OPHION - Manual Técnico

Plataforma de Observabilidade com IA

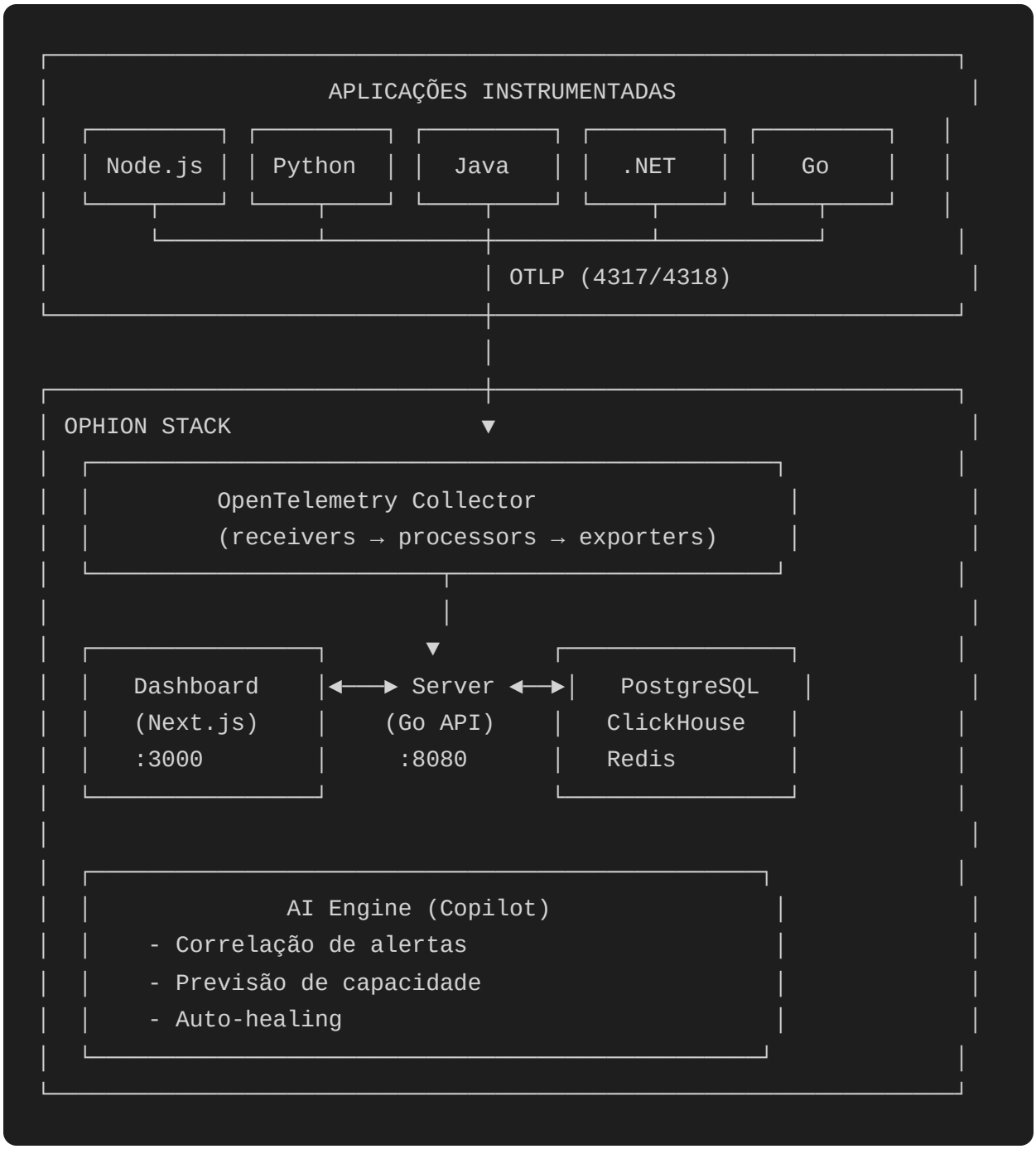
1. Visão Geral

OPHION é uma plataforma de observabilidade open source que combina **métricas, logs e traces** em uma única solução, potencializada por **inteligência artificial** para monitoramento proativo e auto-healing.

2. Stack Tecnológico

Camada	Tecnologia	Função
API Server	Go 1.22	Backend principal
Dashboard	Next.js + TypeScript	Interface web
Collector	OpenTelemetry Collector	Ingestão de telemetria
Database	PostgreSQL	Dados estruturados
Time Series	ClickHouse	Métricas de alta performance
Cache	Redis	Cache e filas
Container	Docker Compose	Orquestração

3. Arquitetura do Sistema



---

4. Estrutura de Diretórios

```
ophion/
├── cmd/
│   ├── server/          # API Server (Go)
│   └── agent/           # Agent de coleta
├── internal/            # Código interno Go
│   ├── api/             # Handlers HTTP
```

```
│   ├── db/               # Repositórios
│   ├── ai/               # Engine de IA
│   └── telemetry/        # Processamento OTLP
├── dashboard/            # Frontend Next.js
│   ├── src/
│   │   ├── app/          # App Router
│   │   ├── components/   # UI Components
│   │   └── lib/          # Utilitários
│   └── package.json
├── deploy/
│   ├── docker/           # Docker configs
│   │   └── otel-collector-config.yaml
│   ├── remote-agent/     # Agent remoto
│   └── instrumentation/  # Scripts de instrumentação
├── examples/             # Exemplos por linguagem
│   ├── otel-nodejs/
│   ├── otel-python/
│   └── docker/
├── configs/              # Configurações
├── docs/                 # Documentação
├── instrument.sh         # Script de auto-instrumentação
├── install.sh            # Instalador
├── docker-compose.yml
├── go.mod / go.sum
└── server                # Binário compilado
```

---

5. Componentes Principais

5.1 OpenTelemetry Collector

Portas: - `4317` - OTLP gRPC - `4318` - OTLP HTTP

Pipeline:

```
receivers:
  otlp:
    protocols:
      grpc:
        endpoint: 0.0.0.0:4317
      http:
```

```
        endpoint: 0.0.0.0:4318

processors:
  batch:
    timeout: 1s
  memory_limiter:
    limit_mib: 512

exporters:
  ophion:
    endpoint: http://server:8080
```

5.2 Server (Go API)

Endpoints principais:

Endpoint	Método	Descrição
`/api/v1/traces`	POST	Ingestão de traces
`/api/v1/metrics`	POST	Ingestão de métricas
`/api/v1/logs`	POST	Ingestão de logs
`/api/v1/services`	GET	Lista serviços
`/api/v1/alerts`	GET/POST	Gerenciamento de alertas
`/api/v1/ai/analyze`	POST	Análise por IA

5.3 Dashboard (Next.js)

Recursos: - Service Map visual - Trace waterfall - Métricas em tempo real - Logs agregados - Alertas e notificações - AI Copilot chat

---

6. Auto-Instrumentação

6.1 Script Universal

```
# Auto-detecta linguagem
./instrument.sh <container-name>

# Especifica linguagem
./instrument.sh my-app nodejs
./instrument.sh my-app python
./instrument.sh my-app java
./instrument.sh my-app dotnet
```

6.2 Suporte por Linguagem

Linguagem	Método	Complexidade
.NET	Auto-instrumentation	Zero code
Node.js	Auto-instrumentation	Zero code
Python	Auto-instrumentation	Zero code
Java	Java Agent	Zero code
Go	SDK (compile-time)	Pequenas mudanças
PHP	SDK	Pequenas mudanças

6.3 Variáveis de Ambiente

```
OTEL_EXPORTER_OTLP_ENDPOINT=http://ophion:4318
OTEL_SERVICE_NAME=my-service
OTEL_RESOURCE_ATTRIBUTES=deployment.environment=production
```

---

7. AI Engine

7.1 Funcionalidades

- **Correlação de Alertas**: Agrupa alertas relacionados
- **Root Cause Analysis**: Identifica causa raiz
- **Previsão de Capacidade**: Prevê saturação de recursos
- **Auto-Healing**: Executa ações corretivas automáticas
- **Copilot**: Chat para consultas em linguagem natural

7.2 Integração OpenAI

```
OPENAI_API_KEY=sk-...
AI_MODEL=gpt-4
```

---

8. Deploy

8.1 Quick Start (Docker)

```
git clone https://github.com/bigtux/ophion.git
cd ophion
docker compose up -d
```

8.2 Acessos

Serviço	URL	Porta
Dashboard	http://localhost:3000	3000
API	http://localhost:8080	8080
OTLP gRPC	localhost:4317	4317
OTLP HTTP	localhost:4318	4318

8.3 Produção

```
# Com install.sh
./install.sh --production

# Ou manualmente
docker compose -f docker-compose.prod.yml up -d
```

---

9. Requisitos do Sistema

Recurso	Mínimo	Recomendado
CPU	2 cores	4+ cores
RAM	4 GB	8+ GB
Disco	20 GB SSD	100+ GB SSD
Docker	20.10+	Latest
Docker Compose	v2+	Latest

---

10. Configuração

10.1 Variáveis de Ambiente

```
# Server
PORT=8080
DATABASE_URL=postgres://user:pass@localhost:5432/ophion
REDIS_URL=redis://localhost:6379
CLICKHOUSE_URL=clickhouse://localhost:9000

# Auth
JWT_SECRET=your-secret
AGENT_KEY=agent-secret-key
```

```
# AI
OPENAI_API_KEY=sk-...
AI_ENABLED=true

# Notifications
TELEGRAM_BOT_TOKEN=...
TELEGRAM_CHAT_ID=...
```

10.2 Configuração do Collector

```
# otel-collector-config.yaml
receivers:
  otlp:
    protocols:
      grpc:
        endpoint: 0.0.0.0:4317
      http:
        endpoint: 0.0.0.0:4318

processors:
  batch:
    timeout: 1s
    send_batch_size: 1024

exporters:
  otlphttp:
    endpoint: http://server:8080/api/v1

service:
  pipelines:
    traces:
      receivers: [otlp]
      processors: [batch]
      exporters: [otlphttp]
```

---

11. Segurança

- Autenticação JWT para dashboard
- API Key para agents (`AGENT_KEY`)
- TLS opcional para OTLP
- Isolamento de rede via Docker
- Logs de auditoria

---

12. Desenvolvimento

```
# Backend (Go)
go run ./cmd/server

# Frontend (Next.js)
cd dashboard
npm install
npm run dev

# Testes
go test ./...
```

---

13. Troubleshooting

Problema	Solução
Traces não aparecem	Verificar OTEL_EXPORTER_OTLP_ENDPOINT
Dashboard lento	Aumentar RAM do ClickHouse
Collector crash	Verificar memory_limiter no config
Auth falhando	Verificar JWT_SECRET

---

14. Licença

AGPL-3.0 (Community Edition)

Documento gerado automaticamente - OPHION