HX Infrastructure Ansible - Team Training Manual

Table of Contents

Introduction

Welcome to the HX Infrastructure Ansible platform! This comprehensive training manual will guide you through all aspects of our enterprise-grade infrastructure automation system. By the end of this training, you will be proficient in managing, deploying, and troubleshooting our infrastructure automation platform.

Learning Objectives

After completing this training, you will be able to:
- Navigate and use the HX Infrastructure Ansible platform effectively
- Perform routine operational tasks with confidence
- Troubleshoot common issues independently
- Follow emergency procedures during incidents
- Implement best practices for security and compliance
- Contribute to continuous improvement initiatives

Prerequisites

Required Knowledge

- Basic Linux system administration
- Understanding of networking concepts (TCP/IP, DNS, HTTP/HTTPS)
- Familiarity with command-line interfaces
- Basic understanding of cloud computing concepts
- Version control systems (Git) fundamentals

Required Tools

- SSH client (PuTTY, OpenSSH, or similar)
- Text editor (VS Code, vim, nano)

- Web browser (Chrome, Firefox, Safari)
- VPN client (if accessing from external networks)

Access Requirements

- VPN access to corporate network
- SSH access to infrastructure servers
- Grafana dashboard access credentials
- Ansible Automation Platform access
- Emergency contact information

Getting Started

Environment Overview

Our infrastructure automation platform consists of several key components:

Core Components

- **Ansible Automation Platform**: Central automation engine
- **Grafana Dashboards**: Monitoring and visualization
- **Prometheus**: Metrics collection and alerting
- **GitLab**: Source code management and CI/CD
- **Vault**: Secrets management

Environment Types

- **Development**: Testing and development work
- **Staging**: Pre-production validation
- **Production**: Live customer-facing systems

Initial Setup

1. Access Verification

```
# Test SSH access to control node
ssh username@ansible-control.company.com

# Verify Ansible installation
ansible --version

# Test inventory access
ansible all --list-hosts
```

2. Dashboard Access

- Navigate to: https://grafana.company.com
- Login with provided credentials
- Verify access to operational dashboards

3. Repository Access

```
# Clone the main repository
git clone https://gitlab.company.com/infrastructure/hx-infrastructure-ansible.git

# Navigate to project directory
cd hx-infrastructure-ansible

# Review project structure
ls -la
```

Project Structure Overview

```
hx-infrastructure-ansible/
├── playbooks/           # Main automation playbooks
├── roles/              # Reusable automation roles
├── environments/       # Environment-specific configurations
├── docs/               # Documentation and guides
├── scripts/            # Utility scripts
├── tests/              # Testing frameworks
└── monitoring/         # Monitoring configurations
```

Core Concepts

Ansible Fundamentals

Inventory Management

Inventory files define the systems we manage:

```
[web_servers]
web01.company.com
web02.company.com

[database_servers]
db01.company.com
db02.company.com

[all:vars]
ansible_user=automation
ansible_ssh_private_key_file=/path/to/key
```

Playbook Structure

Playbooks define automation tasks:

```
---
- name: Deploy web application
  hosts: web_servers
  become: yes
  roles:
    - common
    - web_server
    - application_deployment
  vars:
    app_version: "{{ version | default('latest') }}"
```

Role Organization

Roles provide reusable automation components:

```
roles/web_server/
├── tasks/main.yml      # Main task definitions
├── handlers/main.yml   # Event handlers
├── templates/          # Configuration templates
├── files/              # Static files
├── vars/main.yml       # Role variables
└── defaults/main.yml   # Default values
```

Infrastructure Components

Blue-Green Deployment

Zero-downtime deployment strategy:
- **Blue Environment**: Currently active production
- **Green Environment**: New version being deployed
- **Traffic Switch**: Seamless transition between environments
- **Rollback Capability**: Quick reversion if issues occur

Monitoring Stack

Comprehensive monitoring solution:
- **Prometheus**: Metrics collection and storage
- **Grafana**: Visualization and dashboards
- **AlertManager**: Alert routing and management
- **Node Exporter**: System metrics collection

Backup and Recovery

Enterprise-grade data protection:
- **Automated Backups**: Scheduled data protection
- **Multi-destination Storage**: Local and cloud backup targets
- **Integrity Validation**: Automated backup verification
- **Recovery Testing**: Regular restoration validation

Daily Operations

Routine Tasks

1. System Health Checks

```
# Check overall system status
ansible all -m ping

# Verify service status
ansible web_servers -m service -a "name=nginx state=started"

# Check disk space
ansible all -m shell -a "df -h"
```

2. Dashboard Monitoring

Daily dashboard review checklist:
- [ ] System availability metrics

- [ ] Performance indicators within thresholds
- [ ] No critical alerts active
- [ ] Backup completion status
- [ ] Security event summary

3. Log Review

```
# Check system logs for errors
ansible all -m shell -a "journalctl --since '1 hour ago' --priority=err"

# Review application logs
ansible web_servers -m shell -a "tail -n 100 /var/log/application.log"
```

Deployment Procedures

Standard Deployment Process

1. **Pre-deployment Validation**
   `bash`
   `# Verify environment readiness`
   `ansible-playbook playbooks/pre_deployment_check.yml -e environment=production`

2. **Application Deployment**
   `bash`
   `# Deploy using blue-green strategy`
   `ansible-playbook playbooks/deploy_application.yml \`
   `-e environment=production \`
   `-e version=v2.1.0 \`
   `-e deployment_strategy=blue_green`

3. **Post-deployment Validation**
   `bash`
   `# Verify deployment success`
   `ansible-playbook playbooks/post_deployment_check.yml -e environment=production`

Emergency Deployment

For critical fixes requiring immediate deployment:

```
# Emergency deployment with reduced validation
ansible-playbook playbooks/emergency_deploy.yml \
  -e environment=production \
  -e version=hotfix-v2.1.1 \
  -e skip_validation=false \
  -e emergency_contact="ops@company.com"
```

Maintenance Activities

Weekly Maintenance

- System updates and patches
- Backup validation and testing
- Performance review and optimization
- Security scan and vulnerability assessment

Monthly Maintenance

- Capacity planning review
- Disaster recovery testing
- Documentation updates
- Team training and knowledge sharing

Advanced Procedures

Blue-Green Deployment Management

Deployment Workflow

1. **Environment Preparation**
   ```
   bash
     # Prepare green environment
     ansible-playbook roles/blue_green_deployment/playbooks/prepare_environment.yml \
       -e target_environment=green \
       -e application_version=v2.2.0
   ```

2. **Health Validation**
   ```
   bash
     # Comprehensive health checks
     ansible-playbook roles/blue_green_deployment/playbooks/health_checks.yml \
       -e target_environment=green
   ```

3. **Traffic Switch**
   ```
   bash
     # Switch traffic to green environment
     ansible-playbook roles/blue_green_deployment/playbooks/switch_traffic.yml \
       -e from_environment=blue \
       -e to_environment=green
   ```

4. **Rollback (if needed)**
   ```
   bash
     # Emergency rollback to blue environment
     ansible-playbook roles/blue_green_deployment/playbooks/rollback.yml \
       -e target_environment=blue \
       -e reason="Performance degradation detected"
   ```

Disaster Recovery Procedures

Backup Validation

```
# Validate recent backups
ansible-playbook roles/disaster_recovery/playbooks/validate_backups.yml \
  -e validation_count=3 \
  -e backup_age_hours=24
```

Recovery Simulation

```
# Perform recovery simulation
ansible-playbook roles/disaster_recovery/playbooks/recovery_simulation.yml \
  -e simulation_environment=staging \
  -e recovery_point="2024-01-15 14:30:00"
```

Full Recovery Process

```
# Execute full disaster recovery
ansible-playbook roles/disaster_recovery/playbooks/full_recovery.yml \
  -e recovery_environment=production \
  -e backup_source=s3://company-backups/latest \
  -e notification_email=emergency@company.com
```

Performance Optimization

System Performance Analysis

```
# Collect performance metrics
ansible-playbook roles/performance_optimization/playbooks/performance_analysis.yml \
  -e analysis_duration=3600 \
  -e metrics_output=/tmp/performance_report.json
```

Auto-scaling Configuration

```
# Configure auto-scaling policies
ansible-playbook roles/performance_optimization/playbooks/configure_autoscaling.yml \
  -e min_instances=2 \
  -e max_instances=10 \
  -e scale_out_threshold=70 \
  -e scale_in_threshold=30
```

Troubleshooting Guide

Common Issues and Solutions

1. Ansible Connection Issues

Problem: Cannot connect to target hosts

```
UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh"}
```

Solution:

```
# Check SSH connectivity
ssh -i /path/to/key user@target-host

# Verify inventory configuration
ansible-inventory --list

# Test with verbose output
ansible target-host -m ping -vvv
```

2. Playbook Execution Failures

Problem: Tasks failing during playbook execution

Diagnostic Steps:

```
# Run with increased verbosity
ansible-playbook playbook.yml -vvv

# Check syntax
ansible-playbook playbook.yml --syntax-check

# Dry run to identify issues
ansible-playbook playbook.yml --check
```

3. Service Deployment Issues

Problem: Application not starting after deployment

Investigation Process:

```
# Check service status
ansible web_servers -m systemd -a "name=myapp state=started"

# Review service logs
ansible web_servers -m shell -a "journalctl -u myapp --since '10 minutes ago'"

# Verify configuration files
ansible web_servers -m shell -a "nginx -t"
```

4. Performance Degradation

Problem: System performance below expected levels

Analysis Steps:

```
# Check system resources
ansible all -m shell -a "top -bn1 | head -20"

# Monitor network connectivity
ansible all -m shell -a "netstat -tuln"

# Review application metrics
# Access Grafana dashboard for detailed analysis
```

Escalation Procedures

Level 1: Self-Resolution (0-30 minutes)

- Review documentation and troubleshooting guides
- Check monitoring dashboards for obvious issues
- Attempt standard resolution procedures
- Document findings and actions taken

Level 2: Team Escalation (30-60 minutes)

- Contact team lead or senior team member
- Provide detailed issue description and attempted solutions

- Collaborate on advanced troubleshooting
- Consider temporary workarounds

Level 3: Management Escalation (60+ minutes)

- Notify management of ongoing critical issue
- Engage vendor support if applicable
- Consider emergency procedures
- Prepare incident post-mortem documentation

Emergency Procedures

Incident Response Framework

Incident Classification

- **P1 (Critical)**: Complete service outage, data loss risk
- **P2 (High)**: Significant performance degradation, partial outage
- **P3 (Medium)**: Minor issues, workarounds available
- **P4 (Low)**: Cosmetic issues, no business impact

Response Timeline

- **P1**: 15 minutes initial response, 1 hour resolution target
- **P2**: 30 minutes initial response, 4 hours resolution target
- **P3**: 2 hours initial response, 24 hours resolution target
- **P4**: Next business day response, 1 week resolution target

Emergency Contacts

Primary Contacts

- **Team Lead**: +1-555-0101 (24/7)
- **Infrastructure Manager**: +1-555-0102 (Business hours)
- **On-call Engineer**: +1-555-0103 (24/7 rotation)

Escalation Contacts

- **Director of Engineering**: +1-555-0201
- **CTO**: +1-555-0301
- **Emergency Hotline**: +1-555-9911

Critical Procedures

Emergency Rollback

```
# Immediate rollback to last known good state
ansible-playbook playbooks/emergency_rollback.yml \
  -e environment=production \
  -e rollback_reason="Critical performance issue" \
  -e emergency_contact="ops@company.com"
```

Service Isolation

```
# Isolate problematic service
ansible-playbook playbooks/isolate_service.yml \
  -e service_name=problematic-app \
  -e isolation_method=traffic_redirect
```

Emergency Scaling

```
# Emergency resource scaling
ansible-playbook playbooks/emergency_scale.yml \
  -e scale_factor=2 \
  -e target_services=web,api \
  -e duration=3600
```

Best Practices

Security Best Practices

Access Management

- Use individual user accounts (no shared accounts)
- Implement least privilege access principles
- Regularly review and audit access permissions
- Use SSH keys instead of passwords
- Enable multi-factor authentication where possible

Secrets Management

```
# Use Ansible Vault for sensitive data
ansible-vault create secrets.yml

# Encrypt existing files
ansible-vault encrypt inventory/production/group_vars/all/vault.yml

# Edit encrypted files
ansible-vault edit secrets.yml
```

Network Security

- Use VPN for remote access
- Implement network segmentation
- Regular security scanning and updates
- Monitor for suspicious activities

Operational Best Practices

Change Management

- Always use version control for changes
- Test changes in development/staging first
- Document all changes and their rationale
- Implement peer review for critical changes
- Maintain rollback procedures for all changes

Monitoring and Alerting

- Set up comprehensive monitoring for all services
- Configure meaningful alerts (avoid alert fatigue)
- Regularly review and tune alert thresholds
- Maintain up-to-date runbooks for common alerts

Documentation

- Keep documentation current and accurate
- Document all procedures and configurations
- Maintain troubleshooting guides
- Share knowledge through team training sessions

Performance Best Practices

Ansible Optimization

```
# Optimize playbook performance
- hosts: all
  strategy: free  # Parallel execution
  gather_facts: no  # Skip if not needed
  serial: 10  # Batch processing
```

Resource Management

- Monitor resource utilization regularly
- Implement auto-scaling where appropriate
- Optimize application configurations
- Regular capacity planning reviews

Certification Path

Level 1: Basic Operations (Week 1-2)

Prerequisites: Complete prerequisites and initial setup

Learning Objectives:
- Navigate the platform effectively
- Perform basic health checks
- Execute standard deployments
- Access and interpret monitoring dashboards

Certification Requirements:
- [ ] Complete hands-on lab exercises
- [ ] Pass written assessment (80% minimum)
- [ ] Demonstrate basic troubleshooting skills
- [ ] Shadow experienced team member for 1 week

Level 2: Advanced Operations (Week 3-4)

Prerequisites: Level 1 certification

Learning Objectives:
- Perform blue-green deployments
- Execute disaster recovery procedures

- Troubleshoot complex issues
- Optimize system performance

Certification Requirements:
- [ ] Complete advanced lab scenarios
- [ ] Pass comprehensive assessment (85% minimum)
- [ ] Successfully handle simulated incidents
- [ ] Lead deployment in staging environment

Level 3: Expert Operations (Month 2-3)

Prerequisites: Level 2 certification + 1 month experience

Learning Objectives:
- Design and implement new automation
- Mentor junior team members
- Lead incident response efforts
- Contribute to platform improvements

Certification Requirements:
- [ ] Complete expert-level projects
- [ ] Pass expert assessment (90% minimum)
- [ ] Lead production deployment
- [ ] Contribute to documentation/training materials

Ongoing Education

- Monthly team training sessions
- Quarterly technology updates
- Annual certification renewal
- Conference attendance and knowledge sharing

Resources and References

Documentation Links

- Ansible Official Documentation (https://docs.ansible.com/)
- Grafana Documentation (https://grafana.com/docs/)
- Prometheus Documentation (https://prometheus.io/docs/)
- Company Infrastructure Wiki (https://wiki.company.com/infrastructure)

Training Materials

- Video tutorials: `/opt/training/videos/`
- Lab environments: `https://lab.company.com`
- Practice scenarios: `/opt/training/scenarios/`

Support Channels

- **Slack**: #infrastructure-automation
- **Email**: infrastructure-team@company.com
- **Ticketing**: https://helpdesk.company.com
- **Emergency**: +1-555-9911

Document Version: 1.0
Last Updated: {{ ansible_date_time.iso8601 }}
Next Review: Monthly
Maintained By: Infrastructure Automation Team