LexOS Production Deployment Guide

🔱 H100 GPU Infrastructure Deployment

This guide provides comprehensive instructions for deploying LexOS in a production environment optimized for NVIDIA H100 GPUs.

Table of Contents

Prerequisites

Hardware Requirements

- **GPU Nodes**: Minimum 1 node with NVIDIA H100 GPUs (recommended: 2+ nodes for HA)
- **CPU**: 32+ cores per GPU node
- **Memory**: 256GB+ RAM per GPU node
- **Storage**: 2TB+ NVMe SSD per node
- **Network**: 100Gbps+ InfiniBand or Ethernet for multi-node setups

Software Requirements

- **Kubernetes**: v1.28+ with GPU support
- **NVIDIA GPU Operator**: v23.9.0+
- **Container Runtime**: containerd v1.6+ or Docker v20.10+
- **NVIDIA Drivers**: v525.60.13+ (CUDA 12.0+)
- **Helm**: v3.12+
- **kubectl**: v1.28+

Network Requirements

- **Load Balancer**: For ingress traffic
- **DNS**: Proper domain configuration
- **SSL Certificates**: Valid certificates for HTTPS
- **Firewall**: Configured for required ports

Infrastructure Setup

1. Kubernetes Cluster Preparation

```
# Verify cluster is ready
kubectl cluster-info
kubectl get nodes

# Check GPU nodes
kubectl get nodes -l accelerator=nvidia-h100
```

2. GPU Operator Installation

```
# Add NVIDIA Helm repository
helm repo add nvidia https://helm.ngc.nvidia.com/nvidia
helm repo update

# Install GPU Operator
helm install gpu-operator nvidia/gpu-operator \
  --namespace gpu-operator \
  --create-namespace \
  --set driver.enabled=true \
  --set toolkit.enabled=true \
  --set devicePlugin.enabled=true \
  --set dcgmExporter.enabled=true \
  --set gfd.enabled=true \
  --set migManager.enabled=true \
  --set operator.defaultRuntime=containerd

# Verify installation
kubectl get pods -n gpu-operator
```

3. Storage Configuration

```
# Create storage classes (adjust for your provider)
kubectl apply -f k8s/storage.yaml

# Verify storage classes
kubectl get storageclass
```

Security Configuration

1. Secrets Management

⚠️ **CRITICAL: Update all placeholder values in secrets before deployment**

```
# Copy and customize secrets
cp k8s/secrets.yaml k8s/secrets-production.yaml

# Edit secrets with actual values
vim k8s/secrets-production.yaml

# Apply secrets
kubectl apply -f k8s/secrets-production.yaml
```

2. SSL/TLS Certificates

```
# Option 1: Use cert-manager for automatic certificates
helm install cert-manager jetstack/cert-manager \
  --namespace cert-manager \
  --create-namespace \
  --set installCRDs=true

# Option 2: Manual certificate installation
kubectl create secret tls lexos-tls \
  --cert=path/to/certificate.crt \
  --key=path/to/private.key \
  -n lexos
```

3. Network Policies

```
# Apply network policies for security
kubectl apply -f k8s/network-policies.yaml
```

Deployment Process

1. Automated Deployment

```
# Make deployment script executable
chmod +x scripts/deploy-production.sh

# Run deployment with custom image tag
./scripts/deploy-production.sh --image-tag v2.0.0 --perf-test

# Monitor deployment progress
kubectl get pods -n lexos -w
```

2. Manual Deployment Steps

```
# 1. Create namespace
kubectl apply -f k8s/namespace.yaml

# 2. Apply secrets (ensure they're customized)
kubectl apply -f k8s/secrets-production.yaml

# 3. Apply configuration
kubectl apply -f k8s/configmap.yaml

# 4. Deploy storage
kubectl apply -f k8s/storage.yaml

# 5. Deploy Redis cluster
kubectl apply -f k8s/redis-cluster.yaml

# 6. Wait for Redis to be ready
kubectl wait --for=condition=Ready pod -l app=redis-cluster -n lexos --timeout=300s

# 7. Deploy main application
kubectl apply -f k8s/lexos-deployment.yaml

# 8. Deploy autoscaling
kubectl apply -f k8s/hpa.yaml

# 9. Deploy ingress
kubectl apply -f k8s/ingress.yaml

# 10. Deploy monitoring
kubectl apply -f monitoring/
```

3. Verify Deployment

```
# Check deployment status
kubectl rollout status deployment/lexos-api -n lexos

# Check pods
kubectl get pods -n lexos

# Check services
kubectl get services -n lexos

# Check GPU allocation
kubectl describe pods -n lexos -l app=lexos-api | grep -A 5 -B 5 "nvidia.com/gpu"

# Run health check
kubectl run health-check --rm -i --restart=Never --image=curlimages/curl -- \
  curl -f http://lexos-api-service.lexos.svc.cluster.local:8000/health
```

Monitoring & Observability

1. Prometheus & Grafana Setup

```
# Install Prometheus Operator
helm install prometheus-operator prometheus-community/kube-prometheus-stack \
  --namespace monitoring \
  --create-namespace \
  --set grafana.adminPassword=your-secure-password

# Apply custom monitoring rules
kubectl apply -f monitoring/prometheus-rules.yaml

# Import Grafana dashboard
# Use the dashboard JSON from monitoring/grafana-dashboard.json
```

2. OpenTelemetry Configuration

```
# Deploy OpenTelemetry Collector
kubectl apply -f monitoring/otel-config.yaml

# Verify collector is running
kubectl get pods -n lexos -l app=otel-collector
```

3. Log Aggregation

```
# Deploy Fluent Bit for log collection
helm install fluent-bit fluent/fluent-bit \
  --namespace logging \
  --create-namespace \
  --set config.outputs="[OUTPUT]\n    Name forward\n    Match *\n    Host
elasticsearch\n    Port 9200"
```

4. Alerting Setup

```
# Configure Alertmanager
kubectl apply -f monitoring/alertmanager-config.yaml

# Test alerts
kubectl apply -f monitoring/test-alert.yaml
```

Mobile App Deployment

1. Build Configuration

```
cd mobile/lexos-mobile

# Install dependencies
npm install

# Configure environment
cp .env.example .env.production
# Edit .env.production with production values

# Build for Android
eas build --platform android --profile production

# Build for iOS
eas build --platform ios --profile production
```

2. App Store Deployment

```
# Submit to Google Play Store
eas submit --platform android --profile production

# Submit to Apple App Store
eas submit --platform ios --profile production
```

Maintenance & Operations

1. Scaling Operations

```
# Manual scaling
kubectl scale deployment lexos-api --replicas=3 -n lexos

# Update HPA settings
kubectl patch hpa lexos-api-hpa -n lexos -p '{"spec":{"maxReplicas":5}}'

# Check scaling status
kubectl get hpa -n lexos
```

2. Updates & Rollouts

```
# Update deployment image
kubectl set image deployment/lexos-api lexos-api=ghcr.io/lexhelios/lexworking:v2.1.0 -
n lexos

# Monitor rollout
kubectl rollout status deployment/lexos-api -n lexos

# Rollback if needed
kubectl rollout undo deployment/lexos-api -n lexos
```

3. Backup Operations

```
# Backup using Velero
velero backup create lexos-backup-$(date +%Y%m%d) --include-namespaces lexos

# Backup secrets manually
kubectl get secrets -n lexos -o yaml > lexos-secrets-backup.yaml

# Backup persistent data
kubectl exec -n lexos deployment/lexos-api -- tar czf - /app/data | gzip > lexos-data-
backup.tar.gz
```

4. Certificate Renewal

```
# Check certificate expiry
kubectl get certificate -n lexos

# Force renewal (cert-manager)
kubectl annotate certificate lexos-tls cert-manager.io/issue-temporary-certificate="" -
n lexos

# Manual certificate update
kubectl create secret tls lexos-tls --cert=new-cert.crt --key=new-key.key -n lexos --
dry-run=client -o yaml | kubectl apply -f -
```

Troubleshooting

1. Common Issues

Pod Stuck in Pending State

```
# Check node resources
kubectl describe nodes

# Check GPU availability
kubectl get nodes -o json | jq '.items[].status.allocatable."nvidia.com/gpu"'

# Check pod events
kubectl describe pod <pod-name> -n lexos
```

GPU Not Detected

```
# Check GPU operator pods
kubectl get pods -n gpu-operator

# Check device plugin logs
kubectl logs -n gpu-operator -l app=nvidia-device-plugin-daemonset

# Verify GPU nodes
kubectl get nodes -l accelerator=nvidia-h100 -o wide
```

High Memory Usage

```
# Check memory usage
kubectl top pods -n lexos

# Check GPU memory
kubectl exec -n lexos deployment/lexos-api -- nvidia-smi

# Adjust memory limits
kubectl patch deployment lexos-api -n lexos -p '{"spec":{"template":{"spec":{"contain-
ers":[{"name":"lexos-api","resources":{"limits":{"memory":"128Gi"}}}]}}}}'
```

2. Performance Issues

Low GPU Utilization

```
# Check vLLM configuration
kubectl logs -n lexos deployment/lexos-api -c lexos-api | grep vllm

# Verify tensor parallelism
kubectl exec -n lexos deployment/lexos-api -- python -c "
from server.gpu import get_h100_optimizer
optimizer = get_h100_optimizer()
print(optimizer.get_gpu_info())
"

# Adjust batch size
kubectl set env deployment/lexos-api VLLM_MAX_NUM_SEQS=512 -n lexos
```

High Latency

```
# Check request queue
kubectl exec -n lexos deployment/lexos-api -- curl localhost:8002/metrics | grep queue

# Scale up replicas
kubectl scale deployment lexos-api --replicas=3 -n lexos

# Check network latency
kubectl run network-test --rm -i --restart=Never --image=nicolaka/netshoot -- \
  ping lexos-api-service.lexos.svc.cluster.local
```

3. Debugging Commands

```
# Get comprehensive cluster info
kubectl cluster-info dump --output-directory=cluster-dump

# Check all resources in namespace
kubectl get all -n lexos

# Get detailed pod information
kubectl describe pods -n lexos

# Check logs from all containers
kubectl logs -n lexos -l app=lexos-api --all-containers=true --tail=100

# Execute into pod for debugging
kubectl exec -it -n lexos deployment/lexos-api -- /bin/bash

# Check GPU status from within pod
kubectl exec -n lexos deployment/lexos-api -- nvidia-smi

# Check Python environment
kubectl exec -n lexos deployment/lexos-api -- python -c "
import torch
print(f'PyTorch version: {torch.__version__}')
print(f'CUDA available: {torch.cuda.is_available()}')
print(f'GPU count: {torch.cuda.device_count()}')
for i in range(torch.cuda.device_count()):
    print(f'GPU {i}: {torch.cuda.get_device_name(i)}')
"
```

Performance Optimization

1. H100 GPU Optimization

```
# Check current GPU configuration
kubectl exec -n lexos deployment/lexos-api -- python -c "
from server.gpu import get_h100_optimizer
optimizer = get_h100_optimizer()
print('GPU Info:', optimizer.get_gpu_info())
print('Health:', optimizer.monitor_gpu_health())
"

# Run GPU benchmark
kubectl exec -n lexos deployment/lexos-api -- python -c "
from server.gpu import get_h100_optimizer
optimizer = get_h100_optimizer()
results = optimizer.benchmark_gpu_performance(duration=30)
print('Benchmark Results:', results)
"
```

2. vLLM Optimization

```
# Check vLLM metrics
kubectl exec -n lexos deployment/lexos-api -- curl localhost:8001/metrics

# Optimize for throughput
kubectl set env deployment/lexos-api \
  VLLM_MAX_NUM_SEQS=512 \
  VLLM_MAX_NUM_BATCHED_TOKENS=16384 \
  -n lexos

# Optimize for latency
kubectl set env deployment/lexos-api \
  VLLM_MAX_NUM_SEQS=128 \
  VLLM_ENFORCE_EAGER=true \
  -n lexos
```

3. Resource Optimization

```
# Adjust resource requests/limits
kubectl patch deployment lexos-api -n lexos -p '
{
  "spec": {
    "template": {
      "spec": {
        "containers": [
          {
            "name": "lexos-api",
            "resources": {
              "requests": {
                "memory": "64Gi",
                "cpu": "16",
                "nvidia.com/gpu": "8"
              },
              "limits": {
                "memory": "128Gi",
                "cpu": "32",
                "nvidia.com/gpu": "8"
              }
            }
          }
        ]
      }
    }
  }
}'

# Enable CPU affinity for better performance
kubectl patch deployment lexos-api -n lexos -p '
{
  "spec": {
    "template": {
      "spec": {
        "containers": [
          {
            "name": "lexos-api",
            "env": [
              {
                "name": "OMP_NUM_THREADS",
                "value": "16"
              },
              {
                "name": "CUDA_VISIBLE_DEVICES",
                "value": "0,1,2,3,4,5,6,7"
              }
            ]
          }
        ]
      }
    }
  }
}'
```

Disaster Recovery

1. Backup Strategy

```
# Daily automated backups
cat << EOF | kubectl apply -f -
apiVersion: batch/v1
kind: CronJob
metadata:
  name: lexos-backup
  namespace: lexos
spec:
  schedule: "0 2 * * *"  # Daily at 2 AM
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: backup
            image: velero/velero:v1.12.0
            command:
            - /bin/sh
            - -c
            - |
              velero backup create lexos-daily-\$(date +%Y%m%d) \
                --include-namespaces lexos \
                --storage-location default \
                --volume-snapshot-locations default \
                --ttl 720h0m0s
          restartPolicy: OnFailure
EOF
```

2. Recovery Procedures

```
# List available backups
velero backup get

# Restore from backup
velero restore create lexos-restore-$(date +%Y%m%d) \
  --from-backup lexos-daily-20240101 \
  --include-namespaces lexos

# Monitor restore progress
velero restore describe lexos-restore-$(date +%Y%m%d)

# Verify restored deployment
kubectl get all -n lexos
```

3. Multi-Region Setup

```
# Set up cross-region replication
# Configure backup storage in multiple regions
velero backup-location create backup-region-2 \
  --provider aws \
  --bucket lexos-backups-region-2 \
  --config region=us-east-1

# Schedule cross-region backups
velero schedule create lexos-cross-region \
  --schedule="0 6 * * *" \
  --storage-location backup-region-2 \
  --include-namespaces lexos
```

Security Checklist

- [ ] All secrets updated with production values
- [ ] SSL/TLS certificates configured and valid
- [ ] Network policies applied
- [ ] RBAC configured with least privilege
- [ ] Container images scanned for vulnerabilities
- [ ] Security headers enabled
- [ ] Rate limiting configured
- [ ] Monitoring and alerting active
- [ ] Backup and recovery tested
- [ ] Access logs enabled
- [ ] Encryption at rest enabled
- [ ] Regular security updates scheduled

Support & Maintenance

Regular Maintenance Tasks

1. **Weekly**:
   - Review monitoring dashboards
   - Check certificate expiry dates
   - Review security alerts
   - Update container images

2. **Monthly**:
   - Performance optimization review
   - Capacity planning assessment
   - Security vulnerability scan
   - Backup restoration test

3. **Quarterly**:
   - Disaster recovery drill
   - Security audit
   - Performance benchmarking
   - Documentation updates

Contact Information

- **Technical Support**: support@lexos.ai
- **Security Issues**: security@lexos.ai
- **Emergency Contact**: +1-XXX-XXX-XXXX

🔱 LexOS Production Deployment Guide - Optimized for H100 GPU Infrastructure 🔱

Last Updated: August 2025
Version: 2.0.0