# **SD-WAN Device Configuration Retrieval Playbook Documentation**

## **Overview**

The **get\_device\_configuration.yml** playbook is an Ansible automation script designed to retrieve device configurations from Cisco SD-WAN environments. This playbook leverages the vManage REST API to extract detailed device configuration information from the vManage controller and produces organized configuration files for analysis, backup, and documentation purposes.

## **Use Case**

**Use Case 11: Get device configuration - Fetch device configuration**

This playbook addresses the need to:

* Retrieve running configurations from SD-WAN edge devices
* Extract device-specific configuration details for backup purposes
* Document device settings for change management and compliance
* Export configuration data for offline analysis and troubleshooting
* Provide automated configuration retrieval for regular audits and documentation updates

## **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 | (required) |

## **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"

device\_type: "vedge"

device\_id: ""

### **Directory Structure**

The playbook creates the following directory structure:

playbook\_directory/

├── get\_device\_configuration.yml

└── generated/

├── device\_config\_10.10.1.11.txt

├── device\_config\_10.10.1.13.txt

├── device\_config\_10.10.1.15.txt

├── device\_config\_10.10.1.17.txt

└── device\_inventory\_summary.txt

## **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 required environment variable is missing
* Prevents execution failures due to missing credentials
* Provides clear error messages for troubleshooting

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

**Purpose:** Creates the output directory for generated configuration files

**What it does:**

* Creates the **generated** directory relative to the playbook location
* Sets appropriate permissions (755) for file access
* Ensures the output location exists before configuration retrieval
* Uses the "../generated" path structure as specified

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

**Purpose:** Verifies the vManage controller is accessible before attempting configuration retrieval

**What it does:**

* Makes a REST API call to **/dataservice/system/device/controllers**
* Uses basic authentication with provided credentials
* Sets **60-second timeout** to handle slow connections
* Ignores SSL certificate validation for internal/self-signed certificates
* Stores connectivity results for validation

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

**Purpose:** Stops execution if connectivity test fails

**What it does:**

* Checks if the connectivity test returned **HTTP 200** status
* Fails the playbook with descriptive error if vManage is unreachable
* Prevents unnecessary API operations when connectivity issues exist
* Provides clear failure messaging for troubleshooting

#### **Task 5: Retrieve Device Inventory**

**Purpose:** Gets complete list of devices from vManage

**API endpoint:** /dataservice/device

**What it does:**

* Makes GET request to retrieve all devices in the SD-WAN fabric
* Collects comprehensive device information including:
  + Device ID and hostname
  + Device type and model
  + System IP and site ID
  + Operational status and version
  + UUID and reachability status

#### **Task 6: Filter Devices by Type**

**Purpose:** Applies device type filtering to target specific devices

**What it does:**

* Filters device list by **device\_type** variable (default: "vedge")
* Supports filtering by: vedge, vmanage, vsmart, vbond
* Creates filtered device list for targeted configuration retrieval
* Maintains all devices if no filter specified

#### **Task 7: Filter by Device ID (Optional)**

**Purpose:** Allows targeting of specific individual devices

**What it does:**

* Further filters by specific **device\_id** if provided
* Enables single device configuration retrieval
* Useful for troubleshooting or targeted configuration extraction

#### **Task 8: Retrieve Device Configurations**

**Purpose:** Fetches running configurations from filtered devices

**API endpoint:** /dataservice/template/config/attached/{uuid}

**What it does:**

* Makes GET requests for each device using device UUID
* Retrieves complete template-attached configurations
* Uses 120-second timeout for large configurations
* Handles API errors gracefully with ignore\_errors
* Stores configuration data for file generation

#### **Task 9: Retrieve Device RMA Details (Fallback)**

**Purpose:** Attempts alternative data retrieval method

**API endpoint:** /dataservice/system/device/rma/{deviceId}

**What it does:**

* Provides fallback method for device information
* Attempts RMA detail retrieval for additional device data
* Handles 404 errors gracefully (expected for most environments)
* Supplements primary configuration data when available

#### **Task 10: Save Device Configuration Files**

**Purpose:** Creates individual configuration files for each device

**Generated files:** device\_config\_{deviceId}.txt

**What it does:**

* Creates separate file for each successfully retrieved configuration
* Includes comprehensive device metadata:
  + Device identification (ID, hostname, type)
  + Network information (system IP, site ID)
  + Operational data (status, version, reachability)
* Stores complete configuration data in readable format
* Handles API errors by including error details

#### **Task 11: Save RMA Detail Files**

**Purpose:** Creates supplementary device detail files

**Generated files:** device\_rma\_{deviceId}.txt

**What it does:**

* Saves RMA detail information when available
* Provides additional device information for reference
* Includes error details when RMA endpoint is unavailable
* Complements primary configuration files

#### **Task 12: Create Device Inventory Summary**

**Purpose:** Generates consolidated device inventory report

**Generated file:** device\_inventory\_summary.txt

**What it does:**

* Creates summary of all processed devices
* Includes filtering criteria and device counts
* Lists device details for each discovered device:
  + Device ID, hostname, and type
  + System IP and site identification
  + Current status and software version
* Provides quick reference for device inventory

#### **Task 13: Display Completion Results**

**Purpose:** Provides execution status and file locations

**What it displays:**

* Total number of devices processed
* Success confirmation message
* Full path to generated files directory
* Summary of operation completion

## **Configuration File Contents**

The generated device configuration files typically include:

* **Device Metadata:** Complete device identification and status information
* **System Configuration:** Hostname, system IP, site ID, and version details
* **Network Configuration:** Interface settings, IP addressing, and tunnel configurations
* **VPN Settings:** VPN configurations and routing information
* **Security Policies:** IPSec settings and authentication configurations
* **Template Information:** Applied template names and configuration sources
* **Routing Configuration:** Static routes, OSPF settings, and OMP configurations
* **Service Settings:** System services and feature configurations

## **API Endpoints Used**

| **Endpoint** | **Purpose** | **Method** |
| --- | --- | --- |
| /dataservice/system/device/controllers | Connectivity test | GET |
| /dataservice/device | Device inventory retrieval | GET |
| /dataservice/template/config/attached/{uuid} | Device configuration retrieval | GET |
| /dataservice/system/device/rma/{deviceId} | Device RMA details (fallback) | GET |

## **Error Handling**

The playbook implements comprehensive error handling:

* **Connectivity failures:** Stops execution with clear error messages
* **API errors:** Continues processing other devices, logs errors in output files
* **Missing configurations:** Documents unavailability with error details
* **Endpoint failures:** Uses fallback methods where available

## **Customization Options**

* **Device filtering:** Modify device\_type to target specific device types
* **Single device:** Set device\_id to retrieve specific device only
* **Output location:** Adjust generated\_dir path as needed
* **Timeout values:** Modify API timeout values for different network conditions