

Arik Alon



Arik Alon

CTO, Robusta Dev

Maintainer, HolmesGPT CNCF

[LinkedIn](#)

Robusta Dev

HolmesGPT

AI SRE — CNCF Sandbox

HolmesGPT — AI SRE for cloud-native

HolmesGPT is an open source AI agent that investigates incidents, performs root cause analysis, and remediates them — automatically. Connects to your existing tools. No new infrastructure.

Investigates

Queries all your tools

Root cause

Prometheus, logs, events

Remediates

PRs, kubectl, actions

Open source

CNCF · MIT license

A big PR lands in production

The bidder team ships a major feature — geo targeting, bulk bids, budget pacing. The PR looks good. Reviews pass. It merges into main.

merged · #42 · arikalon1

Pull requests 3 Agents Actions Projects Security Insights Settings

Add bidder service v2.4.1 with geo targeting, bulk bids, and budget pacing #42

 Merged arikalon1 merged 2 commits into main from claude/create-bidder-demo-BDi1T  1 hour ago

Conversation 0 Commits 2 Checks 0 Files changed 18 +1,127 

 arikalon1 commented 1 hour ago • edited  Member 

This PR introduces a complete bidder service implementation (v2.4.1) with support for real-time bid computation, caching, metrics collection, and budget pacing.

Key Changes

- **Bid Handler** (`api/bid_handler.go`): Implements single and bulk bid request processing with request validation, cache integration, and metrics recording
- **Bidding Engine** (`bidding/engine.go`): Computes optimal bids based on user segments, geographic location, device type, and campaign priority
- **Cache Handler** (`cache/handler.go`): In-memory cache with TTL-based expiration, LRU eviction, and background cleanup. Tracks hit/miss/eviction metrics
- Configurable TTL (default 5 minutes) with environment-specific overrides for staging/load-test
- **Router** (`api/router.go`): Sets up HTTP routes with middleware chain (recovery, request logging, rate limiting)
- **Campaign Models** (`model/campaign.go`): Campaign configuration and statistics tracking
- **Geo Models** (`model/geo.go`): Geographic targeting and bid modifier structures



 claude added 2 commits 1 hour ago

  Add bidder service base implementation (v2.4.0)  Verified 6e83ad8

  Release v2.4.1: geo targeting, bulk bids, pacing, and cache improvements  Verified ff80693

Reviewers  No reviews

Assignees  No one—[assign yourself](#)

Labels  None yet

Projects  None yet

Milestone  No milestone

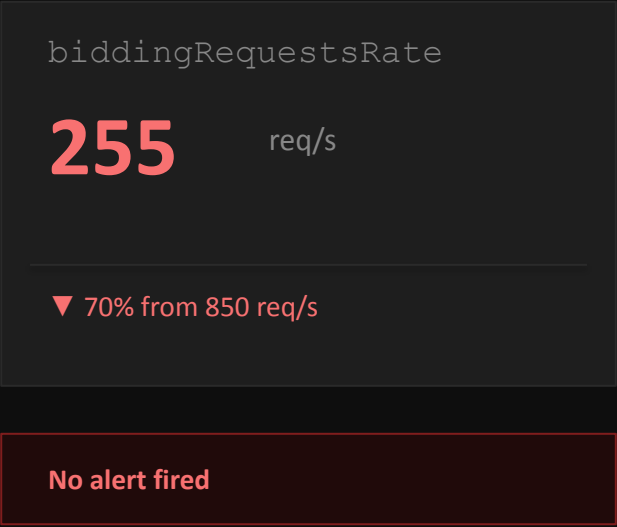
Development  Successfully merging this pull request may close these issues.

None yet

Notifications  Unsubscribe Customize

You're receiving notifications because you're watching this repository.

Something is wrong — but nobody knows yet



Without Holmes, this goes unnoticed for hours.

Holmes fires — immediately

4 / 8

- ✓ Metric drop detected
- ✓ Bad commit identified
- ✓ Fix PR opened



holmes APP 1 minute ago



Investigation complete

Here's the concise version:



Incident Detected — `bidding-service`

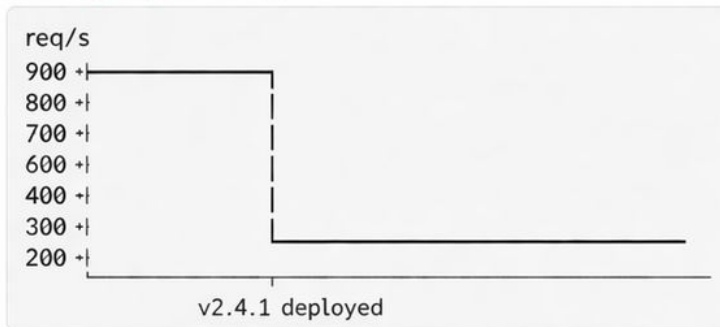
Triggered by: deployment `v2.4.1` • Severity: Critical



Impact

Metric	Before	After	Change
biddingRequestsRate	850 req/s	255 req/s	▼ 70%
Cache hit ratio	92%	0%	▼ 100%
p99 latency	45ms	380ms	▲ 8.4×

`biddingRequestsRate` — Last 1h



[See less](#)



Root Cause

`cache/handler.go:44` — TTL set to `0`, cache effectively disabled. Every bid lookup bypasses Redis and hits the backend directly, causing 70% throughput drop.



Fix PR #43 — Restore cache TTL to 300s

Holmes agent: gke-prod



Related Pull Request #43

fix(bidding): restore cache TTL — v2.4.1 set TTL=0 disabling cache



[arikalon1](#) wants to merge 1 commit into `main`

[fix/bidding-service-cache-ttl](#)

`bidding-service/cache/handler.go`

```
@@ -54,7 +60,7 @@ func (h *BidCacheHandler) GetCachedBid(  
54 // SetCachedBid stores a bid response in cache  
55 // with the configured TTL.  
56 func (h *BidCacheHandler) SetCachedBid(ctx context.Context,  
57     key := fmt.Sprintf("bid:auction:%s", auctionID)  
57 - ttl := time.Duration(0) * time.Second // TTL for bid cache er  
44 + ttl := DefaultBidCacheTTL // 300 * time.Second  
63  
59 if err := h.client.Set(ctx, key, data, ttl).Err(); err != nil {  
61     h.logger.Error("cache write failed",
```



Fix: Restore cache TTL to 300s

Adds `DefaultBidCacheTTL` (300s) and uses it for cached bid entries.

We defined a healthcheck for the bidder

A simple YAML resource tells Holmes what to watch — and what to do when something breaks.

`trigger` on every new deployment

`checks` 3 metrics, 5-minute window

`onFailure` find commit → open PR

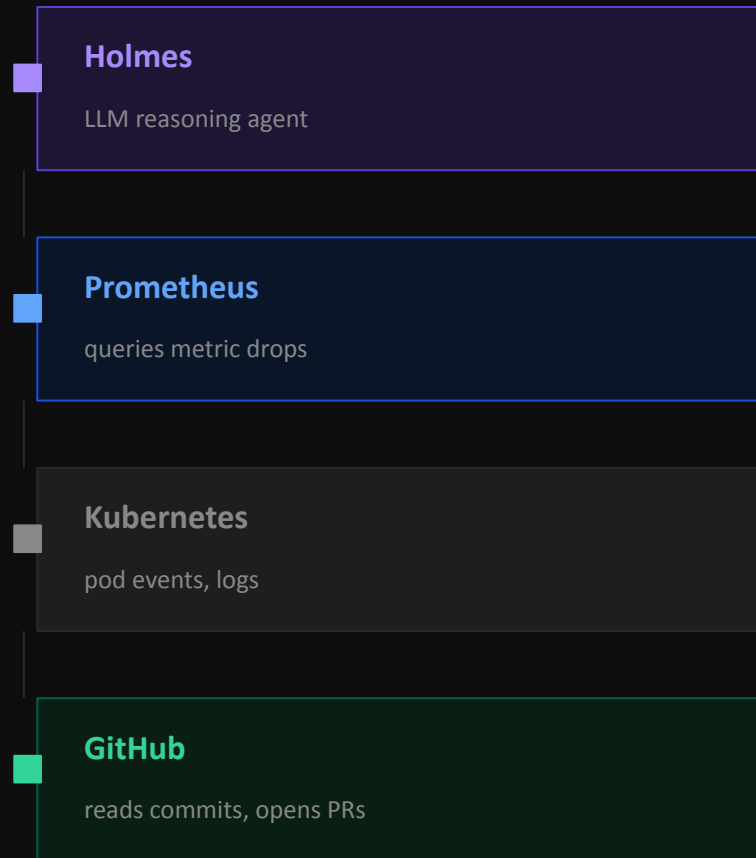
```
apiVersion: holmesgpt.dev/v1alpha1
kind: HealthCheck
metadata:
  name: bidder-deployment-check
  namespace: production
spec:
  query: |
    1. Check the bidder deployment health. Check the logs for errors, events, pod status
    2. Check for any bidding and latency related metrics. Check if there's more than 10% degradation in any of it.
    3. In case you detect an issue, check if it's related to a recent change. Check github, and create a PR with a fix
    4. If there's no option to fix, create a PR to rollback to the previous version
  mode: alert
  destinations:
    - type: slack
      config:
        channel: "#incidents"
```

Holmes queries your tools — then acts

Holmes connects to your existing stack via MCP servers, HTTP Apis, Database connectors or custom integrations.

It queries, correlates, reasons — then executes the remediation.

Any tool with an MCP server can be plugged in — Datadog, PagerDuty, Jira, custom scripts.



Any remediation handler can be attached

The most common remediation keeps humans in the loop. But Holmes can go further — as far as you're comfortable. You define the boundary. Holmes works within it.

Open a pull request

Human reviews and merges — full control

human in the loop

Trigger GitHub Action

Automated rollback pipeline, no human needed

semi-automated

Run kubectl commands

Restart pods, scale deployments, patch configs

fully autonomous

Self-healing infrastructure, built in the open

Open source · MIT · CNCF Sandbox

github

robusta-dev/holmesgpt

01

Deploy HolmesGPT

helm install or kubectl apply

02

Connect your tools

Prometheus, GitHub, k8s via MCP

03

Define a healthcheck

one YAML file per workload

04

Watch it work

extend trust as confidence grows