# **SD-WAN Health Checks Playbook Documentation**

## **Overview**

The SD-WAN health checks playbook is an Ansible automation script designed to perform comprehensive health monitoring of Cisco SD-WAN environments. This playbook connects to the vManage controller (version 20.15) and executes various health checks to assess the overall system status, device connectivity, and network performance.

## **Playbook Information**

* **Filename:** health\_checks.yml
* **Use Case:** #7 - health\_checks - Run health checks
* **Target vManage Version:** 20.15
* **Output Location:** generated/ folder (relative to playbook directory)

## **Prerequisites**

### **Environment Variables Required**

* VMANAGE\_HOST - vManage controller hostname/IP
* VMANAGE\_USERNAME - Authentication username
* VMANAGE\_PASSWORD - Authentication password
* VMANAGE\_PORT - HTTPS port (defaults to 443)

### **System Requirements**

* Ansible installed and configured
* Network connectivity to vManage controller
* Valid credentials with API access permissions

## **Detailed Task Analysis**

### **Task 1: Environment Variable Validation**

**Purpose:** Ensures all required credentials are available before proceeding

**What it does:**

* Validates that vmanage\_host, vmanage\_username, vmanage\_password, and vmanage\_port are set
* Fails immediately if any critical environment variables are missing
* Prevents failed health check attempts due to missing credentials

### **Task 2: Directory Creation**

**Purpose:** Creates the output directory structure

**Generated folder:**

* generated/ - All health check results are saved here

### **Task 3: API Connectivity Test**

**Purpose:** Verifies the vManage controller is accessible

**API Endpoint:** /dataservice/system/device/controllers **What it checks:**

* Basic API connectivity
* Authentication validity
* Network reachability
* Response time

### **Task 4: System Status Check**

**Purpose:** Retrieves overall system health status

**API Endpoint:** /dataservice/system/status **What it provides:**

* System operational status
* Service availability
* Resource utilization indicators

### **Task 5: Device Inventory Check**

**Purpose:** Gets complete device inventory and reachability status

**API Endpoint:** /dataservice/device **What it provides:**

* Complete device list
* Device reachability status
* Device types and roles
* Online/offline device counts

### **Task 6: Control Connections Check**

**Purpose:** Monitors control plane connectivity

**API Endpoint:** /dataservice/device/control/connections **What it provides:**

* Control connection status between devices
* Connection state information
* Control plane health indicators

### **Task 7: BFD Sessions Check**

**Purpose:** Monitors Bidirectional Forwarding Detection sessions

**API Endpoint:** /dataservice/device/bfd/sessions **What it provides:**

* BFD session status
* Session state information
* Link health indicators

### **Task 8: OMP Peers Check**

**Purpose:** Monitors Overlay Management Protocol peer relationships

**API Endpoint:** /dataservice/device/omp/peers **What it provides:**

* OMP peer status
* Peer relationship health
* Routing protocol connectivity

### **Task 9: Active Alarms Check**

**Purpose:** Retrieves current system alarms

**API Endpoint:** /dataservice/alarms **What it provides:**

* Active alarm count
* Alarm severity levels
* System issues requiring attention

### **Task 10: Certificate Status Check**

**Purpose:** Monitors certificate health and validity

**API Endpoint:** /dataservice/certificate/stats **What it provides:**

* Certificate validation status
* Certificate expiration information
* PKI health indicators

### **Task 11: Cluster Status Check**

**Purpose:** Monitors cluster management health (for clustered vManage deployments)

**API Endpoint:** /dataservice/clusterManagement/health/status **What it provides:**

* Cluster node status
* Cluster synchronization health
* High availability status

### **Task 12: Health Statistics Calculation**

**Purpose:** Processes collected data into summary statistics

**Generated statistics:**

* Overall connectivity status (PASS/FAIL)
* System status (PASS/FAIL)
* Total device count
* Online device count
* Control connections count
* BFD sessions count
* OMP peers count
* Active alarms count
* Certificate status (PASS/FAIL)
* Cluster status (PASS/FAIL)

### **Tasks 13-21: Data Persistence**

**Purpose:** Saves all collected health check data to JSON files

**Generated files:**

* connectivity\_test.json - API connectivity test results
* system\_status.json - System health status data
* device\_list.json - Complete device inventory and status
* control\_connections.json - Control plane connection data
* bfd\_sessions.json - BFD session information
* omp\_peers.json - OMP peer relationship data
* current\_alarms.json - Active alarms and alerts
* certificate\_status.json - Certificate health information
* cluster\_status.json - Cluster management status

### **Task 22: Health Summary Report**

**Purpose:** Creates a human-readable summary of overall health status

**Generated file:** health\_check\_summary.txt

**Report contents:**

* Execution timestamp
* vManage host information
* Health check status summary (PASS/FAIL for each component)
* Device summary (total, online, offline counts)
* Network status (connections, sessions, peers, alarms)
* Overall health assessment (HEALTHY/ISSUES DETECTED)

## **Output Files**

All output files are saved to the generated/ directory:

### **JSON Data Files**

* **connectivity\_test.json** - Raw API connectivity test results
* **system\_status.json** - System health status data
* **device\_list.json** - Complete device inventory with status
* **control\_connections.json** - Control plane connection information
* **bfd\_sessions.json** - BFD session details
* **omp\_peers.json** - OMP peer relationship data
* **current\_alarms.json** - Active system alarms
* **certificate\_status.json** - Certificate validation status
* **cluster\_status.json** - Cluster health information

### **Summary Report**

* **health\_check\_summary.txt** - Human-readable health status summary

## **Health Check Criteria**

### **Overall Health Assessment**

The playbook determines overall health based on:

* API connectivity must be successful (PASS)
* System status must be healthy (PASS)
* At least one device must be online

If any of these criteria fail, the overall status is marked as "ISSUES DETECTED"

### **Individual Component Status**

Each health check component is evaluated independently:

* **PASS** - Component is healthy and functioning normally
* **FAIL** - Component has issues or is unreachable

## **Usage Instructions**

### **Manual Execution**

* Set required environment variables:  
    
   export VMANAGE\_HOST="your-vmanage-host.com"
* export VMANAGE\_USERNAME="your-username"
* export VMANAGE\_PASSWORD="your-password"
* Run the playbook:  
    
   ansible-playbook health\_checks.yml

1. Check results in the generated/ directory

### **Pipeline Integration**

The playbook can be integrated into CI/CD pipelines for automated health monitoring:

* Schedule regular health checks
* Generate alerts based on health status
* Track health trends over time