# **SD-WAN Device Monitoring Playbook Documentation**

## **Overview**

The **monitor\_devices.yml** playbook is an Ansible automation script designed to monitor Cisco SD-WAN device status and performance. This playbook connects to the vManage controller to collect real-time device information and saves the data as JSON files for analysis and troubleshooting.

## **Use Case**

**Use Case 14: Monitor devices - Get real-time SD-WAN device status and performance data**

This playbook addresses the need to:

* Monitor all SD-WAN devices in the network
* Check device operational status and health
* Collect device performance statistics and counters
* Get network connectivity information (BFD, OMP, control connections)
* Export monitoring data for analysis and reporting
* Provide automated device monitoring for operations teams

## **Prerequisites**

### **Environment Variables**

The following environment variables must be set before running the playbook:

| **Variable** | **Description** | **Default Value** |
| --- | --- | --- |
| **VMANAGE\_HOST** | vManage controller hostname/IP | vmanage-amfament-prod.sdwan.cisco.com |
| **VMANAGE\_USERNAME** | Username for vManage authentication | automation |
| **VMANAGE\_PASSWORD** | Password for vManage authentication |  |

## **Playbook Structure**

### **Variables Configuration**

vars:

vmanage\_host: "{{ lookup('env', 'VMANAGE\_HOST') | default('vmanage-amfament-prod.sdwan.cisco.com') }}"

vmanage\_username: "{{ lookup('env', 'VMANAGE\_USERNAME') | default('automation') }}"

vmanage\_password: "{{ lookup('env', 'VMANAGE\_PASSWORD') | default('') }}"

vmanage\_port: "443"

generated\_dir: "{{ playbook\_dir }}/../generated"

### **Directory Structure**

The playbook creates the following directory structure:

playbook\_directory/

├── monitor\_devices.yml

└── generated/

├── devices\_inventory.json

├── device\_status.json

├── device\_counters.json

├── control\_connections.json

├── bfd\_sessions.json

├── omp\_peers.json

├── system\_info.json

└── monitoring\_summary.txt

## **Task Analysis**

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

**Purpose:** Makes sure all required login details are available before starting

**What it does:**

* Checks that **VMANAGE\_HOST**, **VMANAGE\_USERNAME**, and **VMANAGE\_PASSWORD** are set
* Stops immediately if any required information is missing
* Prevents errors due to missing login details
* Shows clear error messages for troubleshooting

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

**Purpose:** Creates the output folder for monitoring data

**What it does:**

* Creates the **generated** folder next to the playbook file
* Sets proper file permissions (755) for access
* Makes sure the output location exists before collecting data
* Creates parent folders if they don't exist

#### **Task 3: vManage Connectivity Test**

**Purpose:** Checks if the vManage controller can be reached before collecting data

**What it does:**

* Makes a REST API call to **/dataservice/system/device/controllers**
* Uses basic login with provided username and password
* Sets **60-second timeout** to handle slow connections
* Ignores SSL certificate validation for internal certificates
* Saves connection results for checking

#### **Task 4: Connectivity Validation**

**Purpose:** Stops the playbook if connection test fails

**What it does:**

* Checks if the connection test returned **HTTP 200** status
* Stops the playbook with error message if vManage cannot be reached
* Prevents data collection when connection problems exist
* Shows clear failure messages for troubleshooting

#### **Task 5: Get All Devices Inventory**

**Purpose:** Collects complete list of all SD-WAN devices

**API endpoint:** /dataservice/device

**What it does:**

* Gets list of all devices in the SD-WAN network
* Collects basic device information like hostname, device type, site ID
* Saves device inventory data for reference
* Provides foundation data for other monitoring tasks

#### **Task 6: Get Device Status**

**Purpose:** Collects operational status for all devices

**API endpoint:** /dataservice/device/monitor

**What it does:**

* Gets current operational status of all devices
* Shows which devices are up, down, or having issues
* Collects reachability and connectivity status
* Provides real-time device health information

#### **Task 7: Get Device Counters**

**Purpose:** Collects performance statistics and counters

**API endpoint:** /dataservice/device/counters

**What it does:**

* Gets device performance statistics
* Collects traffic counters and utilization data
* Shows interface statistics and throughput
* Provides performance monitoring data

#### **Task 8: Get Control Connections Status**

**Purpose:** Monitors control plane connectivity

**API endpoint:** /dataservice/device/control/connections

**What it does:**

* Gets control connection status between devices
* Shows vSmart controller connectivity
* Monitors control plane health
* May fail in sandbox environments (handled gracefully)

#### **Task 9: Get BFD Sessions**

**Purpose:** Monitors BFD (Bidirectional Forwarding Detection) sessions

**API endpoint:** /dataservice/device/bfd/sessions

**What it does:**

* Gets BFD session status between devices
* Shows tunnel health and connectivity
* Monitors fast failure detection
* May fail in sandbox environments (handled gracefully)

#### **Task 10: Get OMP Peers**

**Purpose:** Monitors OMP (Overlay Management Protocol) peer relationships

**API endpoint:** /dataservice/device/omp/peers

**What it does:**

* Gets OMP peer status and relationships
* Shows routing protocol connectivity
* Monitors overlay network health
* May fail in sandbox environments (handled gracefully)

#### **Task 11: Get System Information**

**Purpose:** Collects system details for all devices

**API endpoint:** /dataservice/device/system/info

**What it does:**

* Gets system information like CPU, memory, uptime
* Collects software version information
* Shows hardware details and status
* Provides comprehensive system health data

#### **Tasks 12-18: Save Data Files**

**Purpose:** Saves all collected monitoring data to JSON files

**Generated files:**

* **devices\_inventory.json** - Complete device list
* **device\_status.json** - Device operational status
* **device\_counters.json** - Performance statistics
* **control\_connections.json** - Control plane status
* **bfd\_sessions.json** - BFD session information
* **omp\_peers.json** - OMP peer relationships
* **system\_info.json** - System information

**What it does:**

* Converts API responses to readable JSON format
* Only saves files when data is successfully collected
* Creates separate files for different types of monitoring data
* Preserves all collected information for analysis

#### **Task 19: Create Monitoring Summary**

**Purpose:** Creates a summary report of the monitoring execution

**Generated file:** **monitoring\_summary.txt**

**What it does:**

* Shows monitoring execution date and time
* Lists all generated files and their purpose
* Shows total number of devices monitored
* Provides quick overview of monitoring results

## **Monitoring Data Contents**

The generated files typically include:

* **Device Inventory:** Device hostnames, types, site IDs, serial numbers
* **Device Status:** Up/down status, reachability, last seen times
* **Device Counters:** Traffic statistics, interface utilization, error counts
* **Control Connections:** vSmart connectivity, control plane health
* **BFD Sessions:** Tunnel health, session status, failure detection
* **OMP Peers:** Routing peer relationships, overlay connectivity
* **System Information:** CPU usage, memory, uptime, software versions