Security Guide

This document outlines security best practices and hardening procedures for the Proxmox MCP Server.

🔒 Security Overview

The Proxmox MCP Server implements multiple layers of security to protect your infrastructure:

- **Authentication**: JWT-based authentication with Proxmox VE integration
- **Authorization**: Role-based access control (RBAC) with fine-grained permissions
- **Encryption**: TLS/SSL for all communications
- **Audit Logging**: Comprehensive security event logging
- **Network Security**: Rate limiting, firewall integration, and network segmentation

🛡️ Authentication & Authorization

JWT Configuration

```
# Generate a strong secret key
openssl rand -hex 32

# Set in environment
export SECRET_KEY="your-generated-secret-key"
export JWT_EXPIRE_MINUTES=60
export JWT_ALGORITHM=HS256
```

RBAC Roles

The system supports three built-in roles with different permission levels:

Admin Role

- Full access to all operations
- Can manage users and permissions
- Access to system configuration

```
admin_permissions:
  - vm:create, vm:delete, vm:start, vm:stop, vm:clone, vm:migrate
  - lxc:create, lxc:delete, lxc:start, lxc:stop, lxc:clone
  - cluster:configure, cluster:monitor, cluster:ha
  - storage:create, storage:delete, storage:configure
  - network:create, network:delete, network:configure
  - backup:create, backup:restore, backup:delete, backup:schedule
  - system:configure, system:monitor, system:logs
```

Operator Role

- Limited operational access
- Cannot modify system configuration
- Can perform routine operations

```
operator_permissions:
  - vm:start, vm:stop, vm:clone, vm:monitor
  - lxc:start, lxc:stop, lxc:clone, lxc:monitor
  - backup:create, backup:monitor
  - cluster:monitor, storage:monitor, network:monitor
```

Viewer Role

- Read-only access
- Monitoring and reporting only

```
viewer_permissions:
  - vm:monitor, lxc:monitor, cluster:monitor
  - storage:monitor, network:monitor, backup:monitor
  - system:monitor
```

Custom Roles

Create custom roles by extending the base permission system:

```
# Custom role definition
CUSTOM_ROLES = {
    "backup_admin": [
        "backup:create", "backup:restore", "backup:delete",
        "backup:schedule", "backup:monitor",
        "storage:monitor", "vm:monitor", "lxc:monitor"
    ],
    "network_admin": [
        "network:create", "network:delete", "network:configure",
        "network:monitor", "cluster:monitor"
    ]
}
```

🔐 SSL/TLS Configuration

Certificate Generation

Self-Signed Certificates (Development)

```
# Create SSL directory
mkdir -p config/nginx/ssl

# Generate private key
openssl genrsa -out config/nginx/ssl/proxmox-mcp.key 2048

# Generate certificate
openssl req -new -x509 -key config/nginx/ssl/proxmox-mcp.key \
    -out config/nginx/ssl/proxmox-mcp.crt -days 365 \
    -subj "/C=US/ST=State/L=City/O=Organization/CN=proxmox-mcp.local"
```

Let’s Encrypt (Production)

```
# Install certbot
sudo apt-get install certbot

# Generate certificate
sudo certbot certonly --standalone \
    -d proxmox-mcp.yourdomain.com \
    -d n8n.yourdomain.com \
    -d grafana.yourdomain.com

# Copy certificates
sudo cp /etc/letsencrypt/live/proxmox-mcp.yourdomain.com/fullchain.pem \
    config/nginx/ssl/proxmox-mcp.crt
sudo cp /etc/letsencrypt/live/proxmox-mcp.yourdomain.com/privkey.pem \
    config/nginx/ssl/proxmox-mcp.key

# Set permissions
sudo chown $USER:$USER config/nginx/ssl/*
chmod 600 config/nginx/ssl/proxmox-mcp.key
chmod 644 config/nginx/ssl/proxmox-mcp.crt
```

TLS Configuration

Update `config/nginx/nginx.conf` for production:

```
# Strong SSL configuration
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-AES256-GCM-SHA384;
ssl_prefer_server_ciphers off;
ssl_session_cache shared:SSL:10m;
ssl_session_timeout 10m;

# HSTS
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;

# OCSP Stapling
ssl_stapling on;
ssl_stapling_verify on;
ssl_trusted_certificate /etc/nginx/ssl/chain.pem;
```

🔥 Firewall Configuration

UFW (Ubuntu Firewall)

```
# Reset firewall
sudo ufw --force reset

# Default policies
sudo ufw default deny incoming
sudo ufw default allow outgoing

# SSH access (adjust port as needed)
sudo ufw allow 22/tcp

# HTTP/HTTPS
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp

# Proxmox VE (if on same network)
sudo ufw allow from 192.168.1.0/24 to any port 8006

# Internal services (Docker network)
sudo ufw allow from 172.20.0.0/16

# Enable firewall
sudo ufw enable
```

iptables Rules

```
# Create custom chain for Proxmox MCP
iptables -N PROXMOX_MCP

# Rate limiting for API endpoints
iptables -A PROXMOX_MCP -p tcp --dport 8080 -m limit --limit 10/min -j ACCEPT
iptables -A PROXMOX_MCP -p tcp --dport 8080 -j DROP

# Allow established connections
iptables -A PROXMOX_MCP -m state --state ESTABLISHED,RELATED -j ACCEPT

# Apply chain
iptables -A INPUT -j PROXMOX_MCP
```

🛡️ Network Security

Rate Limiting

Configure rate limiting in nginx:

```
# Rate limiting zones
limit_req_zone $binary_remote_addr zone=api:10m rate=10r/s;
limit_req_zone $binary_remote_addr zone=webhook:10m rate=100r/s;
limit_req_zone $binary_remote_addr zone=auth:10m rate=1r/s;

# Apply limits
location /api/ {
    limit_req zone=api burst=20 nodelay;
    # ... other config
}

location /webhook/ {
    limit_req zone=webhook burst=50 nodelay;
    # ... other config
}

location /auth/ {
    limit_req zone=auth burst=5 nodelay;
    # ... other config
}
```

IP Whitelisting

Restrict access to sensitive endpoints:

```
# Metrics endpoint - internal only
location /metrics {
    allow 172.20.0.0/16;  # Docker network
    allow 10.0.0.0/8;     # Internal network
    deny all;

    proxy_pass http://proxmox_mcp/metrics;
}

# Admin endpoints
location /api/v1/admin/ {
    allow 192.168.1.100;  # Admin workstation
    allow 10.0.1.0/24;    # Admin network
    deny all;

    proxy_pass http://proxmox_mcp;
}
```

Network Segmentation

Use Docker networks for isolation:

```yaml
# docker-compose.yml
networks:
  frontend:
    driver: bridge
    ipam:
      config:
        - subnet: 172.20.1.0/24
  backend:
    driver: bridge
    internal: true
    ipam:
      config:
        - subnet: 172.20.2.0/24

services:
  nginx:
    networks:
      - frontend

  proxmox-mcp-server:
    networks:
      - frontend
      - backend

  postgres:
    networks:
      - backend
```

🔍 Audit Logging

Security Event Logging

Configure comprehensive security logging:

```python
# Security events to log
SECURITY_EVENTS = [
    'authentication_success',
    'authentication_failure',
    'authorization_denied',
    'privilege_escalation_attempt',
    'suspicious_activity',
    'configuration_change',
    'data_access',
    'system_modification'
]

# Log format
SECURITY_LOG_FORMAT = {
    'timestamp': '%(asctime)s',
    'level': '%(levelname)s',
    'event_type': '%(event_type)s',
    'user': '%(user)s',
    'source_ip': '%(source_ip)s',
    'resource': '%(resource)s',
    'action': '%(action)s',
    'result': '%(result)s',
    'details': '%(details)s'
}
```

Log Monitoring

Set up log monitoring with fail2ban:

```
# /etc/fail2ban/jail.local
[proxmox-mcp-auth]
enabled = true
port = 80,443,8080
filter = proxmox-mcp-auth
logpath = /var/log/proxmox-mcp-server.log
maxretry = 5
bantime = 3600
findtime = 600

# /etc/fail2ban/filter.d/proxmox-mcp-auth.conf
[Definition]
failregex = .*authentication_failure.*source_ip.*<HOST>.*
ignoreregex =
```

🔒 Data Protection

Encryption at Rest

Encrypt sensitive data in the database:

```
from cryptography.fernet import Fernet

class EncryptedField:
    def __init__(self, key):
        self.cipher = Fernet(key)

    def encrypt(self, data):
        return self.cipher.encrypt(data.encode()).decode()

    def decrypt(self, encrypted_data):
        return self.cipher.decrypt(encrypted_data.encode()).decode()

# Usage
encryption_key = os.environ.get('ENCRYPTION_KEY')
encrypted_field = EncryptedField(encryption_key)

# Encrypt Proxmox credentials
encrypted_password = encrypted_field.encrypt(proxmox_password)
```

Secrets Management

Use Docker secrets for sensitive data:

```yaml
# docker-compose.yml
secrets:
  proxmox_password:
    file: ./secrets/proxmox_password.txt
  jwt_secret:
    file: ./secrets/jwt_secret.txt
  db_password:
    file: ./secrets/db_password.txt

services:
  proxmox-mcp-server:
    secrets:
      - proxmox_password
      - jwt_secret
    environment:
      - PROXMOX_PASSWORD_FILE=/run/secrets/proxmox_password
      - SECRET_KEY_FILE=/run/secrets/jwt_secret
```

🚨 Security Monitoring

Intrusion Detection

Set up OSSEC for host-based intrusion detection:

```xml
<!-- /var/ossec/etc/ossec.conf -->
<ossec_config>
  <localfile>
    <log_format>json</log_format>
    <location>/var/log/proxmox-mcp-server.log</location>
  </localfile>

  <rules>
    <include>proxmox_mcp_rules.xml</include>
  </rules>
</ossec_config>
```

Security Metrics

Monitor security metrics with Prometheus:

```python
from prometheus_client import Counter, Histogram

# Security metrics
auth_attempts = Counter('auth_attempts_total', 'Authentication attempts', ['result'])
auth_failures = Counter('auth_failures_total', 'Authentication failures', ['reason'])
permission_denials = Counter('permission_denials_total', 'Permission denials', ['re-
source'])
suspicious_activity = Counter('suspicious_activity_total', 'Suspicious activity', ['typ
e'])

# Usage
auth_attempts.labels(result='success').inc()
auth_failures.labels(reason='invalid_credentials').inc()
```

Alerting Rules

Configure Prometheus alerting:

```yaml
# security_alerts.yml
groups:
  - name: security.rules
    rules:
      - alert: HighAuthFailureRate
        expr: rate(auth_failures_total[5m]) > 0.1
        for: 2m
        labels:
          severity: warning
        annotations:
          summary: "High authentication failure rate"
          description: "Authentication failure rate is {{ $value }} per second"

      - alert: SuspiciousActivity
        expr: increase(suspicious_activity_total[1h]) > 10
        for: 0m
        labels:
          severity: critical
        annotations:
          summary: "Suspicious activity detected"
          description: "{{ $value }} suspicious activities in the last hour"
```

🔧 Security Hardening

Container Security

Harden Docker containers:

```dockerfile
# Use non-root user
FROM python:3.11-slim
RUN useradd -m -u 1000 proxmox && \
    chown -R proxmox:proxmox /app
USER proxmox

# Remove unnecessary packages
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
    curl gcc && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

# Set security options
LABEL security.no-new-privileges=true
```

```yaml
# docker-compose.yml security options
services:
  proxmox-mcp-server:
    security_opt:
      - no-new-privileges:true
    cap_drop:
      - ALL
    cap_add:
      - NET_BIND_SERVICE
    read_only: true
    tmpfs:
      - /tmp
      - /var/tmp
```

System Hardening

Apply system-level hardening:

```
# Disable unnecessary services
sudo systemctl disable bluetooth
sudo systemctl disable cups
sudo systemctl disable avahi-daemon

# Kernel hardening
echo 'kernel.dmesg_restrict = 1' | sudo tee -a /etc/sysctl.conf
echo 'kernel.kptr_restrict = 2' | sudo tee -a /etc/sysctl.conf
echo 'net.ipv4.conf.all.send_redirects = 0' | sudo tee -a /etc/sysctl.conf
echo 'net.ipv4.conf.all.accept_redirects = 0' | sudo tee -a /etc/sysctl.conf

# Apply changes
sudo sysctl -p
```

📋 Security Checklist

Pre-Deployment

- [ ] Generate strong secret keys
- [ ] Configure SSL/TLS certificates
- [ ] Set up firewall rules
- [ ] Configure rate limiting
- [ ] Enable audit logging
- [ ] Set up monitoring and alerting
- [ ] Review and customize RBAC roles
- [ ] Encrypt sensitive configuration data

Post-Deployment

- [ ] Verify SSL/TLS configuration
- [ ] Test authentication and authorization
- [ ] Validate firewall rules
- [ ] Check log collection and monitoring
- [ ] Perform security scanning
- [ ] Review access logs
- [ ] Test backup and recovery procedures
- [ ] Document security procedures

Ongoing Maintenance

- [ ] Regular security updates
- [ ] Certificate renewal
- [ ] Log review and analysis
- [ ] Security metric monitoring
- [ ] Penetration testing
- [ ] Access review and cleanup
- [ ] Backup verification
- [ ] Incident response testing

🚨 Incident Response

Security Incident Procedures

1. **Detection and Analysis**
   - Monitor security alerts
   - Analyze suspicious activities
   - Determine incident scope

2. **Containment**
   - Isolate affected systems
   - Preserve evidence
   - Prevent further damage

3. **Eradication and Recovery**
   - Remove threats
   - Restore systems
   - Implement additional controls

4. **Post-Incident Activities**
   - Document lessons learned
   - Update security procedures
   - Improve monitoring

Emergency Contacts

```yaml
# security_contacts.yml
incident_response_team:
  primary: security@yourdomain.com
  secondary: admin@yourdomain.com

escalation:
  level_1: team-lead@yourdomain.com
  level_2: cto@yourdomain.com

external:
  cert: cert@yourdomain.com
  legal: legal@yourdomain.com
```

📚 Additional Resources

- OWASP Top 10 (https://owasp.org/www-project-top-ten/)
- NIST Cybersecurity Framework (https://www.nist.gov/cyberframework)
- CIS Controls (https://www.cisecurity.org/controls/)
- Docker Security Best Practices (https://docs.docker.com/engine/security/)
- Proxmox VE Security (https://pve.proxmox.com/wiki/Security)

Remember: Security is an ongoing process, not a one-time setup. Regularly review and update your security measures to protect against evolving threats.