Troubleshooting Guide - Multi-Cloud Manager

Common issues and their solutions.

Table of Contents

Startup Issues

Container Won't Start

Symptoms:

```bash
docker ps -a
# Shows container as "Exited"
```

Solutions:

1. **Check logs:**

```bash
docker logs do-multi
```

2. **Common causes:**
   - Missing environment variables
   - Port 8010 already in use
   - Invalid configuration

3. **Fix port conflict:**

```bash
# Find what's using port 8010
sudo lsof -i :8010

# Kill the process or change PORT in .env
PORT=8011
```

4. **Verify environment file:**

```bash
# Check .env exists and has required variables
cat .env | grep -E "API_TOKEN|DROPLET_ID|REGISTRY"
```

Health Endpoint Returns 503

Symptoms:

```bash
curl http://localhost:8010/health
# Connection refused or timeout
```

Solutions:

1. **Check container is running:**

```bash
docker ps | grep do-multi
```

2. **Check logs for errors:**

```bash
docker logs -f do-multi
```

3. **Restart container:**

```bash
```

```
docker restart do-multi
```

4. **Rebuild if needed:**

```bash
docker-compose down
docker-compose build --no-cache
docker-compose up -d
```

Authentication Problems

401 Unauthorized

Symptoms:

```json
{
  "detail": "Missing Authorization header"
}
```

Solutions:

1. **Check token format:**

```bash
# Correct format
Authorization: Bearer secretkey_7f8b4e2a9d034b5cb7219d6f81e3d2c1

# Wrong format (missing "Bearer ")
Authorization: secretkey_7f8b4e2a9d034b5cb7219d6f81e3d2c1
```

2. **Verify token in .env:**

```bash
cat .env | grep API_TOKEN
```

3. **Test with curl:**

```bash
curl -H "Authorization: Bearer YOUR_TOKEN_HERE" \
  http://localhost:8010/logs
```

JWT Verification Failed

Symptoms:

```json
{
  "detail": "Invalid JWT token: Signature verification failed"
}
```

Solutions:

1. **Check JWT expiry:**
   - Tokens expire after 1 hour
   - Fetch a new token from Registry
2. **Verify JWKS is loaded:**

```bash
docker logs do-multi | grep JWKS
# Should see: "✅ JWKS loaded with X public keys"
```

3. **Check fallback is working:**

```bash
docker logs do-multi | grep fallback
# Should see: "💡 Will use API_TOKEN fallback"
```

4. **Use API_TOKEN instead:**

```bash
# If JWT isn't working, use simple token
curl -H "Authorization: Bearer secretkey_7f8b4e2a9d034b5cb7219d6f81e3d2c1" \
  http://localhost:8010/logs
```

Cloud Provider Errors

503 Service Unavailable (Provider Not Configured)

Symptoms:

```json
{
  "detail": "DigitalOcean not configured"
}
```

Solutions:

1. **Check token is set:**

```bash
cat .env | grep DO_TOKEN
# Should not be empty
```

2. **Verify token format:**
   - DigitalOcean: `dop_v1_...`
   - Hetzner: 64 character string
   - Vultr: 32 character string

3. **Restart after adding token:**

```bash
docker restart do-multi
```

Provider API Rate Limit (429 Too Many Requests)

Symptoms:

```json
{
  "detail": "Rate limit exceeded"
}
```

Rate Limits:

- **DigitalOcean:** 5000 requests/hour
- **Hetzner:** Varies by endpoint
- **Vultr:** 30 requests/second

Solutions:

1. **Wait and retry:**
    - Implement exponential backoff
    - Wait 1-5 minutes before retrying
2. **Cache responses:**
    - Don't poll /list endpoints too frequently
    - Cache results for 30-60 seconds
3. **Use multi/list sparingly:**
    - Calls all providers at once
    - Counts toward each provider's limit

Invalid Provider Credentials

Symptoms:

```json
{
  "detail": "Unable to authenticate you"
}
```

Solutions:

1. **Verify token validity:**
    - Test token directly with provider API
    - Regenerate if expired
2. **Check token permissions:**
    - Needs read/write access
    - Check provider dashboard for token status

3. **Test with curl:**

DigitalOcean:

```bash
curl -H "Authorization: Bearer YOUR_DO_TOKEN" \
  https://api.digitalocean.com/v2/account
```

Hetzner:

```bash
curl -H "Authorization: Bearer YOUR_HETZNER_TOKEN" \
  https://api.hetzner.cloud/v1/servers
```

Vultr:

```bash
curl -H "Authorization: Bearer YOUR_VULTR_TOKEN" \
  https://api.vultr.com/v2/account
```

Registry Integration

Cannot Fetch JWT Token

Symptoms:

```bash
❌ Failed to fetch JWT: HTTP 401
```

Solutions:

1. **Verify Registry key:**

```bash
cat .env | grep REGISTRY_KEY
# Should match: regkey_2f14c9b6e9b047d2b8c5a7cf93b2e4da
```

2. **Check Registry is reachable:**

```bash
curl https://drop18.fullpotential.ai/health
```

3. **Test token fetch manually:**

```bash
curl -X POST \
  "https://drop18.fullpotential.ai/auth/token?droplet_id=drop4.fullpotential.ai" \
  -H "X-Registry-Key: regkey_2f14c9b6e9b047d2b8c5a7cf93b2e4da"
```

Registration/Heartbeat Failures

Symptoms:

```bash
❌ Registry internal error
⚠️  Heartbeat failed: 500
```

Solutions:

1. **Check payload format:**
   - Must include `id`, `host`, `droplet_id` at top level
   - Review logs for exact error
2. **Verify network connectivity:**

```bash
docker exec do-multi curl -v https://drop18.fullpotential.ai/health
```

3. **Check Registry status:**
   - May be temporary Registry issue
   - Droplet will auto-retry
4. **View heartbeat logs:**

```bash
```

```
docker logs do-multi | grep heartbeat
```

JWKS Not Available (404)

Symptoms:

```bash
⚠️ JWKS endpoint not available
💡 Will use API_TOKEN fallback
```

Impact:

- Incoming JWT verification disabled
- Falls back to API_TOKEN authentication
- Not a critical error

Solutions:

1. **This is expected behavior:**
   - JWKS may not be deployed yet
   - Fallback handles this gracefully
2. **Verify fallback works:**

```bash
curl -H "Authorization: Bearer secretkey_7f8b4e2a9d034b5cb7219d6f81e3d2c1" \
  http://localhost:8010/logs
```

3. **No action needed unless:**
   - You specifically need RS256 JWT verification
   - Other droplets require JWKS-verified tokens

Performance Issues

High CPU Usage

Symptoms:

```bash
docker stats do-multi
# Shows >80% CPU
```

Solutions:

1. **Check for loops:**

```bash
docker logs do-multi | grep ERROR
```

2. **Reduce heartbeat frequency:**

```bash
# In .env, increase interval
HEARTBEAT_INTERVAL=60  # Default is 30
```

3. **Check for excessive logging:**

```bash
docker logs do-multi | wc -l
# If very high, may need log rotation
```

High Memory Usage

Symptoms:

```bash
docker stats do-multi
# Shows >1GB memory
```

Solutions:

1. **Check log/event buffers:**
   - Logs buffer: 1000 entries
   - Events buffer: 100 entries

   - Actions buffer: 50 entries

2. **Restart to clear buffers:**

```bash
docker restart do-multi
```

3. **Set memory limit:**

```yaml
# In docker-compose.yml
services:
  multi-cloud-manager:
    mem_limit: 512m
```

Slow Response Times

Symptoms:

- API calls take >2 seconds
- Timeout errors

Solutions:

1. **Check provider API latency:**
   - Provider APIs may be slow
   - Use `/multi/list` carefully (calls all providers)

2. **Network issues:**

```bash
# Test connectivity
docker exec do-multi curl -w "@curl-format.txt" \
  https://api.digitalocean.com/v2/account
```

3. **Increase timeouts:**
   - HTTP client timeout is 30 seconds
   - Provider APIs may need more time

Docker Issues

Port Already in Use

Symptoms:

```bash
Error: bind: address already in use
```

Solutions:

1. **Find process using port:**

```bash
sudo lsof -i :8010
```

2. **Kill process:**

```bash
sudo kill -9 <PID>
```

3. **Or change port:**

```bash
# In .env
PORT=8011

# Restart
docker-compose down
docker-compose up -d
```

Container Keeps Restarting

Symptoms:

```bash
```

```
docker ps -a
# STATUS: Restarting
```

Solutions:

1. **Check logs for crash:**

```bash
docker logs do-multi --tail 100
```

2. **Common causes:**
   - Python import errors
   - Missing dependencies
   - Configuration errors
3. **Rebuild image:**

```bash
docker-compose down
docker-compose build --no-cache
docker-compose up -d
```

Volume/Permission Issues

Symptoms:

```bash
Permission denied: '.env'
```

Solutions:

1. **Check file permissions:**

```bash
ls -la .env
chmod 644 .env
```

2. **Check Docker user:**

```bash
docker exec do-multi whoami
# Should match file owner
```

Debugging Commands

Essential Debug Commands

```bash
# View live logs
docker logs -f do-multi

# View last 100 log lines
docker logs do-multi --tail 100

# Check container status
docker ps -a | grep do-multi

# Check resource usage
docker stats do-multi

# Inspect container
docker inspect do-multi

# Access container shell
docker exec -it do-multi /bin/bash

# Test health endpoint
curl http://localhost:8010/health

# View all environment variables
docker exec do-multi env

# Check Python version
docker exec do-multi python --version

# List installed packages
docker exec do-multi pip list
```

Log Analysis

```bash
# Filter by error level
docker logs do-multi | grep ERROR

# Filter by specific component
docker logs do-multi | grep JWT
docker logs do-multi | grep heartbeat
docker logs do-multi | grep JWKS

# Count errors
docker logs do-multi | grep ERROR | wc -l

# Get startup logs
docker logs do-multi | head -50

# Get recent logs
docker logs do-multi | tail -50
```

Network Debugging

```bash
# Test Registry connectivity
docker exec do-multi curl -v https://drop18.fullpotential.ai/health

# Test provider API
docker exec do-multi curl -H "Authorization: Bearer $DO_TOKEN" \
  https://api.digitalocean.com/v2/account

# Check DNS resolution
docker exec do-multi nslookup drop18.fullpotential.ai

# Check network interfaces
docker exec do-multi ip addr
```

Getting Help

If you've tried all solutions and still have issues:

1. **Collect diagnostic info:**

```bash
# Save logs
docker logs do-multi > droplet-logs.txt

# Save configuration (remove sensitive data!)
cat .env > droplet-config.txt

# Save container info
docker inspect do-multi > container-info.txt
```

2. **Check documentation:**
   - README.md
   - INTEGRATION_GUIDE.md
   - API_REFERENCE.md
3. **Contact steward:**
   - GitHub Issues
   - @Hassan (Steward)
4. **Include in report:**
   - Error message
   - Steps to reproduce
   - Logs (sanitized)
   - Environment (OS, Docker version)
   - What you've tried already