HEALTH CHECK HANDBOOK

VIRTUAL CONTROL

VMWARE CLOUD FOUNDATION SOLUTIONS

Fleet / SDDC Manager Health Check Handbook

SDDC Manager and Fleet Management health checks covering lifecycle operations, bundle management, credential rotation, and workload domain validation.

SDDC MANAGER | LIFECYCLE | BUNDLES | CREDENTIALS | DOMAINS

VCF 9.0 VMware Cloud Foundation

PROPRIETARY & CONFIDENTIAL

Fleet / SDDC Manager Health Check Handbook

Comprehensive Health Verification for Fleet & SDDC Manager in VCF 9

Author: Virtual Control LLC **Date:** March 2026 **Version:** 1.0 **Classification:** Internal Use **Platform:** VMware Cloud Foundation 9.0 / SDDC Manager 5.2.x

Table of Contents

1. Overview & Purpose

This handbook provides a **complete health check procedure** for the SDDC Manager (Fleet Manager) in a VCF 9.0 environment. SDDC Manager is the **central orchestration and lifecycle management** component of VCF. Its health directly affects your ability to:

- Deploy and manage workload domains
- Perform lifecycle updates (patches, upgrades)
- Commission/decommission ESXi hosts
- Manage certificates across all VCF components
- Monitor component inventory and compliance

When to Run This Health Check

Trigger	Priority
Before any LCM operation (upgrade/patch)	**Critical**
After any LCM operation	**Critical**
Weekly routine maintenance	**Recommended**
After infrastructure changes	**Recommended**
When tasks are failing/stuck	**Troubleshooting**

Environment Variables: Throughout this document:
`$SDDC` = SDDC Manager FQDN (e.g., sddc-manager.lab.local)
`$SDDC_USER` = admin@local (or SSO admin)
`$SDDC_PASS` = SDDC Manager password
`$TOKEN` = Bearer token acquired via API

2. Prerequisites

Required Access

Access Type	Target	Credentials
HTTPS (443)	SDDC Manager	admin@local or administrator@vsphere.local
SSH (22)	SDDC Manager appliance	vcf / password
SSH (22)	SDDC Manager appliance	root / password
REST API	SDDC Manager :443	Bearer token

Token Acquisition

```
export SDDC="sddc-manager.lab.local"
export SDDC_USER="admin@local"
export SDDC_PASS="YourPassword123!"

# Acquire access token
TOKEN=$(curl -sk -X POST "https://$SDDC/v1/tokens" \
  -H "Content-Type: application/json" \
  -d "{\"username\":\"$SDDC_USER\",\"password\":\"$SDDC_PASS\"}" | jq -r
'.accessToken')

echo "Token: ${TOKEN:0:20}..."

# Convenience function
sddc_api() {
  curl -sk -H "Authorization: Bearer $TOKEN" \
    -H "Content-Type: application/json" \
    "https://$SDDC$1" 2>/dev/null
}
```

Token Expiry: SDDC Manager tokens expire after 30 minutes. Re-acquire if you get 401 responses.

3. Quick Reference — All Checks Summary

#	Check	Method	PASS	WARN	FAIL
4.1	Service Status	SSH/systemctl	All services active	Non-critical stopped	Critical service down
5.1	PostgreSQL	SSH	Running, accepting connections	High connection count	Not running
5.2	DB Size	SQL	< 5 GB	5-10 GB	> 10 GB
6.1	Component Inventory	API	All components `ACTIVE`	Any `WARNING`	Any `ERROR`
7.1	Current Version	API	Latest VCF version	1 version behind	2+ versions behind
8	Bundles	API	Bundles available	Download in progress	Download failed
9	DNS/NTP	API/CLI	All resolving, time sync	Intermittent DNS	DNS failure or NTP > 5s
10	Certificates	API	All > 30 days	Any 7-30 days	Any < 7 days or expired
11	Tasks	API	No failed tasks	Old failed tasks	Recent critical failures
12	Hosts	API	All `ASSIGNED` or `UNASSIGNED_USEABLE`	Any `COMMISSIONING`	Any `ERROR`
13	Domains	API	All `ACTIVE`	Any `ACTIVATING`	Any `ERROR`
14	Backup	API/SSH	Configured, recent success	> 24h since last	Not configured
15	API Health	API	Token acquired, < 2s response	2-5s response	API unresponsive
16	Resources	SSH	CPU < 70%, Mem < 80%, Disk < 70%	CPU/Mem/Disk warn	Any critical

4. Service Status

4.1 All Services via systemctl

What: Verify all SDDC Manager services are running.

```
ssh vcf@$SDDC
# Then switch to root or use sudo:
sudo systemctl list-units --type=service --state=running | grep -E
"vcf|sddc|operationsd|commonsvcs|domainmanager|lcm"
```

Expected Output:

```
commonsvcs.service                loaded active running   VCF Common Services
domainmanager.service             loaded active running   VCF Domain Manager
lcm.service                       loaded active running   VCF Lifecycle Manager
operationsd.service               loaded active running   VCF Operations
sddc-manager-ui-app.service       loaded active running   SDDC Manager UI
sddc-support.service              loaded active running   VCF Support Service
```

4.2 VCF Service Status Script

```
# Comprehensive service check
for SVC in commonsvcs domainmanager lcm operationsd sddc-manager-ui-app sddc-
support; do
  STATUS=$(systemctl is-active $SVC 2>/dev/null)
  if [ "$STATUS" = "active" ]; then
    echo "[PASS] $SVC: $STATUS"
  else
    echo "[FAIL] $SVC: $STATUS"
  fi
done
```

4.3 Critical vs Non-Critical Services

Service	Critical	Function	Impact if Down

`commonsvcs`	**Yes**	Shared services (auth, config)	All SDDC Manager functions fail
`domainmanager`	**Yes**	Domain operations	Cannot create/modify workload domains
`lcm`	**Yes**	Lifecycle management	Cannot patch/upgrade
`operationsd`	**Yes**	Task orchestration	No task execution
`sddc-manager-ui-app`	No	Web UI	UI unavailable (API still works)
`sddc-support`	No	Support bundle	Cannot generate support bundles
`postgresql`	**Yes**	Database	Complete SDDC Manager failure
`nginx`	**Yes**	Reverse proxy / API gateway	API and UI unreachable

Result	Criteria	Indicator
PASS	All services `active`	Healthy
WARN	Non-critical service stopped	Limited functionality
FAIL	Any critical service not `active`	SDDC Manager degraded/down

Remediation:
1. Restart specific service: `sudo systemctl restart <service-name>`
2. Restart all VCF services: `sudo systemctl restart commonsvcs domainmanager lcm operationsd`
3. Check logs: `journalctl -u <service-name> --since "1 hour ago"`
4. Full service restart: `sudo /opt/vmware/vcf/operationsmanager/scripts/cli/vcf-service-status.sh`

5. Database Health

5.1 PostgreSQL Status

```
ssh root@$SDDC
systemctl status postgresql
```

Expected Output:

```
● postgresql.service - PostgreSQL database server
     Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled)
     Active: active (running) since Mon 2026-03-20 08:00:00 UTC; 6 days ago
   Main PID: 1234 (postgres)
     Memory: 256.0M
```

Connection Test

```
sudo -u postgres psql -c "SELECT version();"
```

Expected Output:

```
                                 version
-------------------------------------------------------------------------
 PostgreSQL 14.x on x86_64-pc-linux-gnu, compiled by gcc ...
(1 row)
```

5.2 Database Size & Connections

```
# Database size
sudo -u postgres psql -c "
SELECT datname, pg_size_pretty(pg_database_size(datname)) as size
FROM pg_database WHERE datname NOT IN ('template0','template1','postgres')
ORDER BY pg_database_size(datname) DESC;"
```

Expected Output:

```
   datname     |  size
---------------+---------
 sddc_manager_db | 2.1 GB
 lcm_db          | 512 MB
 operations_db   | 256 MB
```

```
# Active connections
sudo -u postgres psql -c "
SELECT datname, count(*) as connections
FROM pg_stat_activity
GROUP BY datname ORDER BY connections DESC;"
```

Result	Criteria	Indicator
PASS	DB size < 5 GB, connections < 100	Healthy
WARN	DB size 5-10 GB or connections 100-200	Monitor
FAIL	DB size > 10 GB or connections > 200	Investigate bloat

5.3 Vacuum Status

```
sudo -u postgres psql -d sddc_manager_db -c "
SELECT schemaname, relname, n_dead_tup, last_autovacuum
FROM pg_stat_user_tables
WHERE n_dead_tup > 10000
ORDER BY n_dead_tup DESC LIMIT 10;"
```

Remediation:
1. Manual vacuum: `sudo -u postgres vacuumdb --all --analyze`
2. If bloated: `sudo -u postgres vacuumdb --full --all` (requires downtime)
3. Check autovacuum: `SHOW autovacuum;` should return `on`

5.4 Database Backup

```
# Check if database backups exist
ls -la /opt/vmware/vcf/sddc-manager/backup/ 2>/dev/null
ls -la /nfs-mount/vcf-backups/ 2>/dev/null
```

6. Component Inventory

6.1 System Components

What: Verify all VCF-managed components are in a healthy state.

```
# List all VCF components
sddc_api "/v1/system" | jq .
```

Get all vCenters

```
sddc_api "/v1/vcenter" | jq '.elements[] | {
  id: .id,
  fqdn: .fqdn,
  version: .version,
  status: .status
}'
```

Get all NSX Managers

```
sddc_api "/v1/nsxt-clusters" | jq '.elements[] | {
  id: .id,
  vipFqdn: .vipFqdn,
  version: .version
}'
```

6.2 Workload Domains

```
sddc_api "/v1/domains" | jq '.elements[] | {
  id: .id,
  name: .name,
  type: .type,
```

```
  status: .status
}'
```

Expected Output:

```
{
  "id": "abc123...",
  "name": "MGMT",
  "type": "MANAGEMENT",
  "status": "ACTIVE"
}
{
  "id": "def456...",
  "name": "WLD-01",
  "type": "VI",
  "status": "ACTIVE"
}
```

6.3 Host Inventory

```
sddc_api "/v1/hosts" | jq '.elements[] | {
  id: .id,
  fqdn: .fqdn,
  status: .status,
  domain: .domain.name
}'
```

Status	Meaning
`ASSIGNED`	Host is part of a workload domain
`UNASSIGNED_USEABLE`	Commissioned, available for assignment
`COMMISSIONING`	Being added to inventory
`DECOMMISSIONING`	Being removed
`ERROR`	Host in error state

7. Lifecycle Management (LCM)

7.1 Current Version

```
# SDDC Manager version
sddc_api "/v1/system" | jq '{
  version: .version,
  build: .build,
  fips_enabled: .fipsEnabled
}'
```

7.2 Available Updates

```
# Check available updates
sddc_api "/v1/system/prechecks" | jq .
```

Bundle Availability

```
sddc_api "/v1/bundles" | jq '.elements[] | {
  id: .id,
  bundleType: .bundleType,
  version: .version,
  status: .downloadStatus,
  components: [.components[].type]
}'
```

7.3 Upgrade Prechecks

```
# Trigger precheck
curl -sk -X POST -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  "https://$SDDC/v1/system/prechecks" \
```

```
  -d '{"bundleId":"<bundle-id>"}'

# Check precheck results
sddc_api "/v1/system/prechecks/<precheck-id>" | jq '.results[] | {
  check: .description,
  status: .status,
  severity: .severity,
  resolution: .resolution
}'
```

Result	Criteria	Indicator
PASS	All prechecks pass	Ready to upgrade
WARN	Warning-level prechecks	Review before proceeding
FAIL	Any critical precheck failure	Must resolve before upgrade

8. Bundle Management

```
# List all bundles
sddc_api "/v1/bundles" | jq '.elements[] | {
  id: .id,
  type: .bundleType,
  version: .version,
  downloadStatus: .downloadStatus,
  applicableVersions: .applicableVersions
}'
```

Download Statuses:

Status	Meaning
`SUCCESSFUL`	Bundle downloaded successfully
`IN_PROGRESS`	Currently downloading
`FAILED`	Download failed
`NOT_STARTED`	Available but not downloaded

Check Depot Connectivity

```
# Test connectivity to VMware depot (online mode)
ssh vcf@$SDDC
curl -sk https://depot.vmware.com/PROD2/vcf/manifest.json | head -5
```

Offline Depot: If using an offline depot, verify the depot server is reachable and the depot path is configured correctly in SDDC Manager → Administration → Depot Settings.

9. DNS & NTP Verification

DNS Configuration

```
# Get DNS config via API
sddc_api "/v1/system/dns-configuration" | jq .
```

Expected Output:

```
{
  "dnsServers": [
    {"ipAddress": "192.168.1.1", "isPrimary": true},
    {"ipAddress": "192.168.1.2", "isPrimary": false}
  ]
}
```

DNS Resolution Test

```
ssh vcf@$SDDC
# Test forward resolution for all VCF components
for HOST in vcenter.lab.local nsx-vip.lab.local sddc-manager.lab.local; do
  echo "$HOST: $(nslookup $HOST | grep Address | tail -1)"
done

# Test reverse resolution
for IP in 192.168.1.70 192.168.1.71 192.168.1.60; do
  echo "$IP: $(nslookup $IP | grep name | head -1)"
done
```

NTP Configuration

```
# Get NTP config
sddc_api "/v1/system/ntp-configuration" | jq .
```

```
# Check time sync on appliance
ssh vcf@$SDDC
timedatectl status
chronyc tracking
```

Result	Criteria	Indicator
PASS	DNS resolves all components, NTP synced < 1s	Healthy
WARN	Slow DNS or NTP drift 1-5s	Monitor
FAIL	DNS failure or NTP not synced	LCM operations will fail

10. Certificate Health

```
# List all certificates managed by VCF
sddc_api "/v1/certificate-authorities" | jq .

# Get certificates for a specific resource
sddc_api "/v1/domains/<domain-id>/resource-certificates" | jq '.elements[] | {
  resource: .resourceFqdn,
  type: .certificateType,
  expiresAt: .expirationDate,
  issuedBy: .issuedBy
}'
```

Check Certificate Expiry via OpenSSL

```
# Check SDDC Manager certificate
echo | openssl s_client -connect $SDDC:443 2>/dev/null | \
  openssl x509 -noout -dates -subject
```

CSR Status

```
sddc_api "/v1/certificate-authorities/csr" | jq .
```

Result	Criteria	Indicator
PASS	All certificates > 30 days from expiry	Healthy
WARN	Any certificate 7-30 days from expiry	Plan renewal
FAIL	Any certificate < 7 days or expired	Immediate action

Certificate Expiry Impact: Expired SDDC Manager certificates will prevent:
- API access (token acquisition fails)
- LCM operations
- Communication with vCenter, NSX, ESXi

Action: Use Certificate Manager or API to replace certificates before expiry.

11. Task & Workflow History

```
# List recent tasks (last 20)
sddc_api "/v1/tasks?limit=20" | jq '.elements[] | {
  id: .id,
  name: .name,
  status: .status,
  type: .type,
  creationTimestamp: .creationTimestamp,
  completionTimestamp: .completionTimestamp
}'
```

Task Statuses:

Status	Meaning
`SUCCESSFUL`	Completed successfully
`FAILED`	Failed — check subtasks for details
`IN_PROGRESS`	Currently executing
`CANCELLED`	Cancelled by user

Check for Failed Tasks

```
sddc_api "/v1/tasks?status=FAILED&limit=10" | jq '.elements[] | {
  id: .id,
  name: .name,
  creationTimestamp: .creationTimestamp,
  errorMessage: .errors
}'
```

Retry a Failed Task

```
curl -sk -X PATCH -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
```

```
  "https://$SDDC/v1/tasks/<task-id>" \
  -d '{"status":"IN_PROGRESS"}'
```

Result	Criteria	Indicator
PASS	No failed tasks in last 7 days	Clean
WARN	Older failed tasks present	Review and clean up
FAIL	Recent critical task failures	Investigate immediately

12. Host Commissioning Status

```
# All hosts with their status
sddc_api "/v1/hosts" | jq '.elements[] | {
  fqdn: .fqdn,
  status: .status,
  domain: .domain.name,
  clusterId: .cluster.id
}'

# Count by status
sddc_api "/v1/hosts" | jq '[.elements[] | .status] | group_by(.) | map({status: .
[0], count: length})'
```

Remediation for stuck COMMISSIONING:

1. Check task status: `sddc_api "/v1/tasks?type=HOST_COMMISSION"`
2. Verify host SSH access and credentials
3. Check DNS resolution for the host
4. Retry: Cancel and recommission

13. Workload Domain Health

```
# List all domains with health
sddc_api "/v1/domains" | jq '.elements[] | {
  name: .name,
  type: .type,
  status: .status,
  clusters: [.clusters[].name]
}'
```

Cluster Health per Domain

```
# List clusters
sddc_api "/v1/clusters" | jq '.elements[] | {
  name: .name,
  status: .status,
  domain: .domainName,
  hostCount: (.hosts | length),
  primaryDatastore: .primaryDatastoreType
}'
```

14. Backup & Restore

```
# Get backup configuration
sddc_api "/v1/system/backup-configuration" | jq .
```

Expected Output:


```
{
  "backupEnabled": true,
  "backupSchedule": {
    "frequency": "DAILY",
    "daysOfWeek": null,
    "hourOfDay": 2,
    "minuteOfHour": 0
  },
  "server": {
    "protocol": "SFTP",
    "host": "backup-server.lab.local",
    "port": 22,
    "directory": "/backups/sddc/"
  },
  "encryption": {
    "passphrase": "***"
  }
}
```


Check Backup History

```
sddc_api "/v1/system/backup-configuration/backups" | jq '.elements[0:3]'
```

Result	Criteria	Indicator
PASS	Backup configured, last success < 24h	Protected
WARN	Backup configured, last success > 24h	Check schedule
FAIL	No backup configured or all recent failed	Critical risk

15. API Health Verification

Token Acquisition Test

```
# Time the token acquisition
time curl -sk -X POST "https://$SDDC/v1/tokens" \
  -H "Content-Type: application/json" \
  -d "{\"username\":\"$SDDC_USER\",\"password\":\"$SDDC_PASS\"}" | jq -r
'.accessToken' > /dev/null
```

Result	Criteria	Indicator
PASS	Token acquired in < 2 seconds	API responsive
WARN	Token acquired in 2-5 seconds	API slow
FAIL	Token acquisition fails or > 5 seconds	API issue

Endpoint Health Check Script

```
ENDPOINTS="/v1/system /v1/domains /v1/hosts /v1/clusters /v1/bundles"
for EP in $ENDPOINTS; do
  START=$(date +%s%N)
  HTTP_CODE=$(curl -sk -o /dev/null -w "%{http_code}" \
    -H "Authorization: Bearer $TOKEN" \
    "https://$SDDC$EP")
  END=$(date +%s%N)
  DURATION=$(( (END - START) / 1000000 ))
  echo "$EP: HTTP $HTTP_CODE (${DURATION}ms)"
done
```

16. Resource Utilization

```
ssh root@$SDDC

# CPU
uptime
top -b -n 1 | head -5

# Memory
free -m

# Disk
df -h
```

Critical Partitions

Partition	PASS	WARN	FAIL
`/` (root)	< 70%	70-85%	> 85%
`/var/log`	< 70%	70-85%	> 85%
`/opt`	< 70%	70-85%	> 85%
DB partition	< 70%	70-85%	> 85%

Remediation:
1. Clean old logs: `find /var/log -name "*.gz" -mtime +30 -delete`
2. Clean old task data: `sddc_api "/v1/tasks?status=SUCCESSFUL" | jq '.elements | length'`
3. Check large files: `du -sh /var/log/* | sort -rh | head -10`

17. Port Reference Table

Inbound Ports

Source	Port	Protocol	Purpose
Admin Browser	443	TCP	Web UI / REST API
Admin	22	TCP	SSH
vCenter	443	TCP	Inventory sync
NSX Manager	443	TCP	Component registration
ESXi Hosts	443	TCP	Host commissioning

Outbound Ports

Destination	Port	Protocol	Purpose
vCenter	443	TCP	vCenter management
NSX Manager	443	TCP	NSX lifecycle
ESXi Hosts	443	TCP	Host preparation
ESXi Hosts	22	TCP	Host configuration (SSH)
DNS Server	53	TCP/UDP	Name resolution
NTP Server	123	UDP	Time synchronization
SFTP Backup	22	TCP	Backup transfer
VMware Depot	443	TCP	Bundle downloads (online)
Offline Depot	443	TCP	Bundle downloads (offline)
PostgreSQL (local)	5432	TCP	Database (localhost)

18. Common Issues & Remediation

18.1 Service Failures

Symptom	Likely Cause	Resolution
UI inaccessible	nginx or UI service down	`systemctl restart nginx sddc-manager-ui-app`
API returns 503	Backend services down	`systemctl restart commonsvcs domainmanager lcm operationsd`
Slow API responses	Database issue or resource exhaustion	Check DB connections and disk space

18.2 Database Issues

Symptom	Likely Cause	Resolution
Connection refused	PostgreSQL not running	`systemctl restart postgresql`
Slow queries	Table bloat / missing vacuum	`vacuumdb --all --analyze`
Disk full (DB)	Large transaction logs	Clean WAL files, increase disk

18.3 LCM Failures

Symptom	Likely Cause	Resolution
Precheck fails	Component version mismatch	Review precheck report, resolve each item
Bundle download fails	Network/proxy issue	Check depot connectivity, proxy settings
Upgrade stuck	Task hung	Check subtasks, cancel and retry

18.4 Task Stuck in IN_PROGRESS

```
# Find stuck tasks (running > 2 hours)
sddc_api "/v1/tasks?status=IN_PROGRESS" | jq '.elements[] | {
  id: .id,
  name: .name,
```

```
  creationTimestamp: .creationTimestamp
}'
```

Remediation:
1. Check subtask status for the stuck task
2. If safe to cancel: `PATCH /v1/tasks/<id>` with `{"status":"CANCELLED"}`
3. Restart VCF services: `systemctl restart commonsvcs domainmanager lcm operationsd`
4. Check for locks: `sddc_api "/v1/tasks?type=LOCK" | jq .`

18.5 Certificate Problems

Symptom	Likely Cause	Resolution
401 on all API calls	SDDC Manager cert expired	Replace via Certificate Manager
LCM fails with cert error	Component cert expired	Replace component certificates first
Trust verification fails	CA not in trust store	Add CA to SDDC Manager trust store

19. CLI Quick Reference Card

Service Management

Command	Purpose
`systemctl status commonsvcs`	Common services status
`systemctl status domainmanager`	Domain manager status
`systemctl status lcm`	Lifecycle manager status
`systemctl status operationsd`	Operations daemon status
`systemctl status postgresql`	Database status
`systemctl status nginx`	Web server / proxy status
`systemctl restart <service>`	Restart a service

Database Commands

Command	Purpose
`sudo -u postgres psql`	Enter PostgreSQL shell
`sudo -u postgres psql -c "SELECT version();"`	Check DB version
`sudo -u postgres psql -l`	List databases
`sudo -u postgres vacuumdb --all --analyze`	Vacuum all databases

System Commands

Command	Purpose
`df -h`	Disk usage
`free -m`	Memory usage
`uptime`	Load average
`timedatectl`	Time sync status
`chronyc tracking`	NTP tracking details
`cat /etc/vmware/vcf/sddc-manager-version`	SDDC Manager version

`nslookup <hostname>`	DNS test
`openssl s_client -connect <host>:443`	Certificate check

Log Files

Log File	Purpose
`/var/log/vmware/vcf/commonsvcs/commonsvcs.log`	Common services
`/var/log/vmware/vcf/domainmanager/domainmanager.log`	Domain manager
`/var/log/vmware/vcf/lcm/lcm.log`	Lifecycle manager
`/var/log/vmware/vcf/operationsmanager/operationsmanager.log`	Operations
`/var/log/vmware/vcf/sddc-support/sddc-support.log`	Support service
`/var/log/nginx/access.log`	Nginx access log
`/var/log/nginx/error.log`	Nginx error log

20. API Quick Reference

Authentication

```
# Acquire token
curl -sk -X POST "https://$SDDC/v1/tokens" \
  -H "Content-Type: application/json" \
  -d '{"username":"admin@local","password":"password"}'

# Use token
curl -sk -H "Authorization: Bearer $TOKEN" "https://$SDDC/v1/..."
```

Key Endpoints

Endpoint	Method	Purpose
`/v1/tokens`	POST	Acquire access token
`/v1/system`	GET	System info / version
`/v1/system/dns-configuration`	GET	DNS config
`/v1/system/ntp-configuration`	GET	NTP config
`/v1/system/backup-configuration`	GET	Backup config
`/v1/system/prechecks`	POST	Trigger upgrade precheck
`/v1/domains`	GET	List workload domains
`/v1/domains/<id>`	GET	Domain details
`/v1/clusters`	GET	List clusters
`/v1/hosts`	GET	List hosts
`/v1/hosts/<id>`	GET	Host details
`/v1/vcenter`	GET	List vCenters
`/v1/nsxt-clusters`	GET	List NSX clusters
`/v1/bundles`	GET	List bundles
`/v1/tasks`	GET	List tasks

`/v1/tasks/<id>`	GET	Task details
`/v1/tasks/<id>`	PATCH	Retry/cancel task
`/v1/certificate-authorities`	GET	CA configuration
`/v1/sddc-managers`	GET	SDDC Manager info

Common Query Parameters

Parameter	Example	Purpose
`limit`	`?limit=50`	Results per page
`offset`	`?offset=0`	Pagination offset
`status`	`?status=FAILED`	Filter by status
`sort`	`?sort=creationTimestamp,DESC`	Sort results
`type`	`?type=HOST_COMMISSION`	Filter by task type