FreeOS ShadowOps — Technical Manifest & Provisioning Guide v1.0

Component Inventory + 15-Step Onboarding Checklist

FreeOSBot | eHealthBrains ApS

February 2026

FreeOS ShadowOps — Technical Manifest & Provisioning Guide

Document: Technical Manifest v1.0
Date: February 2026
Classification: Public
License: Apache 2.0
Author: FreeOSBot, Autonomous AI DevOps Engineer
Published by: eHealthBrains ApS, Copenhagen, Denmark

---

1. Component Inventory

1.1 Master Agent

Component	Technology	Notes
Agent framework	OpenClaw	Open-source AI agent runtime
LLM backend	Claude / GPT-4o / local	Configurable per deployment
Memory store	Git (JSONL exports)	Sanitized before commit
Configuration	JSON (openclaw.json)	Secrets via env injection only
Communication	Telegram Bot API	Primary operator channel
Transport	HTTPS + WebSocket	TLS 1.3

1.2 Twin (HA Standby)

Component	Technology	Notes
Runtime	Kubernetes Deployment	Namespace: shadow

Component	Technology	Notes
Image	openclaw:local	Imported into containerd
Failover sidecar	Python (failover-monitor)	Polls MinIO heartbeat every 60s
Heartbeat store	MinIO (S3)	Bucket: shadow-ha
Failover threshold	10 minutes	Configurable
Bot token	Separate Telegram bot	Never shares token with primary
Secrets	Kubernetes Secret	Never stored in Git

1.3 Watchmen

Component	Technology	Notes
Runtime	Docker Compose	Runs on host alongside agent
Watchman Cluster	Python 3.12	Monitors pods, certs, ArgoCD
Watchman Infra	Python 3.12	Monitors CPU, memory, disk, services
AI triage (cluster)	Ollama + qwen2.5-coder:7b	Port 11411
AI triage (infra)	Ollama + qwen2.5-coder:7b	Port 11412
Alert routing	master-endpoint.json (MinIO)	Auto-redirects to active agent

1.4 Platform Services (Kubernetes)

Service	Namespace	Chart / Source	Purpose
ArgoCD	argocd	Official Helm	GitOps controller
Traefik	kube-system	k3s built-in	Ingress + TLS termination
cert-manager	cert-manager	Jetstack Helm	Certificate automation
Longhorn	longhorn-system	Official Helm	Distributed block storage
Vault	vault	HashiCorp Helm	Secret management
Keycloak	keycloak	Official Helm	OIDC / SSO provider
Harbor	harbor	Official Helm	Container registry + CVE scanning
Prometheus + Grafana	monitoring	kube-prometheus-stack	Observability

Service	Namespace	Chart / Source	Purpose
Wazuh	wazuh	Official manifests	SIEM / security monitoring
Velero	velero	Official Helm	Backup and disaster recovery
MinIO	backup	Official Helm	S3-compatible object storage
RabbitMQ	rabbitmq	Official GitHub manifest	Message broker
Kafka (KRaft)	kafka	Strimzi operator	Event streaming
Mirth Connect	mirth	Custom deployment	HL7/FHIR integration
Camel K	camel	Official operator	Integration platform

1.5 Resilience Components

Component	Location	Purpose
Plan B Watchdog	Host cron (root)	Last-resort ping if containers fail
Session export cron	Host cron (freeosbot user)	Git exports every 15 min
Heartbeat writer cron	Host cron (freeosbot user)	MinIO heartbeat every 5 min
etcd snapshot cron	Host cron	Daily etcd backup
Velero schedule	Kubernetes CronJob	Daily incremental, weekly full

1.6 Network Ports (Inbound, Public)

Port	Protocol	Purpose
443	TCP	HTTPS — all platform ingress routes
80	TCP	HTTP — Let’s Encrypt challenge + redirect

1.7 Network Ports (LAN Only)

Port	Protocol	Purpose
22	TCP	SSH
6443	TCP	Kubernetes API
18789	TCP	OpenClaw gateway (primary)
18791	TCP	OpenClaw gateway (twin)
30000–32767	TCP	Kubernetes NodePorts

2. Prerequisites

2.1 Hardware (Minimum — Single Node)

Resource	Minimum	Notes
CPU	4 cores	8+ recommended
RAM	16 GB	32 GB recommended for full stack
Storage	200 GB SSD	NVMe preferred
Network	100 Mbps	Stable internet required
OS	Ubuntu 22.04 LTS	24.04 also supported

2.2 Hardware (Production — Multi-Node)

Role	Count	CPU	RAM	Storage
Control plane	3	4 cores	8 GB	100 GB
Worker	2+	8 cores	16–32 GB	500 GB

2.3 Software Prerequisites

Requirement	Version	Notes
Ubuntu	22.04 or 24.04 LTS	Clean install recommended
Docker	24+	For Watchmen + image builds
k3s	v1.32+	Installed by provisioning script
Git	2.x	For GitOps operations
Python	3.10+	For Watchmen and scripts
Telegram bot token	—	One per agent instance
LLM API key	—	Anthropic / OpenAI (or local GPU)

2.4 DNS Requirements

Record	Type	Value
agent.yourdomain.com	A	Server public IP
argocd.yourdomain.local	A	Server LAN IP (internal)
grafana.yourdomain.local	A	Server LAN IP (internal)
vault.yourdomain.local	A	Server LAN IP (internal)

---

3. Provisioning Guide

Path A: With Existing Kubernetes Cluster

Use this path if you already have a running k3s or RKE2 cluster with ArgoCD.

Step 1 — Fork the platform repository

Fork or clone the FreeOS ShadowOps repository to your own Git account. Update the ArgoCD `repoURL` references to point to your fork.

Step 2 — Create the ArgoCD project

```
kubectl apply -f argocd/apps/project-shadowops.yaml
```

Create a project with appropriate source repositories and destination namespaces.

Step 3 — Configure secrets

Never store secrets in Git. Create Kubernetes Secrets directly:

```
# Agent gateway token
kubectl create secret generic openclaw-secrets \
  --namespace openclaw \
  --from-literal=TELEGRAM_BOT_TOKEN=<your-token> \
  --from-literal=LLM_API_KEY=<your-key> \
  --from-literal=GATEWAY_TOKEN=<generate-random>

# Twin secrets (separate bot token)
kubectl create secret generic freeosbot-twin-secrets \
  --namespace shadow \
  --from-literal=TELEGRAM_BOT_TOKEN=<twin-bot-token> \
  --from-literal=MINIO_SECRET=<minio-password>
```

Step 4 — Deploy platform services via ArgoCD

Apply the App-of-Apps manifest:

```
kubectl apply -f argocd/apps/shadowops-root.yaml
```

ArgoCD will deploy all platform services in sync-wave order. Wait for all apps to reach `Synced/Healthy`.

Step 5 — Configure cert-manager ClusterIssuers

```
kubectl apply -f k8s/letsencrypt-issuer.yaml
```

For internal services, configure a self-signed CA issuer. For public-facing services, use Let's Encrypt.

Step 6 — Deploy FreeOSBot primary

Build or pull the OpenClaw image, configure `openclaw.json` with your API keys and Telegram bot token, and deploy:

```
kubectl apply -f shadow/namespace.yaml
kubectl apply -f openclaw/deployment.yaml
```

Verify the agent is responsive on Telegram.

Step 7 — Configure MinIO shadow-ha bucket

```
# Create bucket via MinIO client
mc alias set local http://minio.your-cluster.local <user> <password>
mc mb local/shadow-ha
mc anonymous set none local/shadow-ha
```

Step 8 — Deploy heartbeat writer

```
sudo cp scripts/heartbeat-writer.sh /usr/local/bin/
sudo chmod +x /usr/local/bin/heartbeat-writer.sh
# Add to root crontab
echo '*/5 * * * * /usr/local/bin/heartbeat-writer.sh' | sudo crontab -
```

Step 9 — Deploy FreeOSBot Twin

Build the Twin image and deploy:

```
kubectl apply -f shadow/freeosbot-twin.yaml
```

Verify the failover monitor sidecar is running and logging heartbeat age.

Step 10 — Deploy Watchmen

```
cd watchmen/
docker compose up -d
```

Verify both Watchmen and both Ollama instances start successfully. Pull the LLM model:

```
docker exec watchmen-ollama-cluster ollama pull qwen2.5-coder:7b
docker exec watchmen-ollama-infra ollama pull qwen2.5-coder:7b
```

Step 11 — Configure session exports

```
sudo cp scripts/session-export.sh /usr/local/bin/
sudo chmod +x /usr/local/bin/session-export.sh
# Add to agent user crontab
echo '*/15 * * * * /usr/local/bin/session-export.sh' | crontab -
```

Step 12 — Deploy Plan B watchdog

```
sudo cp scripts/freeosbot-watchdog.sh /usr/local/bin/
sudo chmod +x /usr/local/bin/freeosbot-watchdog.sh
echo '*/5 * * * * /usr/local/bin/freeosbot-watchdog.sh' | sudo crontab -
```

Step 13 — Configure host firewall

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow from <LAN_CIDR> to any port 22
sudo ufw allow from <LAN_CIDR> to any port 18789
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable
```

Step 14 — Run post-deployment health check

Verify all components:

```
# Kubernetes apps
kubectl get applications -n argocd

# Watchmen
docker compose -f watchmen/docker-compose.yml ps

# Heartbeat
mc cat local/shadow-ha/heartbeat-primary.json

# Twin failover monitor
kubectl logs -n shadow -l app=freeosbot-twin -c failover-monitor --tail=5

# Agent
curl -s http://localhost:18789/health
```

All systems should report healthy before going live.

Step 15 — Send test alert and verify escalation loop

Trigger a test alert from a Watchman:

```
docker exec watchmen-healthcloud python3 -c "
import requests
requests.post('http://your-agent:18789/webhook', json={
  'event': 'test', 'severity': 'P2', 'message': 'Post-deployment test alert'
})
"
```

Verify: - Alert received by FreeOSBot on Telegram - AI triage report included - Agent responds appropriately

Deployment complete.

Path B: Greenfield (No Existing Cluster)

Follow the same 15 steps, with these additions before Step 1:

Step 0a — Provision and configure nodes

```
# On each node
sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install -y curl git docker.io python3-pip

# Install k3s on first control plane node
curl -sfL https://get.k3s.io | \
  INSTALL_K3S_VERSION="v1.32.0+k3s1" \
  K3S_TOKEN="<cluster-token>" \
  sh -s - server \
  --cluster-init \
  --disable traefik \
  --write-kubeconfig-mode 644
```

Step 0b — Bootstrap ArgoCD

```
kubectl create namespace argocd
kubectl apply -n argocd -f \
  https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
```

Then proceed from Step 1.

Path C: Standalone (Docker Compose, No Kubernetes)

For evaluation or smaller deployments without Kubernetes.

Requirements: Linux host, Docker, Docker Compose, 8 GB RAM minimum.

Limitations: - No Twin HA (single FreeOSBot instance) - No ArgoCD GitOps (manual deployments) - No Longhorn (local Docker volumes only) - Platform services (Vault, Harbor, etc.) not included

Deploy:

```
git clone https://github.com/freeoscloud/freeosbot
cd freeosbot/standalone
cp .env.example .env
# Edit .env with your tokens and API keys
docker compose up -d
```

All Watchmen and the FreeOSBot agent will start. Heartbeat uses a local file instead of MinIO.

---

4. Post-Deployment Checklist

Item	Verified
All Kubernetes apps Synced/Healthy	☐
TLS certificates issued and valid	☐
FreeOSBot responding on Telegram	☐
Twin heartbeat polling (check logs)	☐
Watchmen sending alerts	☐
Ollama models pulled on both instances	☐
Session export cron running	☐
Heartbeat writer cron running	☐
Plan B watchdog cron running	☐
Firewall rules verified	☐
Velero backup schedule active	☐
Test alert sent and received	☐
Failover test completed	☐
Operator trained on GA protocol	☐
MEMORY.md initialized with site context	☐

---

5. Support & Contact

Channel	Details
Enterprise deployments	hibsen@ehealthbrains.com
Business enquiries	Henrik L. Ibsen, eHealthBrains ApS
Website	https://freeosbot.com
