# HyperSDK

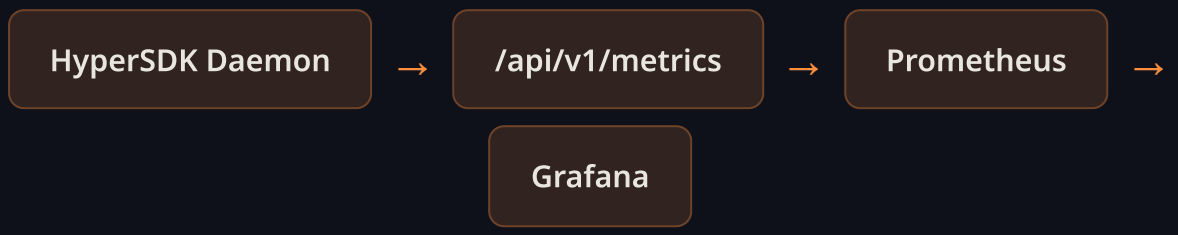
## Monitoring & Observability

Full-stack observability with Prometheus metrics, WebSocket real-time updates, webhook alerting, and Grafana-ready dashboards for complete operational visibility.

Prometheus Native -- Real-Time WebSocket -- Grafana Dashboards

# Metrics Architecture

Prometheus-native metrics exposed via a dedicated API endpoint for seamless integration.




### Prometheus Endpoint

Metrics exposed at `/api/v1/metrics` in standard Prometheus exposition format. Scrape interval configurable; recommended 15s for production.

### System Metrics

Built-in system metrics via the `SystemMetrics` port interface: CPU, memory, disk I/O, network throughput, and goroutine counts collected automatically.

### Port-Based Design

Metrics collection follows hexagonal architecture. The `ports.SystemMetrics` interface decouples collection from export, enabling custom backends.

### Custom Labels

All metrics include provider, region, and job-type labels for multi-dimensional filtering and aggregation in your observability stack.

## Prometheus Scrape Configuration

```
scrape_configs:
  - job_name: 'hypersdk'
    scrape_interval: 15s
```

```
static_configs:
  - targets: ['localhost:8080']
metrics_path: '/api/v1/metrics'
```

# Key Metrics

Comprehensive metrics covering HTTP layer, jobs, WebSocket connections, and runtime health.

| Metric | Type | Description |
|---|---|---|
| `hypersdk_http_requests_total` | Counter | Total HTTP requests by method, path, and status code |
| `hypersdk_http_response_time_seconds` | Histogram | Response latency distribution (p50, p95, p99) |
| `hypersdk_websocket_connections` | Gauge | Current active WebSocket connections |
| `hypersdk_jobs_total` | Counter | Jobs submitted by provider and type |
| `hypersdk_jobs_active` | Gauge | Currently running jobs |
| `hypersdk_jobs_duration_seconds` | Histogram | Job execution duration by provider |
| `hypersdk_memory_bytes` | Gauge | Current memory usage (heap, stack, sys) |
| `hypersdk_goroutines` | Gauge | Active goroutine count |

**8+**
Core Metric Families

**15s**
Recommended Scrape Interval

**p99**
Latency Percentiles

**0%**
Overhead on Hot Path

# WebSocket Real-Time Updates

Live streaming of job progress and system events to dashboards and integrations.

### Live Job Progress

WebSocket streams deliver real-time progress updates for active export and migration jobs. Clients receive percentage completion, bytes transferred, and estimated time remaining.

### Dashboard Auto-Refresh

The React dashboard connects via WebSocket for instant UI updates. No polling required -- state changes propagate in under 100ms to all connected clients.

### Connection Management

Automatic reconnection with exponential backoff. Connection health monitored via ping/pong frames. Graceful degradation to HTTP polling if WebSocket unavailable.

### Authenticated Streams

WebSocket connections require the same authentication as REST API. Token-based auth validated on upgrade. Unauthorized connections are rejected immediately.

## WebSocket Event Types

| Event | Payload | Trigger |
| --- | --- | --- |
| `job.progress` | Job ID, percentage, bytes, ETA | Every 1s during active transfer |

| | | |
|---|---|---|
| `job.status` | Job ID, new status, timestamp | On state transition |
| `system.metrics` | CPU, memory, goroutines | Every 5s to dashboard |
| `alert.fired` | Alert name, severity, details | On threshold breach |

# Alerting & Notifications

Webhook-based alerting for job lifecycle events with support for Slack, Discord, and custom endpoints.

### Slack Integration

Send job notifications directly to Slack channels via incoming webhooks. Rich message formatting with job details, provider info, and action links.

### Discord Webhooks

Native Discord webhook support with embedded messages. Color-coded by event type: green for success, red for failure, orange for warnings.

### Generic Webhooks

Send JSON payloads to any HTTP endpoint. Configurable headers, retry logic with exponential backoff, and HMAC signature verification.

## Webhook Event Types

job.started → job.progress → job.completed

job.failed → job.retry → system.alert

## Configuration Example

```
webhooks:
  - name: "slack-ops"
    url: "https://hooks.slack.com/services/T.../B.../xxx"
    events: ["job.completed", "job.failed"]
```

```
    retry_count: 3
    retry_delay: "5s"
  - name: "pagerduty"
    url: "https://events.pagerduty.com/v2/enqueue"
    events: ["job.failed"]
    headers:
      Authorization: "Token token=YOUR_KEY"
```

# Grafana Dashboards

Pre-built Grafana dashboard panels for API latency, job throughput, and resource usage monitoring.

### API Latency Panel

Histogram panel showing p50, p95, and p99 response times across all API endpoints. Grouped by route for quick identification of slow paths.

- Source: `hypersdk_http_response_time_seconds`
- Visualization: Heatmap + line graph
- Alert threshold: p99 > 500ms

### Job Throughput Panel

Rate of jobs submitted, completed, and failed over time. Stacked area chart by provider for capacity planning and trend analysis.

- Source: `hypersdk_jobs_total`
- Visualization: Stacked area chart
- Alert threshold: failure rate > 5%

### Resource Usage Panel

System resource utilization including memory heap, goroutine count, and GC pause times. Essential for capacity planning and leak detection.

- Source: `hypersdk_memory_bytes`, `hypersdk_goroutines`
- Visualization: Time series + gauge
- Alert threshold: memory > 80% limit

### Active Connections Panel

WebSocket connections, HTTP connection pool usage, and provider API connection counts. Track connection leaks and saturation.

- Source: `hypersdk_websocket_connections`
- Visualization: Stat + time series
- Alert threshold: connections > pool max

## Sample Dashboard JSON (Import to Grafana)

```
{
  "dashboard": {
    "title": "HyperSDK Overview",
    "panels": [
      { "title": "API Latency (p99)", "type": "timeseries",
        "targets": [{"expr": "histogram_quantile(0.99, hypersdk_http_response_time_seconds_
      { "title": "Job Throughput", "type": "timeseries",
        "targets": [{"expr": "rate(hypersdk_jobs_total[5m])"}] },
      { "title": "Memory Usage", "type": "gauge",
        "targets": [{"expr": "hypersdk_memory_bytes{type=\"heap\"}"}] },
      { "title": "Active Goroutines", "type": "stat",
        "targets": [{"expr": "hypersdk_goroutines"}] }
    ]
  }
}
```

## Complete Observability Stack

HyperSDK provides everything you need: metrics export, real-time streaming, alerting, and dashboard templates. Deploy a production-grade monitoring stack in minutes.