

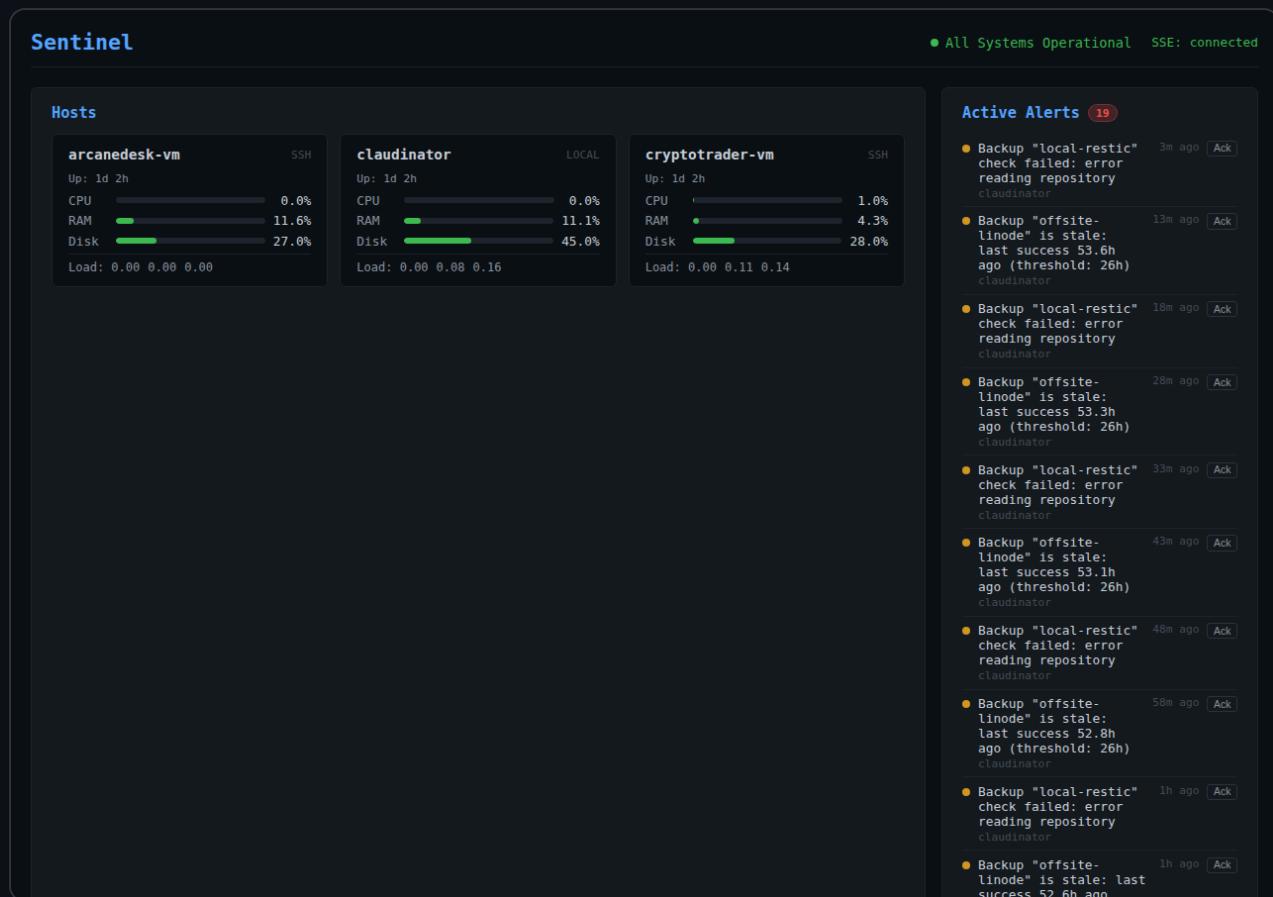
# Sentinel

## Table of Contents

- Sentinel

- What is this? - Screenshots - Quick Start - How to Use - Viewing Your Infrastructure - Monitoring Active Work - Tracking Projects - Reviewing History - Managing Alerts - Restarting Containers - Features - Installation - Prerequisites - Detailed Setup - Running in Production - Configuration - Host Configuration - Environment Variables - Alert Thresholds - Tech Stack - License

Real-time infrastructure monitoring for your entire server fleet — see everything that's happening across all your VMs, Docker containers, and background tasks in one place.



## What is this?

Sentinel is a monitoring dashboard that watches over your infrastructure 24/7. Instead of SSHing into each server to check if things are running, you get a single web interface showing the health of all your hosts, containers, services, and automated tasks. It's

designed for small teams running multiple VMs who need instant visibility without the complexity of enterprise monitoring tools.

## Screenshots

The screenshot displays the main dashboard interface. At the top right, there is a system alert for a backup check failure: "Backup 'local-restic' 2h ago [Ack] check failed: error reading repository claudinator". Below this, the "Docker Containers" section shows five containers: "arcanedesk-postgres" (running 9 hours, 0.0% CPU, 175M Mem), "arcanedesk-valkey" (running 9 hours, 0.2% CPU, 3M Mem), "postgres" (running 26 hours, 0.0% CPU, 52M Mem), "questdb" (running 26 hours, 7.2% CPU, 712M Mem), and "valkey" (running 9 hours, 0.1% CPU, 3M Mem). The "ForgeTeam Activity" section includes a task summary (0 IN PROGRESS, 18 TODO, 1 BLOCKED, 219 DONE) and a list of recent activity items from various users. At the bottom left, there is a "Backups" button.

*The main dashboard gives you an at-a-glance view of all your infrastructure — host status, CPU/memory usage, and running containers*

This screenshot shows a similar view to the first one. The top right features a system alert about a backup failure. The "Docker Containers" section lists the same five containers. The "ForgeTeam Activity" section shows a more detailed list of tasks, including their status, assignees, and priority levels. For example, task #8 is blocked and assigned to "Marketeer" with a low priority. Other tasks listed include resolving open questions, setting up DNS, and various phases of development.

*See what's currently running across your infrastructure — deployments, backups, and automated maintenance tasks*

**Docker Containers**

- arcanedesk-postgres**: arcanedesk-vm, • running (9 hours), CPU: 0.0% Mem: 175M
- arcanedesk-valkey**: arcanedesk-vm, • running (9 hours), CPU: 0.2% Mem: 3M
- postgres**: cryptotrader-vm, • running (26 hours), CPU: 0.0% Mem: 52M
- questdb**: cryptotrader-vm, • running (26 hours), CPU: 7.3% Mem: 713M
- valkey**: cryptotrader-vm, • running (9 hours), CPU: 0.2% Mem: 3M

**ForgeTeam Activity**

IN PROGRESS	TODO	BLOCKED	DONE
0	18	1	219

Active Tasks Projects Recent Activity Open Questions

- WipeStation**: WipeStation - planning, • 7 todo
- Crucible**: 3D Printer Management Platform - planning, • 3 todo, • 11 done
- Marketeer**: Social Media Automation - active, • 2 todo, • 1 blocked, • 7 done
- TCGKungfu**: TCGKungfu Kiosk - active, • 2 todo, • 36 done
- ArrMada**: ArrMada - active, • 1 todo, • 10 done
- ClaudeStick**: ClaudeStick - active, • 1 todo, • 11 done
- DOGE-HABEUS**: DOGE-HABEUS Blockchain - active, • 2 todo, • 8 done

**Backups** Check Now

```
local-restic
Save(<lock/e8d4bd965e>) returned error, retrying
/mnt/backup-w/windows/claudinator-
backups/locks/e8d4bd965e3087b2546591c6375de14f
tmp-4106109358: permission denied Save(<lock/e8
retrying after 1.027815031s: open /mnt/backup-w/
backups/locks/e8d4bd965e3087b2546591c6375de14f
tmp-3076658028: permission denied Save(<lock/e8
```

*Track progress across all your projects with completion rates and task counts*

**Docker Containers**

- arcanedesk-postgres**: arcanedesk-vm, • running (9 hours), CPU: 0.0% Mem: 175M
- arcanedesk-valkey**: arcanedesk-vm, • running (9 hours), CPU: 0.2% Mem: 3M
- postgres**: cryptotrader-vm, • running (26 hours), CPU: 0.0% Mem: 52M
- questdb**: cryptotrader-vm, • running (26 hours), CPU: 7.3% Mem: 713M
- valkey**: cryptotrader-vm, • running (9 hours), CPU: 0.2% Mem: 3M

**ForgeTeam Activity**

IN PROGRESS	TODO	BLOCKED	DONE
0	18	1	219

Active Tasks Projects Recent Activity Open Questions

**Recent Completions**

- Crucible #243 OctoPrint printer adapter: REST API integration 19m ago
- Crucible #242 Moonraker printer adapter: REST + WebSocket integration 19m ago
- Crucible #241 Crucible scaffolding: Next.js 14 + tRPC + Prisma + shadcn/ui 29m ago
- Crucible #196 Determine deployment architecture 37m ago
- Crucible #195 Design database schema 38m ago
- Athena #238 Auto-generated Mermaid diagrams 52m ago
- Athena #240 Athena config file and doc freshness tracking 52m ago
- Athena #236 Terminal output capture and screenshot system 54m ago
- Athena #239 Git-based changelog and version history generator 58m ago
- TCGKungfu #231 Events and tournament calendar system 1h ago
- TCGKungfu #230 Email newsletter system for store events and promotions 1h ago

**Recent Decisions**

- Crucible OctoPrint adapter: polling over WebSocket: Use REST API polling at 2-second intervals for status updates instead of OctoPrint WebSocket (SockJS) 26m ago
- Crucible Printer adapter architecture: PrinterAdapter interface with EventEmitter pattern + MoonrakerAdapter as first implementation. REST for commands/queries, WebSocket JSON-RPC for real-time subscriptions. ConnectionManager singleton for multi-printer lifecycle with auto-reconnect and temperature polling. 26m ago
- Crucible tRPC v11 integration with Next.js 14: Use tRPC v11 with fetch adapter via Next.js App Router route handler instead of @trpc/next package 34m ago
- Crucible Deployment architecture: Dedicated Proxmox VM (4 cores, 8GB RAM, 100GB SSD on fast\_pool). Build on Claudinator, rsync to VM. Docker for PostgreSQL 16 + Valkey 8 only; Node.js app runs directly. Static IPs: 192.168.0.69 (LAN vmbro0) + 10.10.10.4 (internal vmbri). SSH alias: crucible. 38m ago
- Crucible Database Schema Design: Prisma schema with 10 models: User, PrinterGroup, Printer, GcodeFile, PrintJob, Filamentspool, Spoolusage, Printhistory, PrinterTemperature, Notification. Uses cuid() IDs, PostgresSQL enums for statuses/adapters/materials\_snake\_case\_DB. 38m ago

*A live feed of everything happening in your infrastructure — deployments, restarts,*

*backups, and alerts*

The screenshot shows the Sentinel dashboard interface. At the top right, there's a notification for a backup check failure: "Backup 'local-restic' 2h ago [Ack] check failed: error reading repository claudinator". Below this, under "Docker Containers", five containers are listed: "arcanedesk-postgres" (running 9 hours), "arcanedesk-valkey" (running 9 hours), "postgres" (running 26 hours), "questdb" (running 26 hours), and "valkey" (running 9 hours). In the "ForgeTeam Activity" section, there are four status indicators: 0 IN PROGRESS, 18 TODO, 1 BLOCKED, and 219 DONE. The "Open Questions" tab is selected, showing a list of items from various users like FellowshipFirst, WipeStation, ArrMada, TCGKungfu, and others, each with a timestamp.

*Track unresolved issues and questions that need attention across your projects*

## Quick Start

### 1. Clone the repository

```
git clone <repository-url>
cd sentinel
```

### 2. Install dependencies

```
npm install
```

### 3. Set up your configuration

```
cp config.example.json config.json
# Edit config.json with your host details
```

### 4. Start the server

```
npm start
```

## 5. Open your browser

```
http://localhost:3000
```

That's it! The dashboard will start collecting metrics from your configured hosts immediately.

# How to Use

---

## Viewing Your Infrastructure

When you open Sentinel, the **main dashboard** shows you everything at once:

- **Host Status** panel lists all your VMs with their uptime and current state
- **System Metrics** displays live CPU, memory, and disk usage graphs
- **Container Overview** shows which Docker containers are running on each host

Click any host to drill down into detailed metrics and container logs.

## Monitoring Active Work

Switch to the **Active Tasks** tab to see what's currently happening:

- Running deployments and their progress
- Scheduled backup jobs
- Automated maintenance tasks
- Container restarts and health checks

Each task shows a real-time status indicator and elapsed time.

## Tracking Projects

The **Projects** tab gives you a birds-eye view of all ongoing work:

- See how many tasks are running vs. completed for each project
- Click a project to filter the activity feed
- Check completion percentages to spot bottlenecks

## Reviewing History

The **Recent Activity** feed is your infrastructure's event log:

- Every deployment, restart, backup, and alert is timestamped

- Filter by host, project, or event type
- Click any event to see full details and logs

## Managing Alerts

Sentinel automatically creates alerts when something needs your attention:

- Container crashes or failed health checks
- Disk space running low
- Services that have been down too long
- Failed backup jobs

Acknowledge alerts from the dashboard to track what you've addressed.

## Restarting Containers

Need to restart a container? Click the container name in the dashboard, then hit the **Restart** button. You can also enable **Auto-Restart** to automatically recover from crashes.

## Features

---

- **Real-time monitoring** — see updates as they happen, no page refresh needed
- **Multi-host management** — track dozens of VMs from a single dashboard
- **Container visibility** — know which Docker containers are running where
- **Task orchestration** — watch automated deployments and maintenance tasks
- **Smart alerting** — get notified when something actually needs your attention
- **Activity timeline** — complete audit log of infrastructure changes
- **SSH-based collection** — no agents to install on your servers
- **Backup tracking** — verify your backup jobs are running on schedule
- **One-click restarts** — quickly recover containers without SSHing

## Installation

---

### Prerequisites

- **Node.js** 18 or higher
- **SSH access** to the hosts you want to monitor
- **SSH keys** configured for passwordless login

### Detailed Setup

#### 1. Clone and install

```
git clone <repository-url>
cd sentinel
npm install
```

## 2. Create your config file

```
cp config.example.json config.json
```

## 3. Add your hosts to config.json

```
{
  "hosts": [
    {
      "id": "web-01",
      "name": "Web Server 01",
      "hostname": "192.168.1.10",
      "port": 22,
      "username": "admin"
    }
  ]
}
```

## 4. Set up SSH keys

Sentinel needs passwordless SSH access to collect metrics:

```
ssh-keygen -t ed25519 -f ~/.ssh/sentinel
ssh-copy-id -i ~/.ssh/sentinel.pub admin@192.168.1.10
```

## 5. Configure SSH key path

If using a non-default key location, set the environment variable:

```
export SSH_KEY_PATH=~/.ssh/sentinel
```

## 6. Start the server

```
npm start
```

For development with auto-reload:

```
npm run dev
```

## 7. Verify it's working

Open <http://localhost:3000/health> — you should see `{"status": "ok"}`

## Running in Production

For production deployments, use a process manager like PM2:

```
npm install -g pm2
pm2 start src/server.js --name sentinel

pm2 save

pm2 startup
```

## Configuration

### Host Configuration

Each host in `config.json` supports these options:

```
{
  "id": "unique-identifier",
  "name": "Human-friendly name",
  "hostname": "IP or domain",
  "port": 22,
  "username": "ssh-user",
  "tags": ["production", "web"],
  "checkInterval": 30
}
```

- `checkInterval` : How often to collect metrics (seconds, default: 30)
- `tags` : For filtering and grouping hosts in the dashboard

### Environment Variables

- `PORT` — Server port (default: 3000)
- `SSH_KEY_PATH` — Path to SSH private key (default: `~/.ssh/id_rsa`)
- `DB_PATH` — SQLite database location (default: `./data/sentinel.db`)
- `LOG_LEVEL` — Logging verbosity: `error`, `warn`, `info`, `debug`

### Alert Thresholds

Edit alert rules in `config.json`:

```
{
  "alerts": {
    "cpu_threshold": 90,
```

```
"memory_threshold": 85,  
  "disk_threshold": 90,  
  "container_down_minutes": 5  
}  
}
```

## Tech Stack

---

- **Backend:** Express.js + Node.js
- **Database:** SQLite (better-sqlite3)
- **SSH:** ssh2 for remote metric collection
- **Scheduling:** node-cron for periodic checks
- **Frontend:** Vanilla JavaScript (no framework dependencies)

## License

---

Copyright © 2026, TheForge, LLC. All rights reserved.

---

Generated by Athena — © 2026 TheForge, LLC