Monitoring Guide

Overview

This guide provides comprehensive instructions for monitoring the HX-Infrastructure-Ansible automation platform, including system metrics, application performance, security events, and operational health.

Monitoring Architecture

Components

- **Prometheus**: Metrics collection and storage
- **Grafana**: Visualization and dashboards
- **Alertmanager**: Alert routing and notification
- **Node Exporter**: System metrics collection
- **Application Exporters**: Custom metrics collection
- **Log Aggregation**: Centralized logging system

Data Flow

```
Applications → Exporters → Prometheus → Grafana
                    ↓
              Alertmanager → Notifications
```

Prometheus Configuration

Installation and Setup

```
# Deploy Prometheus
ansible-playbook playbooks/monitoring/deploy_prometheus.yml -i inventory.yml

# Configure Prometheus
ansible-playbook playbooks/monitoring/configure_prometheus.yml -i inventory.yml \
  --extra-vars "scrape_interval=15s retention_time=30d"

# Verify Prometheus installation
curl http://prometheus-server:9090/api/v1/status/config
```

Prometheus Configuration File

```yaml
# /etc/prometheus/prometheus.yml
global:
  scrape_interval: 15s
  evaluation_interval: 15s
  external_labels:
    cluster: 'production'
    environment: 'prod'

rule_files:
  - "/etc/prometheus/rules/*.yml"

alerting:
  alertmanagers:
    - static_configs:
        - targets:
          - alertmanager:9093

scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:9090']

  - job_name: 'node-exporter'
    static_configs:
      - targets:
        - 'server1:9100'
        - 'server2:9100'
        - 'server3:9100'

  - job_name: 'ansible-automation-platform'
    static_configs:
      - targets:
        - 'aap-controller:443'
    metrics_path: '/api/v2/metrics/'
    scheme: https
    bearer_token: 'your-aap-token'

  - job_name: 'application'
    static_configs:
      - targets:
        - 'app-server:8080'
    metrics_path: '/metrics'
```

Adding New Targets

```bash
# Add new monitoring target
ansible-playbook playbooks/monitoring/add_target.yml -i inventory.yml \
  --extra-vars "target_host=new-server target_port=9100 job_name=node-exporter"

# Reload Prometheus configuration
curl -X POST http://prometheus-server:9090/-/reload
```

Grafana Configuration

Installation and Setup

```
# Deploy Grafana
ansible-playbook playbooks/monitoring/deploy_grafana.yml -i inventory.yml

# Configure Grafana datasources
ansible-playbook playbooks/monitoring/configure_grafana_datasources.yml -i invent-
ory.yml

# Import dashboards
ansible-playbook playbooks/monitoring/import_dashboards.yml -i inventory.yml
```

Dashboard Management

```
# Export dashboard
curl -H "Authorization: Bearer $GRAFANA_API_KEY" \
  http://grafana-server:3000/api/dashboards/uid/dashboard-uid

# Import dashboard
curl -X POST -H "Content-Type: application/json" \
  -H "Authorization: Bearer $GRAFANA_API_KEY" \
  -d @dashboard.json \
  http://grafana-server:3000/api/dashboards/db

# List all dashboards
curl -H "Authorization: Bearer $GRAFANA_API_KEY" \
  http://grafana-server:3000/api/search
```

Key Dashboards

System Overview Dashboard

- **CPU Usage**: Overall and per-core utilization
- **Memory Usage**: Available, used, and swap utilization
- **Disk Usage**: Space utilization and I/O metrics
- **Network**: Traffic, errors, and connection counts
- **Load Average**: System load over time

Application Performance Dashboard

- **Response Times**: API endpoint response times
- **Throughput**: Requests per second
- **Error Rates**: HTTP error codes and application errors
- **Database Performance**: Query times and connection pools
- **Cache Hit Rates**: Redis/Memcached performance

Infrastructure Dashboard

- **Service Status**: Up/down status of critical services
- **Deployment Status**: Recent deployments and their status
- **Security Events**: Failed logins, suspicious activities
- **Backup Status**: Backup success/failure rates
- **Certificate Expiration**: SSL certificate expiry dates

Alerting Configuration

Alertmanager Setup

```
# Deploy Alertmanager
ansible-playbook playbooks/monitoring/deploy_alertmanager.yml -i inventory.yml

# Configure alert routing
ansible-playbook playbooks/monitoring/configure_alertmanager.yml -i inventory.yml \
  --extra-vars "slack_webhook_url=https://hooks.slack.com/services/..."
```

Alert Rules

```yaml
# /etc/prometheus/rules/alerts.yml
groups:
  - name: system_alerts
    rules:
      - alert: HighCPUUsage
        expr: 100 - (avg by(instance) (irate(node_cpu_seconds_total{mode="idle"}[5m])) * 100) > 80
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "High CPU usage on {{ $labels.instance }}"
          description: "CPU usage is above 80% for more than 5 minutes"

      - alert: HighMemoryUsage
        expr: (node_memory_MemTotal_bytes - node_memory_MemAvailable_bytes) / node_memory_MemTotal_bytes * 100 > 90
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "High memory usage on {{ $labels.instance }}"
          description: "Memory usage is above 90% for more than 5 minutes"

      - alert: DiskSpaceLow
        expr: (node_filesystem_avail_bytes / node_filesystem_size_bytes) * 100 < 10
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "Low disk space on {{ $labels.instance }}"
          description: "Disk space is below 10% on {{ $labels.mountpoint }}"

      - alert: ServiceDown
        expr: up == 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "Service {{ $labels.job }} is down"
          description: "Service {{ $labels.job }} on {{ $labels.instance }} has been down for more than 1 minute"

  - name: application_alerts
    rules:
      - alert: HighErrorRate
        expr: rate(http_requests_total{status=~"5.."}[5m]) / rate(http_requests_total[5m]) * 100 > 5
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "High error rate on {{ $labels.instance }}"
          description: "Error rate is above 5% for more than 5 minutes"

      - alert: SlowResponseTime
        expr: histogram_quantile(0.95, rate(http_request_duration_seconds_bucket[5m])) > 1
        for: 5m
        labels:
          severity: warning
        annotations:
```

```
          summary: "Slow response time on {{ $labels.instance }}"
          description: "95th percentile response time is above 1 second"
```

Notification Channels

```
# /etc/alertmanager/alertmanager.yml
global:
  smtp_smarthost: 'localhost:587'
  smtp_from: 'alerts@company.com'

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 1h
  receiver: 'web.hook'
  routes:
    - match:
        severity: critical
      receiver: 'critical-alerts'
    - match:
        severity: warning
      receiver: 'warning-alerts'

receivers:
  - name: 'web.hook'
    webhook_configs:
      - url: 'http://127.0.0.1:5001/'

  - name: 'critical-alerts'
    slack_configs:
      - api_url: 'https://hooks.slack.com/services/...'
        channel: '#critical-alerts'
        title: 'Critical Alert'
        text: '{{ range .Alerts }}{{ .Annotations.summary }}{{ end }}'
    email_configs:
      - to: 'oncall@company.com'
        subject: 'Critical Alert: {{ .GroupLabels.alertname }}'
        body: '{{ range .Alerts }}{{ .Annotations.description }}{{ end }}'

  - name: 'warning-alerts'
    slack_configs:
      - api_url: 'https://hooks.slack.com/services/...'
        channel: '#alerts'
        title: 'Warning Alert'
        text: '{{ range .Alerts }}{{ .Annotations.summary }}{{ end }}'
```

System Monitoring

Node Exporter Deployment

```
# Deploy Node Exporter
ansible-playbook playbooks/monitoring/deploy_node_exporter.yml -i inventory.yml

# Configure Node Exporter
ansible-playbook playbooks/monitoring/configure_node_exporter.yml -i inventory.yml \
  --extra-vars "enable_collectors=['cpu','memory','disk','network']"

# Verify Node Exporter
curl http://target-server:9100/metrics
```

Custom Metrics Collection

```
# Deploy custom exporter
ansible-playbook playbooks/monitoring/deploy_custom_exporter.yml -i inventory.yml \
  --extra-vars "exporter_name=application_exporter exporter_port=8080"

# Configure custom metrics
ansible-playbook playbooks/monitoring/configure_custom_metrics.yml -i inventory.yml
```

System Health Checks

```
# Automated health check
ansible-playbook playbooks/monitoring/health_check.yml -i inventory.yml

# Generate health report
./scripts/monitoring/generate_health_report.sh

# Check critical services
ansible all -i inventory.yml -m systemd -a "name=nginx"
ansible all -i inventory.yml -m systemd -a "name=mysql"
ansible all -i inventory.yml -m systemd -a "name=redis"
```

Application Monitoring

Application Metrics

```
# Example Python application metrics
from prometheus_client import Counter, Histogram, Gauge, start_http_server
import time

# Metrics definitions
REQUEST_COUNT = Counter('app_requests_total', 'Total app requests', ['method', 'end-
point'])
REQUEST_LATENCY = Histogram('app_request_duration_seconds', 'Request latency')
ACTIVE_USERS = Gauge('app_active_users', 'Number of active users')

# Instrument your application
@REQUEST_LATENCY.time()
def process_request(request):
    REQUEST_COUNT.labels(method=request.method, endpoint=request.endpoint).inc()
    # Process request
    return response

# Start metrics server
start_http_server(8000)
```

Database Monitoring

```
# Deploy database exporter
ansible-playbook playbooks/monitoring/deploy_db_exporter.yml -i inventory.yml \
  --extra-vars "db_type=mysql db_host=db-server db_port=3306"

# Configure database monitoring
ansible-playbook playbooks/monitoring/configure_db_monitoring.yml -i inventory.yml

# Check database performance
mysql -e "SHOW PROCESSLIST;"
mysql -e "SHOW ENGINE INNODB STATUS;"
```

Web Server Monitoring

```
# Configure Nginx monitoring
ansible-playbook playbooks/monitoring/configure_nginx_monitoring.yml -i inventory.yml

# Configure Apache monitoring
ansible-playbook playbooks/monitoring/configure_apache_monitoring.yml -i inventory.yml

# Check web server status
curl http://web-server/nginx_status
curl http://web-server/server-status
```

Log Monitoring

Centralized Logging Setup

```
# Deploy ELK stack
ansible-playbook playbooks/monitoring/deploy_elk_stack.yml -i inventory.yml

# Configure log forwarding
ansible-playbook playbooks/monitoring/configure_log_forwarding.yml -i inventory.yml

# Set up log parsing
ansible-playbook playbooks/monitoring/configure_log_parsing.yml -i inventory.yml
```

Log Analysis

```
# Search logs
curl -X GET "elasticsearch:9200/logs-*/_search" -H 'Content-Type: application/json' -
d'
{
  "query": {
    "match": {
      "message": "ERROR"
    }
  }
}'

# Aggregate logs
curl -X GET "elasticsearch:9200/logs-*/_search" -H 'Content-Type: application/json' -
d'
{
  "aggs": {
    "error_count": {
      "terms": {
        "field": "level.keyword"
      }
    }
  }
}'
```

Log Retention

```
# Configure log retention
ansible-playbook playbooks/monitoring/configure_log_retention.yml -i inventory.yml \
  --extra-vars "retention_days=30"

# Clean old logs
ansible-playbook playbooks/monitoring/cleanup_old_logs.yml -i inventory.yml
```

Security Monitoring

Security Event Monitoring

```
# Deploy security monitoring
ansible-playbook playbooks/monitoring/deploy_security_monitoring.yml -i inventory.yml

# Configure security alerts
ansible-playbook playbooks/monitoring/configure_security_alerts.yml -i inventory.yml

# Monitor failed logins
ansible all -i inventory.yml -m shell -a "grep 'Failed password' /var/log/auth.log |
tail -10"
```

Intrusion Detection

```
# Deploy OSSEC
ansible-playbook playbooks/monitoring/deploy_ossec.yml -i inventory.yml

# Configure IDS rules
ansible-playbook playbooks/monitoring/configure_ids_rules.yml -i inventory.yml

# Check IDS alerts
ansible all -i inventory.yml -m shell -a "tail -f /var/ossec/logs/alerts/alerts.log"
```

Compliance Monitoring

```
# Monitor compliance status
ansible-playbook playbooks/monitoring/monitor_compliance.yml -i inventory.yml

# Generate compliance report
./scripts/monitoring/generate_compliance_report.sh

# Check security configurations
ansible-playbook playbooks/monitoring/check_security_configs.yml -i inventory.yml
```

Performance Monitoring

Performance Metrics

```
# Collect performance metrics
ansible-playbook playbooks/monitoring/collect_performance_metrics.yml -i inventory.yml

# Generate performance report
./scripts/monitoring/generate_performance_report.sh

# Monitor resource usage
ansible all -i inventory.yml -m shell -a "top -bn1 | head -20"
```

Capacity Planning

```
# Analyze capacity trends
ansible-playbook playbooks/monitoring/analyze_capacity_trends.yml -i inventory.yml

# Generate capacity report
./scripts/monitoring/generate_capacity_report.sh

# Predict resource needs
./scripts/monitoring/predict_resource_needs.py
```

Performance Optimization

```
# Identify performance bottlenecks
ansible-playbook playbooks/monitoring/identify_bottlenecks.yml -i inventory.yml

# Optimize system performance
ansible-playbook playbooks/monitoring/optimize_performance.yml -i inventory.yml

# Verify optimization results
ansible-playbook playbooks/monitoring/verify_optimization.yml -i inventory.yml
```

Monitoring Automation

Automated Monitoring Tasks

```
# Schedule monitoring tasks
ansible-playbook playbooks/monitoring/schedule_monitoring_tasks.yml -i inventory.yml

# Automated report generation
ansible-playbook playbooks/monitoring/automate_reports.yml -i inventory.yml

# Self-healing automation
ansible-playbook playbooks/monitoring/configure_self_healing.yml -i inventory.yml
```

Monitoring as Code

```yaml
# monitoring-config.yml
monitoring:
  prometheus:
    scrape_interval: 15s
    retention: 30d
    targets:
      - job: node-exporter
        targets: ['server1:9100', 'server2:9100']
      - job: application
        targets: ['app1:8080', 'app2:8080']

  grafana:
    dashboards:
      - system-overview
      - application-performance
      - security-monitoring

  alerts:
    - name: high-cpu
      threshold: 80
      duration: 5m
    - name: low-disk
      threshold: 10
      duration: 5m
```

Troubleshooting Monitoring

Common Issues

```bash
# Check Prometheus targets
curl http://prometheus:9090/api/v1/targets

# Verify Grafana datasource
curl -H "Authorization: Bearer $API_KEY" http://grafana:3000/api/datasources

# Test alerting
curl -X POST http://alertmanager:9093/api/v1/alerts -d @test-alert.json

# Check exporter status
systemctl status node_exporter
systemctl status prometheus
systemctl status grafana-server
```

Monitoring Health

```bash
# Monitor the monitoring system
ansible-playbook playbooks/monitoring/monitor_monitoring.yml -i inventory.yml

# Check monitoring system health
./scripts/monitoring/check_monitoring_health.sh

# Restart monitoring services
ansible-playbook playbooks/monitoring/restart_monitoring_services.yml -i inventory.yml
```

Best Practices

Monitoring Strategy

- **Monitor what matters**: Focus on business-critical metrics
- **Set meaningful alerts**: Avoid alert fatigue
- **Use SLIs and SLOs**: Define service level objectives
- **Implement progressive alerting**: Escalate based on severity
- **Regular review**: Continuously improve monitoring

Dashboard Design

- **Clear visualization**: Use appropriate chart types
- **Logical grouping**: Organize related metrics
- **Consistent naming**: Use standard naming conventions
- **Responsive design**: Ensure dashboards work on all devices
- **Documentation**: Document dashboard purpose and metrics

Alert Management

- **Actionable alerts**: Every alert should require action
- **Clear descriptions**: Provide context and remediation steps
- **Appropriate severity**: Match alert severity to impact
- **Escalation paths**: Define clear escalation procedures
- **Alert hygiene**: Regularly review and clean up alerts

Related Documentation

- Deployment Runbook (../runbooks/DEPLOYMENT_RUNBOOK.md)
- Troubleshooting Guide (../runbooks/TROUBLESHOOTING_GUIDE.md)
- Security Procedures (../runbooks/SECURITY_PROCEDURES.md)
- Performance Tuning Guide (PERFORMANCE_TUNING.md)