ± 0%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2     50.01k ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2       1.000 ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2   49.75k ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2       1.000 ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2   10.00k ± 0%
geomean                                                                   170.9

                                                                        |       main       |
                                                                        | sampleAppends/op |
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2            50.01k ± 0%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2        50.01k ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2          49.75k ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2      49.75k ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2          10.00k ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2      10.00k ± 0%
geomean                                                                      29.19k

                                                                        |     main     |
                                                                        |     B/op     |
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2         1.551Mi ± 0%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2     51.94Mi ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2       834.1Ki ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2   53.29Mi ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2       442.6Mi ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2   454.4Mi ± 0%
geomean                                                                   29.82Mi

                                                                        |    main     |
                                                                        |  allocs/op  |
Watcher_ReadSegment/data=pr18062/compression=snappy/case=one-go-2          43.00 ± 2%
Watcher_ReadSegment/data=pr18062/compression=snappy/case=per-scrape-2     105.1k ± 0%
Watcher_ReadSegment/data=main18062/compression=snappy/case=one-go-2        32.00 ± 3%
Watcher_ReadSegment/data=main18062/compression=snappy/case=per-scrape-2   104.6k ± 0%
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=one-go-2       14.00M ± 0%
```

```
Watcher_ReadSegment/data=synth5Rec/compression=snappy/case=per-scrape-2    14.03M ± 0%
geomean
```

How often is the watcher re-reading?

Growth / s

```shell
NS="prombench-18062"
POD="prometheus-test-pr-18062-89c9cd459-gxn8t"
kubectl exec -n ${NS} ${POD} -c prometheus -- ls -l /prometheus/wal/00000926
sleep 5s
kubectl exec -n ${NS} ${POD} -c prometheus -- ls -l /prometheus/wal/00000926
sleep 5s
kubectl exec -n ${NS} ${POD} -c prometheus -- ls -l /prometheus/wal/00000926
sleep 5s
kubectl exec -n ${NS} ${POD} -c prometheus -- ls -l /prometheus/wal/00000926

bash x.bwplotka/cp.bwplotka.sh
-rw-r--r--    1 root     root      99470455 Mar  5 11:13 /prometheus/wal/00000926
-rw-r--r--    1 root     root     102152938 Mar  5 11:13 /prometheus/wal/00000926
-rw-r--r--    1 root     root     104982892 Mar  5 11:13 /prometheus/wal/00000926
-rw-r--r--    1 root     root     107704969 Mar  5 11:13 /prometheus/wal/00000926
```

```none
func TestGrowth(t *testing.T) {
	sizes := [4]int{99470455, 102152938, 104982892, 107704969}
	fmt.Println(float64(sizes[3]-sizes[0]) / 15.0)
	fmt.Println(float64(sizes[1]-sizes[0]) / 5.0)
	fmt.Println(float64(sizes[2]-sizes[1]) / 5.0)
	fmt.Println(float64(sizes[3]-sizes[2]) / 5.0)
}
548967.6
536496.6
565990.8
544415.4
```

We should expect avg 0.5MBs / second in prombench env.

Assuming watcher is notified on every Appender commit and watcher reads 128MB in 284.4ms (0.5 MB in 1126860ns ~= 1ms) it's likely ~every commit triggers separate **readAndHandleError**

main() scrape -> wal -> RW2 (BenchmarkE2EScrapeAndRemoteWriteNoChurn) confirms reads are done on **every second commit** on 1K targets:

None

```
goos: darwin
goarch: arm64
pkg: github.com/prometheus/prometheus/cmd/prometheus
cpu: Apple M4 Pro
                                 |    main    |
                                 |   sec/op   |
E2EScrapeAndRemoteWriteNoChurn-4   1.013 ± 0%

                                 |       main       |
                                 | recv_requests/op |
E2EScrapeAndRemoteWriteNoChurn-4         503.5 ± 0%

                                 |      main       |
                                 | recv_samples/op |
E2EScrapeAndRemoteWriteNoChurn-4       1.000M ± 0%

                                 |              main               |
                                 | wal_watcher_notifications_total |
E2EScrapeAndRemoteWriteNoChurn-4                       1.012k ± 1%

                                 |          main           |
                                 | wal_watcher_reads_total |
E2EScrapeAndRemoteWriteNoChurn-4               645.0 ± 13%

                                 |     main     |
                                 |     B/op     |
E2EScrapeAndRemoteWriteNoChurn-4   320.3Mi ± 0%

                                 |    main     |
                                 |  allocs/op  |
E2EScrapeAndRemoteWriteNoChurn-4   3.028M ± 1%
```

Profiles

- (BenchmarkE2EScrapeAndRemoteWriteNoChurn) watcher 16% of allocs, 1% of object allocs

github.com/prometheus/prometheus/tsdb/wlog.(*Watcher).readSegment
/Users/bwplotka/Repos/prometheus/tsdb/wlog/watcher.go

```
Total:      1.20GB     1.98GB (flat, cum) 16.36%
   528            .          .                           rec := r.Record()
   529            .          .                           w.recordsRead.WithLabelValues(dec.Type(rec).String()).Inc()
   530            .          .
   531            .          .                           switch dec.Type(rec) {
   532            .          .                           case record.Series:
   533            .    93.19MB                                   series, err = dec.Series(rec, series[:0])
   534            .          .                                   if err != nil {
   535            .          .                                           return err
   536            .          .                                   }
   537            .   174.90MB                                   w.writer.StoreSeries(series, segmentNum)
   538            .          .
   539            .          .                           case record.Samples:
   540            .          .                                   // If we're not tailing a segment we can ignore any samples records we see.
   541            .          .                                   // This speeds up replay of the WAL by > 10x.
   542            .          .                                   if !tail {
   543            .          .                                           break
   544            .          .                                   }
   545            .   537.76MB                                   samples, err = dec.Samples(rec, samples[:0])
   546            .          .                                   if err != nil {
   547            .          .                                           return err
   548            .          .                                   }
   549            .          .                                   for _, s := range samples {
   550            .          .                                           if s.T > w.startTimestamp {
   551            .          .                                                   if !w.sendSamples {
   552            .          .                                                           w.sendSamples = true
   553            .          .                                                           duration := time.Since(w.startTime)
   554            .          .                                                           w.logger.Info("Done replaying WAL", "duration", duration)
   555            .          .                                                   }
   556      1.20GB     1.20GB                                                   samplesToSend = append(samplesToSend, s)
   557            .          .                                           }
   558            .          .                                   }
   559            .          .                                   if len(samplesToSend) > 0 {
   560            .          .                                           w.writer.Append(samplesToSend)
   561            .          .                                           samplesToSend = samplesToSend[:0]
```

github.com/prometheus/prometheus/tsdb/wlog.(*Watcher).readSegment
/Users/bwplotka/Repos/prometheus/tsdb/wlog/watcher.go

```
Total:           0        1MB (flat, cum) 0.0081%
   648            .          .                           default:
   649            .          .                                   // We're not interested in other types of records.
   650            .          .                           }
   651            .          .                   }
   652            .          .                   if err := r.Err(); err != nil {
   653            .        1MB                           return fmt.Errorf("segment %d: %w", segmentNum, err)
   654            .          .                   }
   655            .          .                   return nil
   656            .          .           }
   657            .          .
   658            .          .           // Go through all series in a segment updating the segmentNum, so we can delete older series.
```

- ([BenchmarkWatcher_ReadSegment](#)) PROD DATA
  - 699.95MB 699.95MB samples = make([]RefSample, 0, minSize)
  - 772.15MB 772.15MB samplesToSend = append(samplesToSend, s)
  - After fix there is still something to do in snappy decode

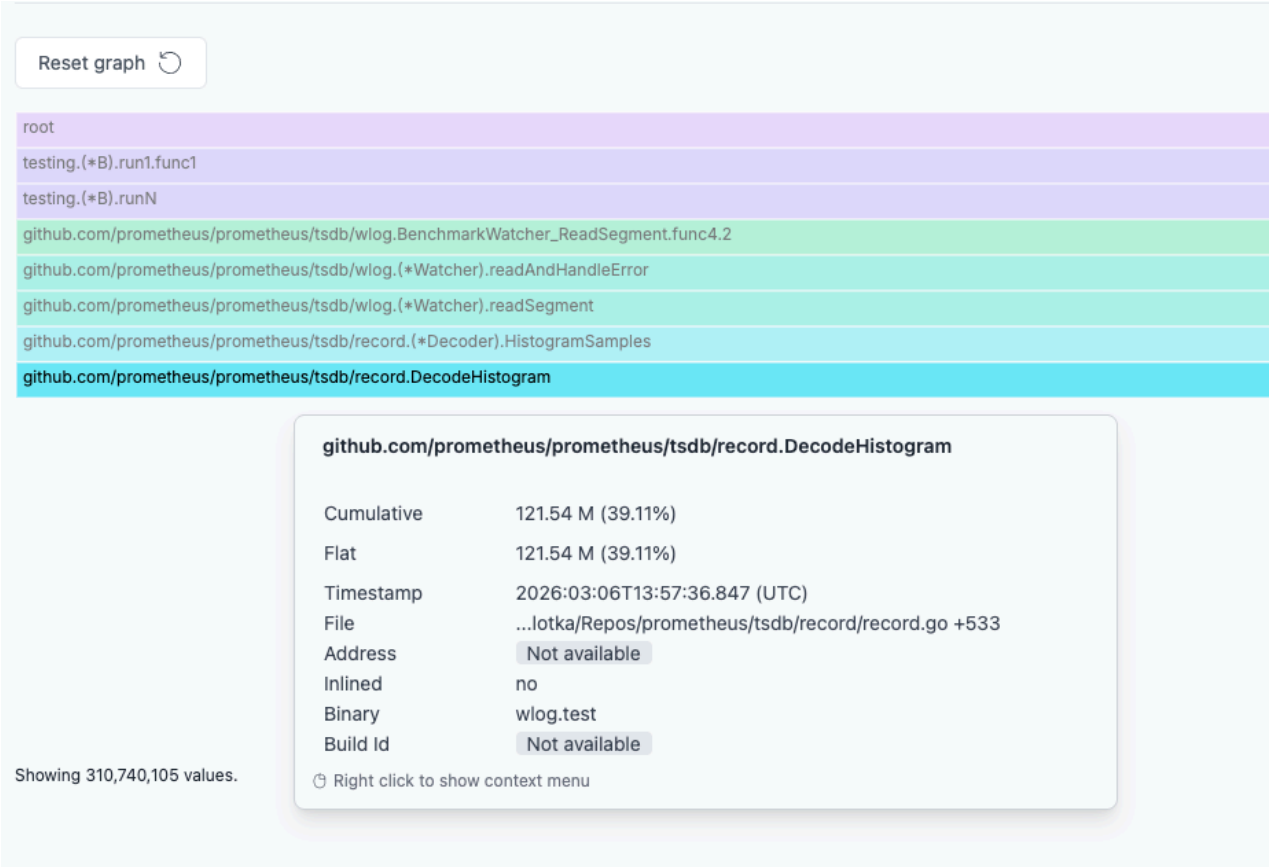
pprof VIEW SAMPLE REFINE CONFIG DOWNLOAD Search regexp

github.com/golang/snappy.Decode

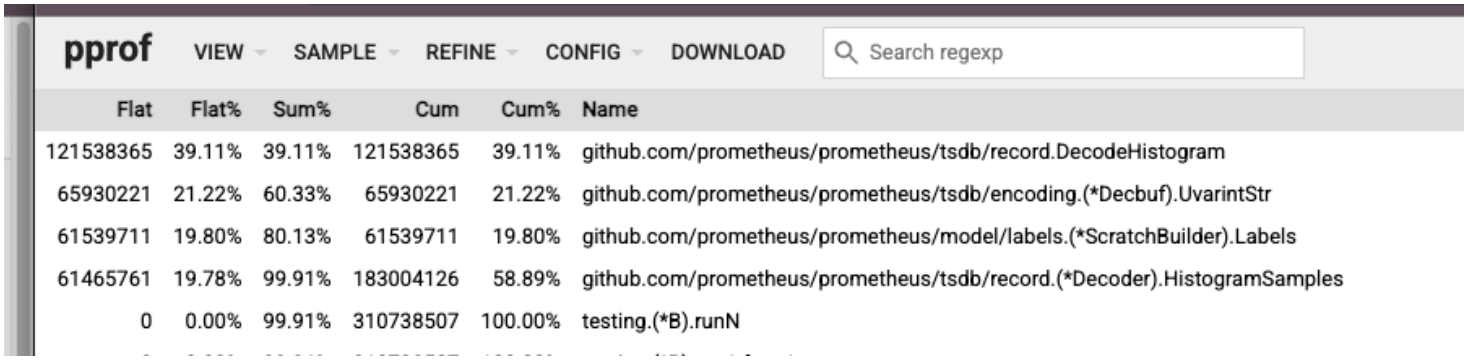
/Users/bwplotka/.gvm/pkgsets/go1.25.5/global/pkg/mod/github.com/golang/snappy@v1.0.0/decode.go

```
Total:         289        289 (flat, cum)  8.00%
   52           .          .           // Otherwise, a newly allocated slice will be returned.
   53           .          .           //
   54           .          .           // The dst and src must not overlap. It is valid to pass a nil dst.
   55           .          .           //
   56           .          .           // Decode handles the Snappy block format, not the Snappy stream format.
   57           .          .           func Decode(dst, src []byte) ([]byte, error) {
   58           .          .                 dLen, s, err := decodedLen(src)
   59           .          .                 if err != nil {
   60           .          .                         return nil, err
   61           .          .                 }
   62           .          .                 if dLen <= len(dst) {
   63           .          .                         dst = dst[:dLen]
   64           .          .                 } else {
   65         289        289                         dst = make([]byte, dLen)
   66           .          .                 }
   67           .          .                 switch decode(dst, src[s:]) {
   68           .          .                 case 0:
   69           .          .                         return dst, nil
   70           .          .                 case decodeErrCodeUnsupportedLiteralLength:
   71           .          .                         return nil, errUnsupportedLiteralLength
   72           .          .                 }
   73           .          .                 return nil, ErrCorrupt
   74           .          .           }
   75           .          .
   76           .          .           // NewReader returns a new Reader that decompresses from r, using the framing
   77           .          .           // format described at
   78           .          .           // https://github.com/google/snappy/blob/master/framing_format.txt
```

-
- ([BenchmarkWatcher_ReadSegment](#)) Synthetic (
  - Decoding histograms is expensive on both size but especially on objects with trillions spans. Optimize later
  - https://pprof.me/fb97345ffc70fc494e2e9ed1aaed2d7c/?profileType=profile%3Aalloc_objects%3Acount%3Aspace%3Abytes



  -



Fix:

- https://github.com/prometheus/prometheus/pull/18250

Next Steps

Ne

NOTE

```
Shell
```