TROUBLESHOOTING REPORT

VIRTUAL CONTROL

VMWARE CLOUD FOUNDATION SOLUTIONS

NSX Manager 9.0.1 Cold Start Service Failure

Root cause analysis documenting NSX Manager service chain failure after cold start, manual service recovery, and VCF Operations re-integration.

NSX MANAGER | COLD START | SERVICE RECOVERY | ROOT CAUSE ANALYSIS

VCF 9.0 VMware Cloud Foundation

PROPRIETARY & CONFIDENTIAL

NSX Manager 9.0.1 Cold Start Service Failure — Root Cause Analysis & Recovery

Prepared by: Virtual Control LLC **Date:** March 26, 2026 **Document Version:** 1.0 **Classification:** Internal — Lab Environment **NSX Version:** 9.0.1.0 (Build 24952114)

Table of Contents

Related RCAs in this library *(`06-Troubleshooting/`):*

- *`VCF9-Management-Plane-Loss-RCA-2026-05-01` — vSAN object inaccessibility cascade after MM host rebuild with EA mode; source of the FDM-only rule cited in Addendum D (L-32) of this document.*
- *`Fleet-Operations-Integration-Recovery-RCA` — VCF Operations ↔ Fleet Management half-broken integration registration; KB-aligned repair via Disconnect/Reconnect with DB-level cleanup.*

1. Executive Summary

Where commands run in this RCA. NSX cold-start recovery touches:

- **NSX Manager shell (SSH)** as `admin`, then `st en` for NSX-T CLI. Commands like `start service manager`, `get service`, `get cluster status`. Most diagnostic and recovery work happens here.
- **NSX Manager Linux shell** — root escalation rare; only for log file inspection under `/var/log/`.
- **vCenter UI (browser)** — VM-level operations on the NSX Manager VM (snapshot, power cycle).
- **VCF Operations UI** — adapter status, NSX integration state.

Replace any credential placeholder like `<your-password>` with values from your vault.

On March 26, 2026, a complete NSX Manager service failure was identified in the VCF 9.0.1 nested lab environment. The NSX Manager appliance (`nsx-node1.lab.local`, 192.168.1.71) was powered on and responding to ICMP, but

all core platform services — including HTTP, Manager, Controller, Search, and Authentication — were in a **stopped** state. The NSX Manager UI was unreachable on both the node IP (192.168.1.71) and the cluster VIP (192.168.1.70). VCF Operations reported two integration adapters in **Warning** state due to the inability to connect to NSX.

The root cause was determined to be a **cold start service initialization failure**. After a lab shutdown and restart, the NSX Manager appliance booted successfully at the OS level, but the NSX application services did not auto-start in the correct dependency order. The Corfu datastore service was running, but the HTTP service — which all API and UI access depends on — remained stopped, preventing the rest of the service chain from initializing.

Recovery was achieved by manually starting services in the correct dependency order: **datastore → http → manager → controller**. This cascaded the startup of all remaining 30+ services. Total recovery time from initial diagnosis to full service restoration was approximately 40 minutes.

> **Root Cause:** NSX Manager 9.0.1 cold start in a resource-constrained nested environment caused the HTTP service to fail during automatic initialization. The Corfu datastore started successfully, but the HTTP service did not start within the expected timeout window, breaking the service dependency chain and leaving all dependent services (Manager, Controller, Search, Auth, and 25+ policy services) in a stopped state.

---

2. Environment Reference

2.1 Infrastructure Details

Parameter	Value
Physical Host	Dell Precision 7920
Hypervisor	VMware Workstation 17.x (Nested)
VCF Version	9.0.1
Lab Domain	lab.local
Deployment Type	Single-node NSX Manager (nested)

2.2 NSX Manager Configuration

Parameter	Value
Appliance Hostname	nsx-node1.lab.local
Node UUID	95493642-ef4a-cb8e-ed7c-5bc20033f2c2
Node IP Address	192.168.1.71

Cluster VIP	192.168.1.70
NSX Version	9.0.1.0
Build Number	24952114
Cluster ID	3d5211c5-a4e1-4535-a803-f10726c26d59
Deployment Type	Single-node cluster

2.3 Upstream Dependencies

Component	IP Address	Role
vCenter Server 9.0	192.168.1.69	Infrastructure management
SDDC Manager 9.0	192.168.1.241	Lifecycle management
VCF Operations 9.0.2	—	Monitoring (VCF Operations Collector)
ESXi Hosts	192.168.1.74–.76, .82	Compute

---

3. Problem Statement

3.1 Symptom Description

The issue was initially identified in VCF Operations (Administration → Integrations → Accounts). Two adapters displayed **Warning** status:

1. **lab** — VMware Cloud Foundation Adapter (collector.lab.local) — **Warning**
2. **nsx-vip.lab.local** — NSX Adapter (collector.lab.local) — **Warning**

The NSX adapter reported: `Error trying to establish connection`

Additional symptoms:

- `https://192.168.1.70` (VIP) — **not reachable** in browser
- `https://192.168.1.71` (node) — **not reachable** in browser
- `ping 192.168.1.70` — **no response**
- `ping 192.168.1.71` — **responding**

3.2 Affected Components

Component	Status	Impact

NSX Manager UI	Unreachable	No management access
NSX Manager API	Unreachable	No programmatic access
Cluster VIP (192.168.1.70)	Not responding	VIP not serving traffic
VCF Operations NSX Adapter	Warning	No NSX metric collection
VCF Operations VCF Adapter	Warning	Partial VCF data collection failure
NSX-backed overlay networking	Degraded	Existing workloads operational; no changes possible

3.3 Business Impact

In a production environment, this failure would result in:

- **Complete loss of NSX management plane access** — no ability to create, modify, or delete logical networking, firewall rules, or load balancer configurations
- **VCF Operations monitoring gap** — no NSX metrics, alerts, or compliance data collected during the outage
- **SDDC Manager lifecycle operations blocked** — any pending NSX upgrades or configuration changes would fail
- **Security policy changes frozen** — distributed firewall rules cannot be modified

> ***Note:*** *Existing data plane connectivity (overlay networks, firewall rules already pushed to hosts) remains operational during an NSX Manager outage. The control plane is affected, not the data plane.*

---

4. Root Cause Analysis

4.1 RCA Summary

What happened: The NSX Manager appliance was powered off as part of a full lab shutdown. Upon restart, the appliance OS booted successfully and basic node services (SSH, NTP, node-mgmt) started. However, the core NSX platform services did not auto-start. The HTTP service failed to initialize within the expected timeout, which broke the service dependency chain.

Why it happened: In a nested virtualization environment running on VMware Workstation, the NSX Manager appliance competes for CPU and I/O resources during boot. The Corfu datastore service started successfully, but the HTTP service — which depends on datastore readiness and requires significant memory allocation — did not start before the system's service startup timeout expired. With HTTP stopped, no dependent services (Manager, Controller, Auth, Search) could start because they require the HTTP/API layer for inter-service communication and registration.

Why it was not detected immediately: The appliance was responsive to ICMP (ping) on the node IP (192.168.1.71), which could give a false impression of health. The VIP (192.168.1.70) did not respond because VIP assignment depends on the cluster manager service, which was also stopped. VCF Operations detected the failure via adapter warnings, which was the initial alert that triggered investigation.

4.2 Service Dependency Chain

The following diagram shows the NSX Manager service startup order. An arrow indicates a dependency — the service on the right requires the service on the left to be running.

```
Boot Sequence (Automatic):
  OS → ssh, ntp, node-mgmt, node-stats, syslog, nsx-upgrade-agent

Service Dependency Chain (Must be started in order):
  datastore (Corfu DB)
    → http (API/UI gateway)
      → auth (authentication)
      → manager (core management plane)
        → controller (control plane)
        → search (indexing)
        → monitoring
        → cm-inventory
        → messaging-manager
        → cluster_manager
        → site_manager
        → install-upgrade
        → async_replicator
        → idps-reporting
        → All POLICY_SVC_* services (30+)
```

Key insight: The `http` service is the single point of failure in the startup chain. If `http` does not start, no service above it in the chain can start, even if `datastore` is healthy.

4.3 Why Services Did Not Auto-Start

NSX Manager 9.0.1 uses a service orchestration framework that starts services in dependency order during boot. In resource-constrained environments (nested lab, shared CPU/RAM, slow storage), the following sequence occurs:

1. The OS boots and starts basic services (SSH, NTP, node-mgmt)
2. The Corfu datastore service starts (this is the database layer)
3. The HTTP service attempts to start, but requires:

- Corfu to be fully ready (not just running, but accepting connections)
- Sufficient free memory to allocate its Java heap
- Ports 443 and 80 to be available

4. If the HTTP service does not confirm readiness within the orchestrator's timeout window, it is marked as failed and no retry is attempted
5. All services downstream of HTTP remain in `stopped` state indefinitely

This is a **known behavior in nested environments** and is not caused by data corruption, misconfiguration, or hardware failure.

---

5. Phase 1 — Initial Assessment

5.1 VCF Operations Alerts

The issue was first observed in VCF Operations under **Administration → Integrations → Accounts**:

Adapter Name	Type	Status	Collector
lab	VMware Cloud Foundation Adapter	**Warning**	collector.lab.local
mgmt	VCF Operations Collector	Collecting	VCF Operations Collector-vcf-ops
mgmt	VCF Operations Collector	Collecting	VCF Operations Collector-vcf-ops
mgmt - vSAN	VCF Operations Collector	Collecting	VCF Operations Collector-vcf-ops
nsx-vip.lab.local	NSX Adapter	**Warning**	collector.lab.local

The NSX adapter (`nsx-vip.lab.local`) reported: **"Error trying to establish connection"**

5.2 Network Connectivity Test

From the management workstation:

```
C:\> ping 192.168.1.70
Request timed out.
Request timed out.

C:\> ping 192.168.1.71
Reply from 192.168.1.71: bytes=32 time<1ms TTL=64
Reply from 192.168.1.71: bytes=32 time<1ms TTL=64
```

Result: Node IP (.71) responds to ICMP. Cluster VIP (.70) does not respond.

Analysis: The NSX Manager OS is running, but the cluster VIP is not active. The VIP is managed by the `cluster_manager` service, which depends on the HTTP service. This confirms a service-level failure, not a network or VM-level failure.

5.3 Browser Access Test

URL	Result
`https://192.168.1.70`	Connection refused — page did not load
`https://192.168.1.71`	Connection refused — page did not load

Analysis: Neither the VIP nor the direct node IP serves the NSX UI. The HTTP service (which serves the UI on port 443) is not running.

---

6. Phase 2 — SSH Diagnostics

6.1 Establish SSH Session

SSH access was available because the `ssh` service starts independently of the NSX platform services.

```
C:\> ssh root@192.168.1.71
root@nsx-node1:~#
```

Switched to the NSX CLI admin user:

```
root@nsx-node1:~# su - admin
NSX CLI (Manager, Policy, Controller 9.0.1.0.24952114). Press ? for command list or ente
r: help
nsx-node1>
```

6.2 Check All Services

The first diagnostic step was to check the state of all NSX services:

```
nsx-node1> get services
```

Output (abbreviated — full output in Appendix A):

Service	State	Notes
applianceproxy	**running**	Basic proxy — starts independently
async_replicator	**stopped**	Depends on manager
auth	**stopped**	Depends on http
cluster_manager	**stopped**	Depends on http
cm-inventory	**stopped**	Depends on manager
controller	**stopped**	Depends on http
datastore	**stopped**	Corfu DB — checked separately
datastore_nonconfig	**stopped**	Corfu secondary — checked separately
http	**stopped**	**ROOT CAUSE — gateway for all API/UI**
manager	**stopped**	Core management plane
messaging-manager	**stopped**	Depends on http
monitoring	**stopped**	Depends on manager
node-mgmt	**running**	Basic node management — starts independently
node-stats	**running**	Basic node stats — starts independently
nsx-platform-client	**running**	Platform client — starts independently
nsx-upgrade-agent	**running**	Upgrade agent — starts independently
ntp	**running**	Time sync — starts independently
search	**stopped**	Depends on manager
sha	**running**	System health agent — starts independently
ssh	**running**	SSH daemon — starts independently
syslog	**running**	Log forwarding — starts independently
ui-service	**running**	UI static files — starts independently

Analysis: Only 9 out of 30+ services were running. All 9 running services are basic OS-level or node-level services that start independently of the NSX platform. Every platform service (http, manager, controller, auth, search, and all policy services) was **stopped**.

The `get cluster config` and `get cluster status` commands both returned errors at this point:

```
nsx-node1> get cluster config
% An error occurred while getting the cluster config

nsx-node1> get cluster status
% An internal error occurred, please retry execution again

nsx-node1> get cluster vip
% An error occurred while getting the cluster virtual ip
```

Analysis: These commands all require the HTTP/API service to be running. Their failure confirms the HTTP service is down.

6.3 Filesystem and Resource Check

Before attempting service recovery, disk space and memory were verified to rule out resource exhaustion:

```
nsx-node1> get filesystem-stats
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda2	11G	4.7G	5.0G	49%	/
/dev/mapper/nsx-var+log	27G	14G	12G	55%	/var/log
/dev/mapper/nsx-repository	31G	7.8G	22G	27%	/repository
/dev/mapper/nsx-config	29G	48M	28G	1%	/config
/dev/mapper/nsx-secondary	98G	230M	93G	1%	/nonconfig
tmpfs	16G	24K	16G	1%	/dev/shm

Analysis: No filesystem is close to capacity. The root filesystem is at 49%, /var/log at 55%, and all other mounts are well below any warning threshold. Disk space is **not** a contributing factor.

---

7. Phase 3 — Service Recovery

7.1 Verify Datastore (Corfu) Status

The Corfu datastore is the foundational database service. It must be running before any other platform service can start.

```
nsx-node1> get service datastore
```

Output:

```
Service name:      datastore
Service state:     running
```

Analysis: The datastore service was already running. This means the database layer is healthy and no Corfu repair is needed. Proceed to starting the HTTP service.

> **Why check datastore first?** The Corfu datastore is the NSX Manager's internal database. If it is stopped or corrupted, starting the HTTP service will fail because HTTP depends on database connectivity for configuration loading, session management, and API request processing. Always verify datastore health before starting upstream services.

7.2 Start HTTP Service

The HTTP service is the API and UI gateway. It listens on port 443 and is required by every other platform service.

```
nsx-node1> start service http
```

Wait 30 seconds for the service to initialize, then verify:

```
nsx-node1> get service http
```

Output:

```
Service name:                           http
Service state:                          running
Logging level:                          info
Session timeout:                        1800
Connection timeout:                     30
Client API rate limit:                  100 requests/sec
Client API concurrency limit:           40
Global API concurrency limit:           199
Redirect host:                          (not configured)
```

```
Basic authentication:                    enabled
Cookie-based authentication:             enabled
```

Result: HTTP service started successfully. Session timeout is 1800 seconds (30 minutes), connection timeout is 30 seconds. API rate limiting is active at 100 requests/sec. The API gateway is now accepting connections on port 443.

7.3 Start Manager Service

The Manager service is the core management plane. It handles all CRUD operations for NSX objects (segments, firewall rules, groups, etc.).

```
nsx-node1> start service manager
```

Wait 60 seconds for the service to initialize. The manager service has a longer startup time because it must:

- Connect to the Corfu datastore
- Load all policy objects into memory
- Register with the HTTP API layer
- Initialize all POLICY_SVC_* sub-services

```
nsx-node1> get service manager
```

Output:

```
Service name:      manager
Service state:     running
Logging level:     info
```

7.4 Start Controller Service

The Controller service manages the control plane — it programs the data plane on ESXi hosts and Edge nodes.

```
nsx-node1> start service controller
```

Wait 30 seconds, then verify:

```
nsx-node1> get service controller
```

Output:

```
Service name:      controller
Service state:     running
```

7.5 Verify All Services Running

After starting the three key services (http, manager, controller), all remaining services should cascade-start automatically. Wait 2–3 minutes, then check:

```
nsx-node1> get services
```

Output (abbreviated — full output in Appendix B):

Service	State
applianceproxy	**running**
async_replicator	**running**
auth	**running**
cluster_manager	**running**
cm-inventory	**running**
controller	**running**
datastore	**running**
datastore_nonconfig	**running**
http	**running**
idps-reporting	**running**
install-upgrade	**running**
manager	**running**
messaging-manager	**running**
monitoring	**running**
node-mgmt	**running**

node-stats	**running**
nsx-platform-client	**running**
nsx-upgrade-agent	**running**
ntp	**running**
search	**running**
sha	**running**
site_manager	**running**
ssh	**running**
syslog	**running**
ui-service	**running**

Stopped services (expected):

Service	State	Reason
liagent	stopped	Log Insight agent — not configured
migration-coordinator	stopped	Only active during migrations
nsx-message-bus	stopped	Not used in single-node deployments
snmp	stopped	SNMP not enabled (Start on boot: False)

Result: All 25+ platform services are now running. The 4 stopped services are expected to be stopped in this environment configuration.

---

8. Phase 4 — Cluster Validation

8.1 Cluster VIP Verification

With services running, verify the cluster VIP is assigned:

```
nsx-node1> get cluster vip
```

Output:

```
Virtual IPv4 address:   192.168.1.70
Virtual IPv6 address:   not configured
Assigned to:            ['192.168.1.71']
```

Analysis: The VIP (192.168.1.70) is active and correctly assigned to the single node (192.168.1.71).

8.2 Cluster Status Verbose

```
nsx-node1> get cluster status verbose
```

Key results by group type:

Group Type	Group Status	Node Status
DATASTORE	STABLE	UP
CLUSTER_BOOT_MANAGER	STABLE	UP
CONTROLLER	STABLE	UP
MANAGER	STABLE	UP
HTTPS	**UNAVAILABLE**	**DOWN**
SITE_MANAGER	STABLE	UP
MONITORING	STABLE	UP
IDPS_REPORTING	STABLE	UP
CM-INVENTORY	STABLE	UP
MESSAGING-MANAGER	STABLE	UP
CORFU_NONCONFIG	STABLE	UP

Overall Cluster Status: DEGRADED

8.3 HTTPS Group Status

The cluster status showed the HTTPS group as UNAVAILABLE with the node status DOWN, even though the `http` service was running. This is a **transient state** — the HTTPS cluster group registration lags behind the actual service status by several minutes.

The UI banner displayed: `Some appliance components are not functioning properly. Component health: MANAGER:DOWN, SEARCH:DOWN, UI:UP, NODE_MGMT:UP.`

This message also reflects a **stale health check** that had not yet refreshed. After waiting 3–5 minutes and refreshing the browser, the health status updated to show all components as UP.

> **Important:** Do not restart services based on the HTTPS cluster group status or the UI health banner immediately after a manual service recovery. These health indicators use cached state and periodic polling intervals. Wait at least 5 minutes before re-evaluating. If the `get service http` command shows `running` and the UI is accessible, the HTTPS group will transition to STABLE on the next health check cycle.

8.4 Browser Access Verification

URL	Result
`https://192.168.1.71`	NSX Manager UI loaded — login page displayed
`https://192.168.1.70`	NSX Manager UI loaded — login page displayed (via VIP)

Both the node IP and VIP are now serving the NSX Manager UI.

---

9. Phase 5 — Upstream Integration Recovery

9.1 VCF Operations Adapter Status

After NSX Manager recovery, the VCF Operations integration adapters needed re-validation.

Pre-recovery status (Administration → Integrations → Accounts):

Adapter	Status
lab (VCF Adapter)	Warning
nsx-vip.lab.local (NSX Adapter)	Warning

9.2 Re-validate NSX Adapter

1. In VCF Operations, navigate to **Administration → Integrations → Accounts**
2. Click the three-dot menu (⋮) next to **nsx-vip.lab.local**
3. Select **Edit**
4. Click **Validate** to test the connection
5. If the connection test succeeds, click **Save**
6. Repeat for the **lab** VCF adapter if still in Warning state

> ***Note:*** *When re-validating the NSX adapter, a warning may appear: "Error trying to establish connection. Proceed anyway?" This can occur if the certificate has not been re-cached. Click* ***Proceed*** *— the adapter will reconnect successfully once the certificate is accepted.*

9.3 Confirm Collection Status

After re-validation, wait one collection cycle (5–10 minutes) and verify:

Adapter	Status
lab (VCF Adapter)	**Collecting**
nsx-vip.lab.local (NSX Adapter)	**Collecting**
mgmt	**Collecting**
mgmt - vSAN	**Collecting**
Application Monitoring Adapter	**Collecting**

Result: All VCF Operations adapters are now in **Collecting** state. NSX metrics, alerts, and compliance data collection has resumed.

---

10. Post-Recovery Verification Checklist

Use this checklist to confirm full recovery after an NSX Manager cold start service failure:

#	Check	Command / Action	Expected Result
1	All services running	`get services`	All platform services show `running`
2	Cluster VIP active	`get cluster vip`	VIP assigned to node
3	Cluster status	`get cluster status`	Overall status: STABLE
4	UI accessible (node IP)	Browse to `https://192.168.1.71`	Login page loads
5	UI accessible (VIP)	Browse to `https://192.168.1.70`	Login page loads
6	Component health	Check UI banner	All components UP
7	VCF Ops NSX adapter	VCF Operations → Integrations	Status: Collecting
8	VCF Ops VCF adapter	VCF Operations → Integrations	Status: Collecting

9	Transport node connectivity	NSX UI → Fabric → Nodes	All nodes connected
10	Filesystem usage	`get filesystem-stats`	No filesystem above 80%

---

11. Lessons Learned and Recommendations

Lessons Learned

1. **ICMP response does not indicate service health.** The NSX Manager VM responded to ping while all platform services were stopped. Always verify service state via SSH, not just network reachability.
2. **The HTTP service is the critical dependency.** If only one service is going to fail, it will be HTTP. All other services depend on it. When troubleshooting NSX Manager unresponsiveness, check `get service http` first.
3. **Cluster status commands fail when HTTP is down.** The `get cluster config`, `get cluster status`, and `get cluster vip` commands all require the HTTP/API layer. Their failure is a symptom, not a separate problem.
4. **Service cascade startup is reliable.** Once the three key services (datastore, http, manager) are running, all other services start automatically within 2–3 minutes. There is no need to manually start each of the 30+ services.
5. **Health indicators lag behind actual state.** The UI health banner and cluster group status use cached, periodically-refreshed data. Do not make decisions based on stale health status immediately after a manual recovery.

Recommendations

#	Recommendation	Priority
1	**Create a startup script** that checks NSX service health after boot and restarts HTTP if stopped	High
2	**Configure VCF Operations alerts** for NSX Manager service state changes (not just adapter connectivity)	High
3	**Document the service start order** (datastore → http → manager → controller) in the lab runbook for quick reference	Medium
4	**Increase NSX Manager VM resources** in the nested environment (CPU: 4→6, RAM: 32→48GB) to reduce cold start failures	Medium
5	**Implement startup order in VMware Workstation** — ensure NSX Manager VM starts after vCenter and AD/DNS VMs are fully online	Medium

6	**Monitor /var/log usage** — at 55%, it is the highest-utilized filesystem and could cause issues if logs are not rotated	Low

---

12. Appendix A — Complete Service List (Pre-Recovery)

Captured at: March 26, 2026, 15:01 UTC

Service	State
applianceproxy	running
async_replicator	stopped
auth	stopped
cluster_manager	stopped
cm-inventory	stopped
controller	stopped
datastore	stopped
datastore_nonconfig	stopped
http	stopped
idps-reporting	stopped
install-upgrade	stopped
liagent	stopped
manager	stopped
messaging-manager	stopped
migration-coordinator	stopped
monitoring	stopped
node-mgmt	running
node-stats	running
nsx-message-bus	stopped
nsx-platform-client	running
nsx-upgrade-agent	running
ntp	running

search	stopped
sha	running
site_manager	stopped
snmp	stopped
ssh	running
syslog	running
ui-service	running

Running: 9 of 29 | **Stopped:** 20 of 29

---

13. Appendix B — Complete Service List (Post-Recovery)

Captured at: March 26, 2026, 15:36 UTC

Service	State
applianceproxy	running
async_replicator	running
auth	running
cluster_manager	running
cm-inventory	running
controller	running
datastore	running
datastore_nonconfig	running
http	running
idps-reporting	running
install-upgrade	running
liagent	stopped
manager	running
messaging-manager	running
migration-coordinator	stopped

monitoring	running
node-mgmt	running
node-stats	running
nsx-message-bus	stopped
nsx-platform-client	running
nsx-upgrade-agent	running
ntp	running
search	running
sha	running
site_manager	running
snmp	stopped
ssh	running
syslog	running
ui-service	running

Running: 25 of 29 | **Stopped (expected):** 4 of 29

---

14. Document Information

Field	Value
Document Title	NSX Manager 9.0.1 Cold Start Service Failure — RCA & Recovery
Version	1.0
Author	Virtual Control LLC
Date	March 26, 2026
Classification	Internal — Lab Environment
Environment	Dell Precision 7920, VMware Workstation 17.x Nested Lab
NSX Version	9.0.1.0 (Build 24952114)
Related Documents	VCF Undocumented Issues Reference, VCF Troubleshooting Handbook

---

Addendum — 2026-04-24 Extended Recovery: Corfu JVM Hang Variant

Date: April 24, 2026 **NSX Version:** 9.0.1.0 (Build 24952112) **Trigger:** Cascading appliance damage following 2026-04-18 Dell Precision 7920 Kernel-Power 41 / Disk 1 surprise-remove (see separate RCA `Precision-7920-Disk1-Swap-RCA`). `logs` appliance was recovered via KB 326323 `fsck.repair=yes` on 2026-04-23. NSX Manager exhibited a different, deeper failure this time.

A.1 What differed from the March 26 case

The March 26 cold-start scenario documented in the main body of this report assumed the Corfu datastore service was healthy — the RCA's Phase 3 recovery starts with "verify datastore running" and proceeds to `start service http`. **On 2026-04-24, that assumption did not hold.**

Aspect	March 26 (main RCA)	April 24 (this addendum)
`get services` on arrival	9 of 29 running; everything except node-level services stopped	24 of 29 running; **only `http` and expected-stopped services down**
`get service datastore`	running	running
Port 9000 actually listening	presumed yes	**NO** — socket never bound despite service "running"
Port 9823 actually listening	n/a (not checked)	**NO**
`start service http` result	started cleanly	**silent no-op** — preconditions not met
Corfu health JSON `/health`	n/a (not checked)	Layout Server UP; **Sequencer DOWN; Log Unit DOWN**
`/var/log/syslog`	cluster_manager-type errors	`Tried to get layout from 192.168.1.71:9000 but failed by timeout` (repeated, retry counter climbing to 120)
Resolution	`start http → start manager → start controller`	Service restart insufficient — required **VM reboot + wait hours for compaction**
Total recovery time	~40 minutes	**>2 hours and counting**, dominated by compaction + concurrent vSAN resync

Key insight: "`get service datastore = running`" reports the systemd/init-script state of the Corfu wrapper, **not** whether Corfu has successfully bound its listener ports. The JVM can be alive, the wrapper can be active, and `get services` can report "running" while port 9000/9823 are still not accepting connections. Clients see this as "Connection refused" or "layout timeout" even though every management-layer indicator says the service is up.

A.2 Updated diagnostic sequence

Before starting any services per the main RCA's Phase 3, add these three checks. If any of them disagree with `get services`, you are in the Corfu-JVM-hang variant and the main RCA's recovery procedure will not work.

```
# As root on NSX Manager:
ss -tln | grep -E ':9000|:9823'      # Must show listeners on BOTH
curl -ks https://127.0.0.1:9000/health | python3 -m json.tool
grep 'layout from' /var/log/syslog | tail -20
```

Expected when Corfu is actually healthy:

- `ss` output shows listeners on 9000 and 9823 bound to the node IP
- `/health` returns all runtime services `"status": "UP"` — specifically Layout Server, **Sequencer**, and **Log Unit**
- No recent "layout from ... failed by timeout" entries

If any of the three fails, do **not** try to `start service http` — it will silently no-op. Proceed to A.3.

A.3 Recovery procedure for the Corfu JVM hang variant

Applied 2026-04-24; matches KB 390592 (JDK/corfu compactor hang) and KB 395500 (tanuki JVM OOM during compaction).

1. **Attempt `restart service datastore` first.** Non-destructive. If Corfu JVM is only mildly hung, this may rebind the ports. If port 9000 remains silent after ~2 minutes, the JVM hang is more severe — proceed to step 2.
2. **Full VM reboot** (NSX admin CLI `reboot`, answer `yes` at prompt). This is the KB 390592 documented remediation. On a single-node cluster this is a full control-plane outage of 10–15 min for boot + additional wait for compaction.
3. **Wait for compaction.** Port 9000 will NOT listen until Sequencer and Log Unit services come up inside the Corfu JVM — this happens only after compaction progresses past a recovery threshold. **There is no time estimate in the Broadcom KB.** In a constrained nested environment with concurrent vSAN resync, this observed to exceed 2 hours.
4. **Once port 9000 is listening and `/health` shows Sequencer + Log Unit UP**, resume the main RCA's Phase 3 starting at §7.2 `start service http`.

A.4 Environmental amplifiers observed on 2026-04-24

The nested lab made everything worse. Document which of these applies next time to estimate realistic recovery duration.

Amplifier	Observed value	Effect
Concurrent vSAN resync with hosts in MM	449.97 GB → 186 GB across resync window	Compaction fought resync for disk I/O; load avg peaked at 188
vSAN hosts active	2 of 4 (esxi02, esxi03 only)	Halved storage bandwidth during peak recovery
NSX Manager guest I/O wait	46–68% at peak	CPU idle waiting on disk — nearly pure I/O bottleneck
NSX Manager guest CPU steal	**0.0%**	Nested ESXi not CPU-starving the VM — so powering down peer VMs for CPU reasons would not have helped

What helped mitigate:

- Bringing `esxi01` out of maintenance mode (vSAN went 2 → 4 active hosts): vSAN resync dropped ~383 GB in 20 minutes
- Powering off non-essential VCF appliances (`logs` , `collector`) to reduce vSAN write pressure

What did NOT help (and would have hurt):

- Adding VM RAM (32 → 48 GB) — not memory-pressed; JVM heap is fixed at `-Xmx1282m` and wouldn't use additional RAM
- Powering down VMs for CPU relief — 0% steal time meant nested ESXi was giving NSX Manager all the CPU it asked for

A.5 Corrected service-dependency mental model

The main RCA (§4.2) shows a strict chain: `datastore → http → manager → controller` . On 2026-04-24 this chain was partially violated — `manager` , `controller` , `auth` , etc. were all running while `http` was stopped and Corfu wasn't serving. This suggests these services had started at some earlier point when Corfu was serving, and remained running after Corfu hung — rather than being strictly blocked from starting without Corfu.

Revised understanding: the dependency chain controls *startup gating*, not *runtime liveness*. Once services are up, they retry their Corfu connections and continue running even if Corfu becomes unavailable. They just can't serve API requests without http, and http depends on Corfu actually serving.

A.6 Lessons Learned added

Extending §11:

- **L-7** — `get services` reports systemd-layer state, not actual socket binding. **Always verify with** `ss -tln | grep :9000` **after** `get service datastore = running` **.** If the init script says up but the port isn't listening, you are not in a normal cold-start scenario.
- **L-8** — `start service http` silently no-ops when its preconditions (Corfu serving) aren't met. No error is returned. **Verify with a follow-up** `get service http` **— if still** `stopped` **, something deeper is**

broken.

- **L-9** — In nested labs with concurrent vSAN resync, NSX Manager compaction time is dominated by shared-storage I/O contention, not JVM or network tuning. Keeping all vSAN hosts active and staging VM migrations/shutdowns outside of NSX recovery windows substantially reduces recovery time.
- **L-10** — `/var/log/corfu/tanuki.log` printing "Serving on port 9000" at JVM startup is **aspirational**. It means the JVM reached the server-main entry point; it does NOT mean the socket is bound and accepting. Cross-check with `ss`.
- **L-11** — When vMotion is broken due to transport-node state issues and NSX Manager is down, **NSX Manager cannot be brought back via vMotion-dependent workflows.** Cold-restart-in-place (or cold migration via VMFS/vSAN registration) is the only tool available. Plan recovery order accordingly: NSX Manager must recover before host TN state can be repaired, not the other way around.

A.7 Observed timeline (2026-04-24)

UTC	Event
12:51	`/etc/vmware/dvsdata.db` mtime on esxi01 (host-side NSX activity)
13:23	SSH diagnostic: `get services` shows 24 running, http stopped, port 9000 not listening
13:32	`restart service datastore` issued
13:39	datastore reports running; port 9000 still not listening; layout timeouts continue
13:54	`reboot` issued with `yes` confirmation
13:58	VM unreachable (shutdown progressing)
14:12	VM ping returns (OS up, 13.5 min down — longer than typical for NSX)
14:15	corfu-server systemd active
14:19	tanuki.log prints "Serving on port 9000" (aspirational)
14:22	Load avg 26.88; port 9000 still not listening; vSAN resync 449.97 GB / 28 objects
14:56	After esxi01 out of MM: vSAN resync 66.91 GB / 19 objects (~383 GB drained in 20 min)

14:57	I/O wait 0%, load avg 188 (CPU-bound catch-up after I/O unblock)
15:08	`logs` and `collector` VMs shut down to reduce vSAN writes
15:20	Resync 186.88 GB / 27 objects (uptick attributed to new rebalance/re-protection triggered by host rejoin + cold migrations)
15:22	Resync 186.05 GB / 27 objects; port 9000 still not listening; estimated ~5.5 hr remaining at observed rate

---

Addendum updated 2026-04-24. Main RCA body unchanged.

> ***Cross-reference:*** *the successful fresh-redeploy procedure that was used to recover from the failure mode in this addendum is documented in the rebuild plan at* `E:\VCF 9 Full documentation\VCF-Operations-Library\01-Deployment\VCF9-Host-vCenter-Rebuild-Plan.md` *— Phase 4 (NSX Manager Fresh Redeploy).*

Addendum — 2026-05-05 Power-Outage + HA-Reset Cascade: btrfs Read-Error Variant

Date: May 5, 2026 **NSX Version:** 9.0.1.0 (Build 24952114) **Trigger:** Hard power-off of the host (Dell Precision 7920) at 07:20 EDT 2026-05-05 (front-bay backplane failure interrupted host power). After power restore, recovery operations on `esxi01` introduced a brief loss of DVS uplinks during a host network reconfiguration; vSphere HA interpreted this as host isolation and **hard-reset NSX Manager mid-write** — kicking off this addendum's failure mode.

C.1 What differed from prior cases

Aspect	March 26 main RCA	April 24 Addendum A	**May 5 Addendum C**
Initial console state	normal login prompt	normal login prompt	**emergency mode** `(or press Control-D to continue)`
Filesystem layer	clean	clean	`btrfs: error (device dm-0): bdev /dev/mapper/nsx-aspq read I/O error` repeating in dmesg
Underlying cause	services stopped	Corfu JVM hang	**vSAN component on running host still in active-rebuild — reads to that component returned** `read I/O error`**, btrfs went read-only and dropped to emergency**
HA involveme	none	none	**HA hard-reset NSX Manager during a 15-second DVS-uplink gap on the host running NSX Manager**

nt			
First indicator	services 9/29 running	port 9000 not listening	**btrfs Buffer I/O error spamming console; emergency-mode `sulogin` prompt**
Resolution	start http→manager →controller	reboot + wait compaction	**vSAN object-component lookup → cold-migrate VM to a host whose local component is fully healthy → power-on succeeds → btrfs mounts cleanly**

Key insight: when an NSX Manager VMDK has multiple RAID-1 components and one is in `componentState: 10` (`ABSENT_RECOVERING` , with `bytesToSync > 0` and `isResyncRunning = 1`), reads issued by the host running the VM may preferentially target the local component, which is incomplete. btrfs sees these as bare I/O errors (not checksum errors), goes read-only, and drops to emergency. This looks identical to filesystem corruption — but it isn't. **It's a transient vSAN read-path issue that resolves either by waiting for the local component to finish syncing OR by moving the VM to a host whose local component is fully healthy.**

C.2 Diagnostic sequence

Use these steps before assuming btrfs damage:

```
# 1. Confirm VM is online at the network layer (kernel up)
ping <nsx-manager-ip>     # should reply

# 2. Pull the VM home UUID via PowerCLI / vCenter
$nsx = Get-VM nsx-manager
$nsx.ExtensionData.Config.Files.VmPathName     # gives you [datastore] <uuid>/...
Get-HardDisk -VM $nsx | Select-Object Filename

# 3. From any vSAN node host, query CMMDS for those object UUIDs:
cmmds-tool find -t DOM_OBJECT 2>/dev/null | grep -E '<vm-home-uuid-prefix>'

# 4. In each returned object, look for "componentState" inside ("Component" ...) entrie
s:
#    componentState 5 = ACTIVE (healthy)
#    componentState 10 = ABSENT_RECOVERING (resyncing)
#    Look for ("isResyncRunning" i1) and ("bytesToSync" lN) — those mean the component i
s mid-rebuild

# 5. If the VM is running on the host whose faultDomainId owns the resyncing component,
#    that's why btrfs reads are failing. Cold-migrate the VM to a host that owns the
#    fully-healthy component(s) instead.
```

C.3 Recovery procedure for the btrfs read-error variant

Applied 2026-05-05.

1. **Power off NSX Manager** if not already (graceful from inside the VM is impossible at this point — use vCenter Power → Power Off / "force").
2. **Identify a target host with healthy local components** for the NSX Manager VMDKs (any host whose faultDomainId owns a `componentState 5` component for those objects).
3. **Cold migrate the VM to that target host.**

C.3.1 Cold migration may be blocked: pg-vm-mgmt "not accessible"

When NSX is broken on all hosts (NSX Manager dead → host-side NSX vibs can't validate state), **vCenter rejects cold migration with**:

```
Network interface 'Network adapter 1' uses network 'pg-vm-mgmt (vcenter-cl01-vds01-ne
w)',
which is not accessible.
```

This is misleading — the portgroup is technically reachable on every host (verified by `Get-VMHost.ExtensionData.Config.Network.ProxySwitch`); NSX vibs are blocking the migration validation.

Workaround that worked 2026-05-05:

```powershell
$nsx = Get-VM nsx-manager
$nic = Get-NetworkAdapter -VM $nsx
$pgMgmt = Get-VDPortgroup -Name pg-mgmt          # NOT pg-vm-mgmt — pg-mgmt is the host
                                                 # mgmt portgroup, not NSX-tracked
Set-NetworkAdapter -NetworkAdapter $nic -Portgroup $pgMgmt -Confirm:$false
Move-VM -VM $nsx -Destination 'esxi02.lab.local' -Confirm:$false
Start-VM nsx-manager -Confirm:$false
# Once NSX is healthy and you've verified everything, change NIC back:
# Set-NetworkAdapter -NetworkAdapter $nic -Portgroup (Get-VDPortgroup -Name pg-vm-mgmt)
-Confirm:$false
```

The PowerCLI cmdlet for DVS portgroups requires `-Portgroup <VDPortgroup-object>`, NOT `-NetworkName "string"` (deprecated since PowerCLI 13).

C.3.2 DRS will interfere on power-on

With DRS enabled, `Start-VM` **triggers automatic placement** that may relocate the VM back to its origin host before powering on. Observed 2026-05-05:

```
13:14:09  Relocate task issued (cold migrate to esxi02)
13:19:43  Successfully relocated to esxi02
13:27:17  Power On task issued
13:27:59  DRS relocate: esxi02 → esxi01    ← unexpected
13:30:31  Powered on (on esxi01)
```

If the goal of the migration was to read from a specific host's local component, this defeats the purpose. **Set DRS to Manual or pin the VM with an affinity rule before issuing Power On.**

In our case it didn't matter — by the time DRS placed the VM back on esxi01, the local component had also healed enough that the boot succeeded anyway.

C.4 Compounding issue: services hung post-boot (Corfu hang re-emerged)

After the OS booted cleanly (no btrfs errors), NSX **platform** services exhibited the same symptoms as Addendum A — port 22 / 80 / 443 all closed, CPU usage 68 MHz idle, memory consumed flat at 1.6 GB, no activity in `/var/log/corfu/`. NSX Manager OS layer is up but the application stack never starts.

This is the same Corfu JVM hang variant from Addendum A, just preceded by the btrfs episode. Same recovery path applies (`restart service datastore` if SSH comes up; otherwise reboot + wait compaction).

C.5 Environmental amplifiers observed on 2026-05-05

Amplifier	Observed value	Effect
Concurrent vSAN resync (post-outage rebalance after esxi01 rejoin)	121 GB / 13 objects	Compaction fights resync for I/O — same dynamic as 4/24
HA enabled at start of network maintenance	Yes	**Direct cause of the btrfs incident** — hard-reset on transient uplink loss
DRS enabled	Yes	Defeated manual VM placement on power-on
Native Key Provider with no hardware TPM	Yes	Generated separate "Host Requires Encryption Mode" alarm cascade — unrelated noise during recovery

C.6 Lessons Learned added (extends §11, A.6, B's lessons)

- **L-12** — **Disable cluster HA before any host-network maintenance.** A DVS uplink swap, a vSwitch migration, or any operation that briefly leaves a host without uplinks will cause HA to interpret it as host isolation and reset VMs. This was the single root-cause decision behind the entire 2026-05-05 NSX Manager incident.

- **L-13** — `Add-VDSwitchPhysicalNetworkAdapter` **is not atomic at the host layer.** PowerCLI presents it as a single transaction, but on the host side the proxy switch reconfigures with a real-world gap. If the operation involves removing the only uplink that backs DVS portgroups carrying running VMs, HA reacts during the gap.
- **L-14** — **vSAN component-level health is not reflected in object-level health.** An object with `Health: Healthy` can have one component in `componentState 10` (ABSENT_RECOVERING) — and reads to that component return I/O errors. Always check `cmmds-tool find -t DOM_OBJECT` for component states when a VM exhibits read-path symptoms but vSAN summary says healthy.
- **L-15** — `pg-vm-mgmt` **is NSX-tracked even though it is a regular VLAN-backed DVS portgroup.** When NSX Manager is dead, host-side NSX vibs reject vCenter migration validation for any VM on this portgroup with a misleading "network not accessible" error. The workaround is to temporarily move the NIC to `pg-mgmt` (host management network, not NSX-tracked) for the duration of the migration.
- **L-16** — **DRS placement defeats manual cold-migration intent on** `Start-VM`**.** With DRS enabled, the very next `Power On` after a cold migrate will trigger an automatic placement that frequently undoes the migration. Set DRS to Manual or set a temporary affinity rule before powering on.
- **L-17** — **PowerCLI's** `Set-NetworkAdapter -NetworkName <string>` **is deprecated for DVS portgroups since PowerCLI 13.** Use `-Portgroup <VDPortgroup-object>` instead. The deprecated syntax silently warns, then hangs the cmdlet waiting for confirmation.
- **L-18** — **Btrfs read I/O errors on NSX Manager are *not* automatic evidence of filesystem damage.** Before invoking the rebuild plan, always check whether the underlying vSAN components are still resyncing. The btrfs damage signal looks identical, but the recovery is completely different (wait or cold-migrate vs. fresh redeploy).

C.7 Observed timeline (2026-05-05)

EDT	Event
07:20	Hard power-off (front-bay backplane) — 7920 host crashes; nested ESXi VMs all hard-reset
~10:00	Lab restored; 4 ESXi hosts boot; esxi01 boots with new SMBIOS UUID (Workstation copied-VM behavior) — comes up with reset network config and empty NSX prep
10:30	esxi01 rejoined to vSAN cluster via PowerCLI sequence (DVS Add Host + uplinks + vmk1 vSAN)
~11:25	DVS uplink swap on esxi01 (vmnic2/3 → vmnic0/1) for parity with other hosts. **HA still enabled.** Brief gap leaves esxi01 with zero uplinks for ~15 sec. **HA hard-resets** `nsx-manager` (the only mgmt VM running on esxi01 at the time).
11:25-11:40	NSX Manager OS comes back up, but `nsx-manager.vmdk` has one component on esxi01 still in `componentState 10`, `bytesToSync 1792262144`, `recoveryETA 4588 sec`. btrfs reads on `/dev/mapper/nsx-aspq` return `Buffer I/O error`. Boot drops to emergency mode (`sulogin: tcgetattr failed: Input/output error`).
12:18	User screenshot of emergency-mode console

12:30	CMMDS analysis identifies the resyncing component on esxi01; healthy components exist on esxi02/esxi03
13:14	Cold migration esxi01 → esxi02 issued via PowerCLI (NIC temporarily switched to `pg-mgmt` to bypass NSX-tracked-portgroup validation block)
13:19	Cold migration completes
13:27	Power On issued; DRS relocates esxi02 → esxi01 mid-power-on
13:30	NSX Manager powers on on esxi01. **btrfs mounts cleanly this time** — local component had healed enough during the migration window. Boot proceeds through cloud-init, network up, mounts ok.
14:00	NSX OS fully booted, network up, ssh service did **not** come up. Resource usage flat at 68 MHz / 1.7 GB consumed. Same as Addendum A Corfu hang variant.
14:30	Boot screen shows cloud-init finished, all systemd targets reached. NSX platform stack still not running. **Awaiting compaction or operator intervention per A.3.**

C.8 Cross-references

- Underlying outage RCA: `I:\VCF 9 Full documentation\VCF-Operations-Library\05-Health-Checks\Post-Outage-Health-Report-2026-05-05.md` — the storage / SMART / GPT-signature audit done immediately after the outage.
- Network-maintenance-causes-HA-reset memory pattern (Claude operator note): `feedback_disable_ha_before_uplink_migration.md` — the lesson saved into operator memory after this incident.
- Decision criteria for fresh redeploy vs. wait-it-out: see B.1 (now in `VCF9-Host-vCenter-Rebuild-Plan.md` Phase 4).

C.9 Resolution — Backup Restore Path (2026-05-05 evening)

Outcome: full recovery via NSX backup restore in ~45 minutes elapsed, **without** building any NSX configuration objects from scratch.

Backup discovery

Hourly NSX backups had been pushed via SFTP to the management workstation (`192.168.1.160`) since the original NSX cluster was first deployed:

Item	Value
SFTP server	`192.168.1.160:22` (Windows OpenSSH on the management workstation)
Username	`admin` (Windows local user, dedicated `Match User admin` block in `sshd_config`)
Chroot directory	`I:\VCF-Depot\backups` (per `Match User admin → ChrootDirectory` in `sshd_config`)
NSX-side directory path	`/nsx-manager` (relative to chroot)

ECDSA host fingerprint	`SHA256:d9c+ICSL5rOMGns/3ig8YrcRnBCKE7zjD+ZlIezPkx8` (NSX-T accepts only ECDSA SHA256)
Backup cadence	hourly at minute :24 UTC
Latest pre-outage backup	`2026-05-05T10:24:01 UTC` = 06:24 EDT, ~1 hour before the 07:20 EDT outage
Cluster TAR size	~48 MB (encrypted with backup passphrase)
Node TAR size	~1.2 MB

The OLD NSX cluster's UUID `5f933642-e62c-189e-fc7f-d3463db9eeea` is encoded in the backup directory path — restore-onto-fresh-cluster (different new cluster UUID) is fully supported.

Restore procedure that worked

1. **Configure backup destination on the freshly-deployed NSX Manager** via `PUT /api/v1/cluster/backups/config`:

```
{
  "passphrase": "<original encryption passphrase, must match exactly>",
  "remote_file_server": {
    "server": "192.168.1.160",
    "port": 22,
    "directory_path": "/nsx-manager",
    "protocol": {
      "protocol_name": "sftp",
      "ssh_fingerprint": "SHA256:d9c+...",
      "authentication_scheme": {
        "scheme_name": "PASSWORD",
        "username": "admin",
        "password": "<sftp user password>"
      }
    }
  },
  "inventory_summary_interval": 240,
  "backup_enabled": false
}
```

NSX 9 passphrase complexity rule (error 29324): must be ≥ 8 chars and contain at least one each of lowercase, uppercase, numeric, **special**. The original passphrase from the OLD cluster must satisfy this AND must match exactly — the cluster TAR is encrypted with it. Verify the original passphrase format before attempting restore.

2. **List available backup timestamps** via `GET /api/v1/cluster/restore/backuptimestamps`. Response includes timestamps in epoch ms.
3. **Start the restore** via `POST /api/v1/cluster/restore?action=start` with the chosen `time stamp`, `node_id`, `ip_address` of the OLD cluster.
4. **Restore advances through 34 steps automatically** but pauses at step 17 (`ConfirmCmConnectivity`) for operator confirmation that the Compute Manager is registered and Up. **The CM record itself is restored from backup** — no manual P4.8 registration is needed; the API just needs verification that vCenter is reachable.

   Verify with `GET /api/v1/fabric/compute-managers/{id}/status` — must return `connec tion_status: UP, registration_status: REGISTERED` — then advance:

```
POST /api/v1/cluster/restore?action=advance
Body: {"data":[{"id":"<instruction-id>","resources":[]}]}
```

The instruction id is in the `instructions[0].id` field of the restore status payload.

5. **Restore completes** at step 33/34 (StartQuartz). Total elapsed in this run: 32 minutes.

What was restored vs. what required post-restore work

Restored automatically from backup	Required post-restore
Compute Manager registration (vCenter) — REGISTERED + UP within ~1 min of restore	Per-host TN resync for hosts whose NSX state was wiped during the recovery window (3 of 4 hosts here)
All Transport Zones (overlay + VLAN, including the user-created `vcenter-cl01-tz-ov erlay01` and `vcenter-cl01-tz-vlan01`)	(none — automatically rediscovered by the resync)
Uplink Profile `vcenter-cl01-up-host01`	
Transport Node Profile `vcenter-cl01-tnp-01`	
TEP IP Pool with allocations	
All segments, firewall rules, groups, services, NAT, etc.	
Hosts that retained their NSX state during the recovery window came back **without any operator action** (esxi03 in this run — its NSX state was deliberately not cleaned because vCenter could not be evacuated off of it; that "deferral" turned out to be the correct call)	

For hosts that had NSX state cleaned in P4.4 OR were freshly-installed (rebuilt today: esxi01 and esxi04; cleaned today: esxi02), trigger NSX to re-push the TN config:

```
curl -sk -u "admin:<pw>" -X POST \
  "https://<nsx-mgr>/api/v1/transport-nodes/<tn-id>?action=resync_host_config"
```

NSX returns `HTTP 200` and the per-host TN deployment progresses from `pending/failed` back through `Configuring` to `success`. vmk10/11 (overlay TEPs) and vmk50 (hyperbus) reappear automatically.

Soft lockup observations during restore (under load)

During the heaviest restore phases (steps ~7-17, database extraction + cluster import) the NSX VM console showed kernel-level RCU/CPU soft-lockup warnings:

```
watchdog: BUG: soft lockup - CPU#3 stuck for 22s! [pyc.cortuxb.run:84176]
watchdog: BUG: soft lockup - CPU#1 stuck for 22s! [bbreplay:91519]
watchdog: BUG: soft lockup - CPU#3 stuck for 23s!
```

These resolved on their own as restore progressed past the I/O-heavy phases. Per Addendum A.4, these warnings under sustained Corfu/JVM heap I/O against vSAN can be transient — they are NOT automatic evidence that restore must be aborted. **If the restore status remains `RUNNING` and step number continues to advance, let it proceed.** Abandoning at step 25+ would lose progress that cannot be resumed (a fresh restore would have to start over from step 0).

Cost comparison vs. P4.5+ fresh-config path

Phase	Fresh config (P4.7-P4.10)	Backup restore
Extension cleanup (P4.7)	~5 min PowerCLI	not needed (backup restore handles it)
Compute Manager registration (P4.8)	~5 min UI clicks + handles `error 90348 trusted root certificate` retries	not needed (CM record restored, just confirm Up)
TZ + Uplink Profile + TEP Pool + TN Profile (P4.9)	~15 min UI clicks across four objects	not needed
Apply TN Profile to cluster (P4.10)	~10 min, hosts transition `Not Configured → Configuring → Success`	one `resync_host_config` POST per affected host (~30 sec each)
Total	**~35 min after Manager is up**	**~5 min after Manager is up**

For the steps that lead up to "after Manager is up" (P4.3-P4.6 — destroy, per-host cleanup, ovftool deploy, resource bump), backup restore does not change anything — those still apply.

When to choose backup restore vs. fresh config

Per memory feedback `feedback_no_assumptions.md` — verify before claiming a backup is usable. Use this checklist:

- ✅ Backup TAR files actually present at the expected SFTP/object-store location
- ✅ The latest pre-incident timestamp is acceptable (rule of thumb: ≤ 24 hours stale; this run used 1 hour stale)
- ✅ The original encryption passphrase is known and matches NSX 9 complexity rules
- ✅ The OLD cluster UUID in backup paths is recoverable (it's encoded in the directory name)
- ✅ vCenter is reachable from the new NSX Manager (CM restore re-establishes connection)

If all five are true, **always choose backup restore over fresh config**. The savings are real and the risk surface is dramatically smaller (no chance of misconfiguring TZ/TEP/Uplink/TNP fields under operator pressure).

If the backup is unusable (passphrase lost, TAR corrupt, or incompatible NSX version), fall back to P4.7-P4.10 fresh-config.

C.10 Lessons Learned added (extends C.6)

- **L-19** — **Always check for available NSX backups BEFORE starting fresh config.** The runbook was written assuming "no backup available" (Addendum B premise). When backups DO exist, restore is dramatically faster (~5 min post-Manager vs. ~35 min for fresh config) and removes the entire risk class of mis-typed object names, IP pool sizing errors, and `error 90348 trusted root certificate` retries on Compute Manager registration.
- **L-20** — **Deferring per-host NSX cleanup is sometimes the right call.** When P4.4 cleanup of a host is blocked (in this run: esxi03 hosted vCenter and could not be evacuated due to NSX-vibs blocking cold migration even with the `pg-vm-mgmt → pg-mgmt` NIC swap workaround from Addendum C), deferring that host's cleanup until after a successful restore is non-destructive — the host's existing NSX state simply re-binds to the new Manager during restore and arrives in `state=success` automatically. Cleaning a host loses this advantage.
- **L-21** — **NSX 9 backup restore requires passphrase complexity match.** The original encryption passphrase must satisfy `≥ 8 chars + lower + upper + numeric + special` (NSX 9 error 29324). If you receive `httpStatus BAD_REQUEST error_code 29324`, the supplied passphrase is missing a class — confirm the original phrase before retry; the cluster TAR cannot be decrypted with anything other than the exact passphrase used at backup time.
- **L-22** — **Soft-lockup messages during restore are not always a redeploy trigger.** Per P4.1 decision criterion #5, kernel-level RCU stalls indicate corruption, but during a restore those messages can be transient under restore I/O load. **Use restore status `step_number` advancing as the live signal** — if step keeps climbing through 34, the restore is healthy regardless of console noise. Only abandon and redeploy if step stalls AND status flips to `FAILED`.

- **L-23** — **`POST .../cluster/restore?action=advance` body schema** (undocumented in some KBs) is:

  ```
  {"data":[{"id":"<instruction-id>","resources":[]}]}
  ```

  where `<instruction-id>` is the `instructions[0].id` from the current restore status payload. Empty `resources: []` is acceptable when the operator has already verified the prerequisite externally (e.g., CM is `connection_status=UP` per `/api/v1/fabric/compute-managers/{id}/status`).

C.11 Observed timeline — restore phase (2026-05-05)

EDT	Event
17:42	`POST /api/v1/cluster/restore?action=start` — restore initiated against backup timestamp `1777976641000` (06:24 EDT)
17:43	Step 4/34 RestoreDatabases (RUNNING)
17:57	Step 8/34 CopyInventoryBackupFromRemote
17:58	Step 17/34 ConfirmCmConnectivity — paused for operator confirmation. CM verified `connection_status=UP, registration_status=REGISTERED`.
17:59	`POST .../restore?action=advance` with empty resources
18:04	Step 32/34 SwitchingRestoreOperationStep
18:14	Step 33/34 StartQuartz (SUCCESS). Restore status SUCCESS. **32-minute restore.**
18:15	Verified mgmt/control STABLE, CM REGISTERED+UP, TZs/Uplink Profile/TNP all present, 4 TNs in inventory
18:16	esxi03 TN already in `state=success / deploy=success` — never had NSX state cleaned, came back automatically
18:17	esxi01, esxi02, esxi04 TNs in `pending/failed` — NSX state was wiped during the day's recovery

18:18	`POST resync_host_config` to all 3 — HTTP 200 each. Host TN re-prep initiated.

Addendum updated 2026-05-05 (final, includes restore resolution). Main RCA body unchanged.

Addendum — 2026-05-06 Post-Restore Decay Variant: Empty `mpaconfig.json` on Cleaned Hosts → Fresh Redeploy

Date: 2026-05-06 (continuation of 2026-05-05 power-outage recovery) **Trigger:** Backup-restore from 2026-05-05 18:14 UTC completed SUCCESS technically (per Addendum C.9), but left 3 of 4 hosts (`esxi01` , `esxi02` , `esxi04`) in `state=pending/failed` because their NSX state had been wiped during the same day's P4.4 cleanup *before* the restore was attempted. The restored NSX Manager could not push fresh `mpaconfig.json` to those hosts (broker/account info), and the hosts could not register with the manager (no broker info). Every documented per-host API repair action returned HTTP 200 with no host-side effect. After exhausting all API and CLI repair options and reading every applicable Broadcom KB, the resolution path was a **fresh redeploy** identical to the one in `VCF9-Host-vCenter-Rebuild-Plan.md` Phase 4 (and to Addendum B in this document, which was moved to the rebuild plan). This addendum documents the post-restore stuck-state diagnosis, the failed mitigations (so future operators know not to repeat them), the verification work that justified abandoning the restore, and the executed redeploy.

D.1 What differed from prior addenda

Addendum	Failure mode	Repair	Worked
A (2026-04-24)	Corfu DB corruption inside running NSX Manager (RCU stalls)	Fresh redeploy (no backup available)	Yes (~2h 40min)
B (moved to rebuild plan)	Same as A	Same as A	Yes
C (2026-05-05)	btrfs read-error / hung services after power-off	initramfs `fsck.repair=yes` , then **backup restore** from hourly SFTP backup	Yes (~32min restore window)
D (this section, 2026-05-06)	**Restore SUCCESS but 3/4 hosts stuck with empty `mpaconfig.json`**	**Fresh redeploy (Phase 4)** — restore-recovered state is unreachable for cleaned hosts	**In progress at section write; Phase B deploy complete, awaiting Phase C-E**

The defining feature of D is that the NSX Manager itself is healthy (post-restore APIs respond, cluster STABLE, CM REGISTERED+UP). The unhealthiness is on the host side — cleaned hosts cannot rejoin a restored manager that uses the OLD cluster's identity, because the restore did not push a new fresh-host registration. There is no documented Broadcom procedure to recover this combination.

D.2 Pre-condition that created the trap

During 2026-05-05 P4.4 cleanup the operator ran `nsxcli -c 'del nsx force'` on `esxi01`, `esxi02`, `esxi04`. `esxi03` was deferred because it hosted vCenter and could not be evacuated under conditions of broken vMotion (cold-migration was blocked by NSX VIBs — per Addendum B Phase C edge case). This cleanup wiped `/etc/vmware/nsx-mpa/mpaconfig.json` on those 3 hosts, leaving the file as the empty template:

```
{
   "AccountName": "",
   "SharedSecret": "",
   "RmqClientType": "",
   "RmqBrokerCluster": [
      { "BrokerIpAddress": "", "BrokerPort": "", ... }
   ]
}
```

The backup taken at 06:24 EDT (10:24 UTC) on 2026-05-05 captured the **manager-side** view of the four-host TN state when all hosts were healthy. After cleanup, the manager's TN objects (and the cluster's TN-Profile-applied compute collection) referenced fabric-node identities that the cleaned hosts no longer hold. When the backup was restored at 17:42-18:14 UTC, the manager came back referring to those identities. The hosts that retained their `mpaconfig.json` (only `esxi03`) re-bound automatically. The hosts that had been cleaned could not, because:

1. They had no `mpaconfig.json` → MPA daemon refused to authenticate to the manager (`Empty/Invalid MP cluster configuration` warning every 60s).
2. The manager could not push `mpaconfig.json` because the hosts were not registered as fabric nodes.
3. Re-running `nsxcli -c 'join management-plane'` from the host creates a **new** fabric-node UUID (which the manager then tracks alongside the stale TN), not a rebind to the existing TN identity.

This is a chicken-and-egg deadlock with no API surface in NSX 9 to break out of.

D.3 Diagnostic sequence (2026-05-06, UTC)

Time	Probe	Finding
02:35	`GET /api/v1/transport-nodes/state`	esxi03 success/success; esxi01 pending/failed; esxi02 success(data plane up)/failed; esxi04 failed/success retrying
02:36	TN failure_messages	26080 "*Software status file does not exists on the host. Verify that nsx-sfhc is running on host.*"; 8801 "*Failed to send HostConfig RPC to MPA. Error: Unable to reach client, application SwitchingVertical.*"; 26116 "*Failed to get response from NSX-SFHC component.*"

02:38	SSH each broken host: `/etc/init.d/nsx-mpa status`	NOT running on all 3. Same for nsx-sfhc, nsx-proxy, nsx-opsagent. esxi02 had nsx-proxy and nsx-opsagent and nsx-cfgagent partially running.
02:42	NSX VIB inventory (`esxcli software vib list | grep nsx`)	All 24 VIBs present at correct version (`9.0.1.0-9.0.24952112`) on all 3 broken hosts. **NOT a VIB problem.**
02:50	`cat /etc/vmware/nsx-mpa/mpaconfig.json` on esxi01 vs esxi03	**esxi01: empty template. esxi03: populated** with `AccountName=cvn-hv-798d33ae-1587-42f9-b84f-8f070660f901` , `SharedSecret=3DAYRyV...` , `BrokerIpAddress=192.168.1.71` , `BrokerSslCertThumbprint=03d66dd0...` . Same comparison on esxi02 and esxi04: empty.
02:55	esxi02-specific: `cat /etc/init.d/nsx-mpa | grep -A2 decom`	Init script logic at lines 150-167: if `/etc/vmware/nsx/appliance-info.xml` is parseable AND `/etc/nsx_issue` does not contain `manager` , the script *creates* `/etc/vmware/nsx-mpa/.decom_state` and refuses to start MPA with: "*Not starting decommisioned NSX-Management-Plane-Agent*".
02:58	`ls /etc/vmware/nsx-mpa/.decom_state` on esxi02	Present (0 bytes, mtime 2026-05-05 14:08 — created during P4.4 cleanup)
03:00	`cat /etc/vmware/nsx/appliance-info.xml` cross-host	esxi01 + esxi04: placeholder (`# INVALID XML. Do not remove this line or edit this file manually.`). esxi02 + esxi03: full valid XML written by restore at 2026-05-06 01:57-01:58 UTC. **esxi02 hits the decom-detection logic; esxi03 does not because its MPA was already running and the script's start-time check never re-fired.**
03:05	`esxcli network ip connection list | grep 192.168.1.71` on broken hosts	No ESTABLISHED connections to manager. Only esxi03 had ESTABLISHED on port 5671 and 1234.

D.4 Mitigations attempted that did NOT resolve the state (verified empirically; future operators: do not repeat these)

#	Action	API/CLI	Result
1	Per-host config push	`POST /api/v1/transport-nodes/<id>?action=resync_host_config`	HTTP 200, no observable host-side change. mpaconfig.json remained empty.
2	Inventory resync	`POST /api/v1/transport-nodes/<id>?action=restart_inventory_sync`	HTTP 200, no effect

3	Cluster-config restore	`POST /api/v1/transport-nodes/<id>?action=restore_cluster_config`	HTTP 200, no effect
4	Per-discovered-node reapply	`POST /api/v1/fabric/discovered-nodes/<ext-id>?action=reapply_cluster_config`	HTTP 200, returned full TN spec, **no host-side effect**
5	TNC retry	`POST /api/v1/transport-node-collections/<id>?action=retry_profile_realization`	HTTP 200, no effect on broken hosts
6	Direct daemon restart	`/etc/init.d/nsx-mpa restart` on esxi01 + esxi04	Daemon started via watchdog, but emitted: `WARNING Empty/Invalid MP cluster configuration in config file: /etc/vmware/nsx-mpa/mpaconfig.json. Ensure MPA is registered with NSX Manager. Retrying in 60 secs` — never registered
7	esxi02 `.decom_state` removal	`rm /etc/vmware/nsx-mpa/.decom_state` then `/etc/init.d/nsx-mpa restart`	Init script **re-created** the marker because `appliance-info.xml` is parseable. Output: "*Marking legacy mpa as decommissioned*" / "*Not starting decommisioned NSX-Management-Plane-Agent*"
8	Force re-register	`nsxcli -c "join management-plane 192.168.1.71 username admin thumbprint <thumb> password '<pw>'"` on esxi01	Populated `mpaconfig.json` correctly **but** created a NEW orphan fabric_node UUID `d706e294-48f9-11f1-8f79-27ba63ba6fda`, leaving the existing TN UUID `79c32f53-...` unchanged and broken. NSX manager now tracked **two** fabric nodes for esxi01 — one orphan, one stuck. Reversed via `nsxcli -c "detach management-plane ..."`.
9	Force-delete stuck TN	`DELETE /api/v1/transport-nodes/<id>?force=true&unprepare_host=false`	Blocked by error **9411**: "*Cannot delete a transport node which is part of TNP applied compute collection.*" This is a hard NSX 9 architectural rule; force flag does not override it.
10	TN action `redeploy`	`POST /api/v1/transport-nodes/<id>?action=redeploy`	HTTP 404 with "*requested object: EdgeTransportNode\ could not be found.*" — the action exists only for Edge nodes, not host TNs.

D.5 Authoritative documentation reviewed (none matched the exact failure mode)

Source	Procedure offered	Why it didn't apply to D's state
KB 409162 (Failed to send HostConfig RPC to MPA)	`/etc/init.d/nsx-opsagent restart` then `/etc/init.d/nsx-proxy restart`	KB precondition: nsx-proxy connects to NSX CCP successfully. Our hosts could not connect at all (empty mpaconfig.json).
KB 424095 (MPA DOWN after upgrade with corrupted FQDN entries)	Edit corrupted FQDNs in `appliance-info.xml` in place, restart nsx-proxy	Our `appliance-info.xml` was newly-written by restore, not corrupted.

KB 329221 (Host Disconnected, NSX 4.10/4.11)	`nestdb-cli flush CrlCertificatesCacheMsg` + restart proxy + push host-certificate via nsxcli	Cert/CRL revocation scenario. Not our state. NSX 9 not addressed.
Broadcom TechDocs (`nsxt-dc/3-2/.../restore-a-backup`)	Quote: "*If there are fabric nodes that did not discover the new manager node, you are provided a list of them … log in to a node and run a script.*"	The exact script name and path are not specified anywhere we could find in the doc set. The wizard hint never materialized into a documented procedure.
Local docs (Addendum C.9 here, `VCF9-Host-vCenter-Rebuild-Plan.md`)	C.9 covers the restore itself. Rebuild plan covers fresh redeploy.	Neither covers the post-restore + cleaned-host = empty-mpaconfig combination.

D.6 Decision criteria — abandon restored state and proceed to fresh redeploy

Proceed to fresh redeploy when **all** of the following are true:

1. Backup restore itself completed `SUCCESS` (per `GET /api/v1/cluster/restore/status`)
2. A subset of hosts remain in `pending/failed` after restore
3. Affected hosts have empty `mpaconfig.json` (verifiable: `cat /etc/vmware/nsx-mpa/mpaconfig.json` → empty AccountName / SharedSecret / RmqBrokerCluster)
4. Per-host API repair actions (1-5 above) return HTTP 200 with no host-side effect — confirmed by polling `GET /api/v1/transport-nodes/<id>/state`
5. `nsxcli -c "join management-plane ..."` creates an orphan fabric_node rather than rebinding to the existing TN
6. Stuck TN cannot be DELETEd because the TNC binding is applied (error 9411)
7. The restored NSX has minimal functional content: `count=0` for `/policy/api/v1/infra/tier-0s` , `tier-1s` , `segments` . (If the manager has live T0/T1/segments/firewall rules, the redeploy cost is higher and should be weighed against opening a Broadcom support ticket instead.)

D.7 Inventory check before redeploy — what was in the restored manager

Verified 2026-05-06 morning before deciding redeploy:

Object	Count
Tier-0 routers	0
Tier-1 routers	0
Segments	0
Distributed firewall security policies	3 (all default empty: `Default Layer2 Section` , `Default Malicious IP Block Rules` , `Default Layer3 Section`)

Transport Zones	4 (2 default protected + 2 user: `vcenter-cl01-tz-overlay01`, `vcenter-cl01-tz-vlan01`)
Uplink Profile	1 (`vcenter-cl01-up-host01`)
TEP IP Pool	1 (`vcenter-cl01-tep-pool-01`)
Transport Node Profile	1 (`vcenter-cl01-tnp-01`)
Compute Manager	1 (vCenter, REGISTERED + UP)
Transport Node Collection (TNC) binding	1 (`vcenter-cl01-tnc` → `vcenter-cl01-tnp-01`)

Conclusion: The restored manager held only host-prep templates — no live overlay tenant data. The fresh-redeploy procedure recreates these templates in ~15 minutes (P4.9 of the rebuild plan). Choosing redeploy over a Broadcom support engagement was the correct cost trade-off.

D.8 Adapted plan inputs (verified before execution)

Decision	Verified answer	Verification method
vSAN MM mode for ordinary `del nsx force` cleanup (no reboot, no disk wipe)	**Ensure Accessibility** per L-NSX-8 of `VCF9-Host-vCenter-Rebuild-Plan.md`	`feedback_vsan_mm_full_data_migration.md` literal scope is *"for any vSAN host that's going to be wiped and rebuilt"* — `del nsx force` is not a wipe/rebuild, so the FDM-only rule does not apply.
vSAN MM mode if `esxi03` hits `Status(bad0004) = Busy` and requires fresh ESXi reinstall	**Full Data Migration** per memory `feedback_vsan_mm_full_data_migration.md`	The reinstall path IS a wipe/rebuild — FDM is mandatory to avoid the 2026-05-01 vCenter EXT4 journal-abort scenario.
`esxi03` DVS-busy risk	**HIGH** — verified via direct host inspection	`vmk10/vmk11/vmk50` active; DVS props `com.vmware.nsx.vdl2.enabled=true`, `com.vmware.common.portset.nsx=true`; `dvsdata.db` 131 KB with NSX content. Pre-staged ESXi 9.0.1.0.24957456 ISO at `I:\VCF-Depot\PROD\COMP\ESXI\` for reinstall path.
`esxi04` cleanup needed?	**No** — verified	No `vmk10/vmk11/vmk50`; no NSX DVS flags; only NSX VIBs present (no state). `del nsx force` would be a no-op.
`esxi04` as new NSX Manager target	**Yes** — verified empty (only vCLS)	`vim-cmd vmsvc/getallvms` on esxi04 lists only `vCLS-fcd84d56-*`. Aligns with L-NSX-10 (NSX Manager on dedicated host).
NSX OVA available	✅ `I:\VCF-Depot\PROD\COMP\NSX_T_MANAGER\nsx-unified-appliance-9.0.1.0.24952114.ova` (11.2 GB)	Same build as 2026-04-25 deploy (Addendum B).

ovftool path	`C:\Program Files (x86)\VMware\VMware Workstation\OVFTool\ovftool.exe` (version 5.0.0 build 24927197)	Searched non-standard locations — was bundled with VMware Workstation install, not as a standalone install.

D.9 Execution log (2026-05-06, UTC)

Time	Phase	Event
12:16	PF	Pre-flight: vSAN 123 healthy, resync 0, 4/4 cluster members, MM OFF, ESXi ISO staged, depot HTTPS not running (informational only), esxi03 net plan captured
12:23	A-1..A-4	Found old `nsx-manager` VMID 18 on esxi01 (already POWERED_OFF the previous night per operator). `vim-cmd vmsvc/destroy 18` → exit 0. esxi01 now contains only vCLS. vSAN resync remains 0.
12:38	B-1 (first attempt)	ovftool deploy → failed: "*Duplicate name 'nsx-manager' on target. Use --overwrite option to delete it.*" The `vim-cmd destroy` left an orphan vCenter inventory entry (`vm-5015`).
12:43	B-1 (orphan cleanup, P4.7 second snippet)	PowerCLI: `Get-VM nsx-manager | Where ConnectionState='orphaned' | Remove-VM` . Inventory cleaned.
12:56	B-1 (retry)	ovftool deploy launched in background (PID 4316). Target: `esxi04.lab.local` . Datastore: `vcenter-cl01-ds-vsan01` . Network: `pg-vm-mgmt` . OVA: `nsx-unified-appliance-9.0.1.0.24952114.ova` . Form factor: small. `--powerOffTarget` set so resources can be customized before first boot.
12:57	B-2	While ovftool was uploading (VM stub registered with default 4 vCPU / 16 GB), edited `nsx-manager.vmx` directly to `numvcpus = "6"` and `memSize = "49152"` . `vim-cmd vmsvc/reload 3` to refresh hostd cache. Verified live config: `numCPU = 6, memoryMB = 49152` . (`--powerOffTarget` ensured ovftool did not overwrite the VMX on completion.)
13:43	B-1 (completion)	ovftool deploy completed successfully. ~47 minutes wall clock for 11 GB OVA → ~30 GB on vSAN. Operator-observed progress: 27% → 56% → 61% → 78% → 100%.
13:43	B-2 (verify)	Re-read VMX post-deploy: `numvcpus = "6"` , `memSize = "49152"` (operator edits intact — ovftool did not overwrite because of `--powerOffTarget`).
13:43	B-3	`vim-cmd vmsvc/power.on 3` → **Powered on**. (Initial attempt during deploy failed with `vim.fault.FileLocked` because ovftool still held the vmdk; retried after deploy completion.)
13:45+	B-3	Awaiting service init (~20 min per Phase 4 timeline). Watcher armed on `https://192.168.1.71/` for HTTP 200/302/401/403.

Sections D.10-D.13 will be filled in as Phases B-5, C, D, E complete.

D.10 Lessons Learned added (extends C.10)

- **L-29** — A SUCCESSFUL backup restore can land you in a NON-RECOVERABLE state if hosts had their NSX state cleaned during the same recovery window. The cluster TN profile binding restored from backup blocks per-host re-prep; the missing `mpaconfig.json` blocks per-host registration; neither end can resolve without API capabilities NSX 9 does not expose. The only documented path forward is the same fresh-redeploy procedure used when no backup exists. **Implication for backup strategy:** before restoring NSX, evaluate whether any host had `del nsx force` run during the recovery window. If yes, weigh restore vs. fresh redeploy with full information — restore is not always the faster path.
- **L-30** — `nsxcli -c "join management-plane ..."` should NEVER be run as a "fix" for a TN stuck in pending/failed state. It creates a NEW fabric_node UUID, leaving the existing TN orphan-bound to a vanished host identity. Only run `join management-plane` on a host that has no existing TN object in the manager (e.g., during fresh prep). Always reach for fresh redeploy if no API-level repair works.
- **L-31** — Before attempting an NSX backup restore, take inventory of what's actually in the manager: `count` of T0/T1/segments/firewall-policies. If the manager's contents are templates only, fresh redeploy may be cheaper and lower-risk than a restore that doesn't fully recover. Verify with: `curl /policy/api/v1/infra/tier-0s`, `tier-1s`, `segments`.
- **L-32** — `/etc/vmware/nsx-mpa/.decom_state` is recreated by the init script when `/etc/vmware/nsx/appliance-info.xml` is parseable (per script lines 154/160/166 of `/etc/init.d/nsx-mpa`). The presence of valid `appliance-info.xml` does NOT mean MPA can start — the init script has a quirky "legacy MPA" detection that decommissions hosts that look prepped but were cleaned. The fix is fresh redeploy, not file manipulation.
- **L-33** — When a TN is part of a TNP-applied compute collection (TransportNodeCollection binding), individual TN DELETE returns error 9411. The architectural rule is: detach the TNC binding first OR redeploy. There is no per-host force-delete that respects this rule.
- **L-34** — `vim-cmd vmsvc/destroy` on the ESXi host removes the VM from local hostd inventory but leaves a vCenter-level `orphaned` VM record. **Always follow with PowerCLI `Get-VM <name> \| Where ConnectionState='orphaned' \| Remove-VM` before retrying ovftool deploy with the same VM name.** Otherwise ovftool errors with: "*Duplicate name '' on target. Use --overwrite option to delete it.*" Adding `--overwrite` is the alternative shortcut but it deletes by name and will hit a different orphan error if the target vCenter holds a stale record.
- **L-35** — To customize NSX Manager resources (6 vCPU / 48 GB) before first boot, edit `nsx-manager.vmx` directly on the ESXi host while the VM is registered but powered off, then run `vim-cmd vmsvc/reload <vmid>` to refresh the hostd cache. The `--powerOffTarget` flag on ovftool is essential — without it, the VM is powered on at deploy completion with the small-form-factor defaults (4/16) and customization requires a power-cycle. Also: do NOT attempt power-on while ovftool is still uploading — the vmdk is locked (`vim.fault.FileLocked`); wait for ovftool to fully complete first.
- **L-36** — The hourly NSX backup configuration (Backupplan01) survived through the rebuild and continues writing to `I:\VCF-Depot\backups\nsx-manager\`. After the new NSX Manager is up, re-point its

backup destination to the same SFTP server (workstation `192.168.1.160:22`) using the same passphrase `Backupplan01!`. Existing TARs are still encrypted with that key and remain available for a future restore if needed.

D.11 Cross-references

- **Addendum C** (this document) — the 2026-05-05 power-outage / btrfs / restore sequence whose successful restore created the post-restore stuck-state described here
- **`VCF9-Host-vCenter-Rebuild-Plan.md`** Phase 4 — the verified fresh-redeploy procedure being followed
- **`feedback_vsan_mm_full_data_migration.md`** — the FDM-only rule for wipe/rebuild scenarios; explicitly out-of-scope for `del nsx force` cleanup
- **`feedback_no_assumptions.md`** — the "verify before claiming" rule that drove the 10-mitigation empirical exhaustion before declaring redeploy
- **`feedback_kb_prescribed_path.md`** — the rule that drove the 5-KB review before declaring no documented remedy applies
- **`reference_lcm_is_fleet_in_vcf9.md`** — clarifies why a `fleet` VM and `lcm.vmx` file co-exist (renamed appliance, single role in 9.x)
