Security Guide for VMware MCP Server

This document outlines security best practices and hardening procedures for the VMware MCP Server in production environments.

Authentication and Authorization

VMware Authentication

1. **Use Service Accounts**
   `bash`
   `# Create dedicated service account in vCenter`
   `# Grant minimal required permissions`
   `# Use strong, unique passwords`

2. **Certificate-Based Authentication**
   `bash`
   `# Enable certificate authentication`
   `export VMWARE_VERIFY_SSL=true`
   `# Use proper CA-signed certificates`

3. **Role-Based Access Control (RBAC)**
   ```bash
   # Enable RBAC in configuration
   export ENABLE_RBAC=true

# Define custom roles with minimal privileges
# Map users to appropriate roles
```

API Security

1. **JWT Token Security**
   ```bash
   # Use strong secret key (256-bit minimum)
   export SECRET_KEY=$(openssl rand -base64 32)

# Set appropriate token expiration
export JWT_EXPIRE_MINUTES=30
```

1. **HTTPS/TLS Configuration**
   `bash`
   `# Use TLS 1.2+ only`
   `# Implement proper certificate management`
   `# Enable HSTS headers`

Network Security

Firewall Configuration

```
# Allow only necessary ports
# VMware MCP Server
iptables -A INPUT -p tcp --dport 8080 -j ACCEPT

# VMware vCenter (HTTPS)
iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT

# Ollama (if enabled)
iptables -A OUTPUT -p tcp --dport 11434 -j ACCEPT

# n8n (if enabled)
iptables -A OUTPUT -p tcp --dport 5678 -j ACCEPT

# Block all other traffic
iptables -P INPUT DROP
iptables -P FORWARD DROP
```

Network Segmentation

1. **DMZ Deployment**
   - Deploy MCP server in DMZ
   - Restrict access to management networks only
   - Use jump hosts for administrative access

2. **VPN Access**
   - Require VPN for all administrative access
   - Implement multi-factor authentication
   - Use certificate-based VPN authentication

Data Protection

Encryption

1. **Data at Rest**
   ```bash
   # Encrypt configuration files
   gpg –cipher-algo AES256 –compress-algo 1 –symmetric config/.env

# Use encrypted storage for logs
# Implement database encryption if using external DB
```

1. **Data in Transit**
   ```bash
   # All communications must use TLS
   # Implement certificate pinning
   # Use mutual TLS where possible
   ```

Secrets Management

1. **Environment Variables**
   ```bash
   ```

```
   # Never hardcode secrets in code
   # Use secure secret management systems
   # Rotate secrets regularly
```

2. **HashiCorp Vault Integration**
   ```bash
   # Example Vault integration
   export VAULT_ADDR="https://vault.example.com"
   export VAULT_TOKEN="$(vault auth -method=ldap username=admin)"

# Retrieve secrets from Vault
VMWARE_PASSWORD=$(vault kv get -field=password secret/vmware/vcenter)
```

Logging and Monitoring

Audit Logging

1. **Enable Comprehensive Logging**
   ```
   bash
   export ENABLE_AUDIT_LOG=true
   export LOG_LEVEL=INFO
   export LOG_FORMAT=json
   ```

2. **Log Forwarding**
   ```
   bash
   # Forward logs to SIEM
   # Configure log retention policies
   # Implement log integrity checking
   ```

Security Monitoring

1. **Failed Authentication Monitoring**
   ```
   bash
   # Monitor for brute force attacks
   # Implement account lockout policies
   # Alert on suspicious activities
   ```

2. **Performance Monitoring**
   ```
   bash
   # Monitor for DoS attacks
   # Implement rate limiting
   # Set up resource usage alerts
   ```

Container Security

Docker Hardening

1. **Base Image Security**
   ```dockerfile
   # Use minimal base images
   FROM python:3.11-slim
   ```

Run as non-root user
RUN useradd –create-home –shell /bin/bash vmware
USER vmware

Remove unnecessary packages
RUN apt-get remove –purge -y wget curl
```

1. **Runtime Security**
   ```bash
   # Run with security options
   docker run --security-opt=no-new-privileges \
              --cap-drop=ALL \
              --read-only \
              --tmpfs /tmp \
              vmware-mcp-server
   ```

Kubernetes Security

1. **Pod Security Standards**
   ```yaml
   apiVersion: v1
   kind: Pod
   metadata:
   name: vmware-mcp-server
   spec:
   securityContext:
   runAsNonRoot: true
   runAsUser: 1000
   fsGroup: 1000
   containers:

   - name: vmware-mcp-server
     securityContext:
     allowPrivilegeEscalation: false
     readOnlyRootFilesystem: true
     capabilities:
     drop:
     - ALL
     ```

2. **Network Policies**
   ```yaml
   apiVersion: networking.k8s.io/v1
   kind: NetworkPolicy
   metadata:
   name: vmware-mcp-server-netpol
   spec:
   podSelector:
   matchLabels:

app: vmware-mcp-server
policyTypes:

- Ingress
- Egress
  ingress:
- from:
- namespaceSelector:
  matchLabels:
  name: management
  ports:
- protocol: TCP
  port: 8080
  ```

Vulnerability Management

Regular Updates

1. **Dependency Management**
   ```bash
   # Regular dependency updates
   pip-audit –requirement requirements.txt

# Automated security scanning
safety check –requirement requirements.txt
```

1. **Base Image Updates**
   ```bash
   # Regular base image updates
   docker pull python:3.11-slim

# Vulnerability scanning
trivy image vmware-mcp-server:latest
```

Security Testing

1. **Static Code Analysis**
   ```bash
   # Security linting
   bandit -r src/

# Code quality analysis
sonarqube-scanner
```

1. **Penetration Testing**
   `bash`
   `# Regular penetration testing`
   `# API security testing`
   `# Infrastructure security assessment`
```

Incident Response

Preparation

1. **Incident Response Plan**
   - Define roles and responsibilities
   - Establish communication channels
   - Create runbooks for common scenarios

2. **Backup and Recovery**
   ```bash
   # Regular configuration backups
   tar -czf vmware-mcp-backup-$(date +%Y%m%d).tar.gz config/ data/

# Test recovery procedures
# Document recovery time objectives (RTO)
```

Detection and Response

1. **Automated Alerting**
   ```bash
   # Configure alerts for:
   # - Failed authentication attempts
   # - Unusual API usage patterns
   # - System resource exhaustion
   # - Certificate expiration
   ```

2. **Forensic Capabilities**
   ```bash
   # Enable detailed logging
   # Preserve log integrity
   # Implement log correlation
   ```

Compliance

Regulatory Requirements

1. **SOX Compliance**
   - Implement change management processes
   - Maintain audit trails
   - Segregate duties

2. **GDPR Compliance**
   - Implement data minimization
   - Provide data portability
   - Enable right to erasure

Industry Standards

1. **CIS Controls**
   - Implement CIS benchmarks
   - Regular compliance assessment
   - Continuous monitoring

2. **NIST Framework**
   - Follow NIST cybersecurity framework
   - Implement risk management processes
   - Regular security assessments

Security Checklist

Pre-Production

- [ ] Change all default passwords
- [ ] Enable TLS/SSL encryption
- [ ] Configure proper firewall rules
- [ ] Implement RBAC
- [ ] Enable audit logging
- [ ] Configure monitoring and alerting
- [ ] Perform security testing
- [ ] Document security procedures

Production

- [ ] Regular security updates
- [ ] Monitor security logs
- [ ] Perform regular backups
- [ ] Test incident response procedures
- [ ] Review access permissions
- [ ] Update security documentation
- [ ] Conduct security training
- [ ] Perform compliance audits

Post-Incident

- [ ] Conduct post-incident review
- [ ] Update security procedures
- [ ] Implement lessons learned
- [ ] Update incident response plan
- [ ] Communicate with stakeholders
- [ ] Document improvements
- [ ] Schedule follow-up assessments

Contact Information

For security issues or questions:
- Security Team: security@example.com
- Emergency Contact: +1-555-SECURITY
- Incident Response: incident-response@example.com

References

- VMware Security Hardening Guides (https://docs.vmware.com/en/VMware-vSphere/index.html)
- OWASP API Security Top 10 (https://owasp.org/www-project-api-security/)

- NIST Cybersecurity Framework (https://www.nist.gov/cyberframework)
- CIS Controls (https://www.cisecurity.org/controls/)