# Inventory Device Serial Headless Operation

## Endpoints Summary

| Method | Path | Swagger |
| --- | --- | --- |
| `GET` | `/inventory_device_serial_headless_operation/` | Swagger ↗ |
| `POST` | `/inventory_device_serial_headless_operation/` | Swagger ↗ |

---

> The Inventory Device Serial Headless Operation endpoints allow you to schedule and manage automated device operations on network devices using their serial numbers. These endpoints enable you to perform TR-069/CWMP operations like getting device parameters, setting configuration values, and managing objects without manual intervention.

**Base URL:** `https://control.zequenze.com/api/v1`

**Authentication:** All endpoints require a Bearer token:

```
Authorization: Bearer <your-api-token>
```

---

## Overview

The Inventory Device Serial Headless Operation API provides a powerful way to manage network devices programmatically through scheduled operations. This API category is specifically designed for automating device management tasks on TR-069/CWMP compatible devices using their serial numbers as identifiers.

**Key Capabilities:**

- **Get Operations**: Retrieve current parameter values from devices (e.g., WiFi statistics, management server settings)
- **Set Operations**: Configure device parameters remotely (e.g., periodic inform intervals, enable/disable features)
- **Add Object Operations**: Create new configuration objects on devices
- **Delete Object Operations**: Remove existing configuration objects from devices

**Common Use Cases:**

- Automated device configuration deployment across multiple devices
- Scheduled parameter collection for monitoring and analytics
- Bulk device management operations
- Remote troubleshooting and diagnostics
- Compliance checking and configuration validation

The operations are executed asynchronously, allowing you to schedule tasks that will be performed when devices next communicate with the management server. The optional `update_status` parameter enables real-time status updates before returning operation results.

---

# Endpoints

## GET /inventory_device_serial_headless_operation/

**Description:** Retrieves a list of scheduled headless operations for devices. This endpoint allows you to view pending, completed, or filtered operations based on transaction ID. Use this to monitor the status of your scheduled device operations and retrieve results.

**Use Cases:**

- Monitor the status of previously scheduled device operations
- Retrieve results from completed device parameter queries
- Check pending operations before scheduling new ones
- Audit trail of device management activities

**Full URL Example:**

```
https://control.zequenze.com/api/v1/inventory_device_serial_headless_operation/?id=12345&update_status=true
```

**Parameters:**

| Parameter | Type | In | Required | Description |
|---|---|---|---|---|
| id | string | query | No | ID of the scheduled transaction to filter results |
| update_status | boolean | query | No | Use configured helpers to update device status before returning the information |

### cURL Example:

```
curl -X GET
"https://control.zequenze.com/api/v1/inventory_device_serial_headless_operation/?update_status
=true" \
  -H "Authorization: Bearer YOUR_API_TOKEN" \
  -H "Content-Type: application/json"
```

### Example Response:


```
[
  {
    "serial_number": "DV001234567890",
    "operation": "get",
    "variables": [
      {
        "variable_name": "Device.ManagementServer.PeriodicInformInterval",
        "value": "300",
        "status": "completed"
      },
      {
        "variable_name": "Device.WiFi.SSID.1.Stats.BytesReceived",
        "value": "1048576000",
        "status": "completed"
      }
    ]
  },
  {
    "serial_number": "DV001234567891",
    "operation": "set",
    "variables": [
      {
```

```
        "variable_name": "Device.ManagementServer.PeriodicInformInterval",
        "value": "60",
        "value_type": "integer",
        "status": "pending"
      }
    ]
  }
]
```

**Response Codes:**

| Status | Description |
|---|---|
| 200 | Success - Returns the list of scheduled operations |
| 401 | Unauthorized - Invalid or missing API token |
| 403 | Forbidden - Insufficient permissions |
| 500 | Internal Server Error - Server processing error |

---

# POST /inventory_device_serial_headless_operation/

**Description:** Creates a new scheduled headless operation for one or more devices. This endpoint allows you to queue device operations that will be executed when the target devices next communicate with the management server. Support for get, set, add object, and delete object operations.

**Use Cases:**

- Schedule parameter retrieval from multiple devices for monitoring
- Deploy configuration changes across a fleet of devices
- Add new WiFi SSIDs or network configurations remotely
- Remove obsolete configuration objects from devices

**Full URL Example:**

```
https://control.zequenze.com/api/v1/inventory_device_serial_headless_operation/
```

**Parameters:**

| Parameter | Type | In | Required | Description |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| data | object | body | Yes | JSON object containing the operation details |

### Request Body Schema:

```
{
  "serial_number": "string",
  "operation": "get|set|add.obj|del.obj",
  "variables": [
    {
      "variable_name": "string",
      "value": "string",
      "value_type": "string|integer|boolean"
    }
  ]
}
```

### cURL Example - Get Operation:

```
curl -X POST "https://control.zequenze.com/api/v1/inventory_device_serial_headless_operation/"
\
  -H "Authorization: Bearer YOUR_API_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "serial_number": "DV001234567890",
    "operation": "get",
    "variables": [
      {"variable_name": "Device.ManagementServer.PeriodicInformInterval"},
      {"variable_name": "Device.WiFi.SSID.1.Stats.BytesReceived"}
    ]
  }'
```

### cURL Example - Set Operation:

```
curl -X POST "https://control.zequenze.com/api/v1/inventory_device_serial_headless_operation/"
\
  -H "Authorization: Bearer YOUR_API_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "serial_number": "DV001234567890",
```

```
    "operation": "set",
    "variables": [
      {
        "variable_name": "Device.ManagementServer.PeriodicInformInterval",
        "value": "60",
        "value_type": "integer"
      },
      {
        "variable_name": "Device.ManagementServer.PeriodicInformEnable",
        "value": "1",
        "value_type": "boolean"
      }
    ]
  }'
```

**Example Response:**

```
{
  "id": "txn_789012345",
  "serial_number": "DV001234567890",
  "operation": "set",
  "variables": [
    {
      "variable_name": "Device.ManagementServer.PeriodicInformInterval",
      "value": "60",
      "value_type": "integer"
    },
    {
      "variable_name": "Device.ManagementServer.PeriodicInformEnable",
      "value": "1",
      "value_type": "boolean"
    }
  ],
  "status": "scheduled",
  "created_at": "2024-01-15T10:30:00Z"
}
```

**Response Codes:**

| Status | Description |
|---|---|

| | |
|---|---|
| 201 | Created - Operation successfully scheduled |
| 400 | Bad Request - Invalid operation parameters or malformed request |
| 401 | Unauthorized - Invalid or missing API token |
| 403 | Forbidden - Insufficient permissions |
| 404 | Not Found - Device with specified serial number not found |
| 422 | Unprocessable Entity - Valid JSON but invalid operation data |

# Common Use Cases

## Use Case 1: Bulk Device Configuration Monitoring

Schedule periodic retrieval of key configuration parameters across multiple devices to ensure compliance with corporate policies. Use GET operations to collect parameters like periodic inform intervals, WiFi settings, and security configurations.

## Use Case 2: Automated Device Onboarding

When new devices are deployed, use SET operations to automatically configure management server settings, WiFi credentials, and other essential parameters without requiring manual intervention.

## Use Case 3: WiFi Network Management

Use ADD.OBJ operations to deploy new WiFi SSIDs across all access points, or DEL.OBJ operations to remove obsolete network configurations during network updates.

## Use Case 4: Performance Monitoring and Analytics

Schedule GET operations to collect device statistics like bytes transmitted/received, connection counts, and performance metrics for analysis and reporting.

## Use Case 5: Security Configuration Updates

Use SET operations to update security parameters, change default passwords, or modify firewall settings across multiple devices simultaneously.

---

# Best Practices

- **Operation Scheduling**: Remember that operations are executed when devices next communicate with the management server. Consider device periodic inform intervals when planning time-sensitive operations.
- **Variable Naming**: Use complete TR-069/CWMP parameter paths (e.g., `Device.ManagementServer.PeriodicInformInterval`) to ensure proper parameter identification.
- **Value Types**: Always specify the correct `value_type` for SET operations (string, integer, boolean) to prevent type conversion errors on the device.
- **Batch Operations**: Group related parameters in a single operation request to minimize the number of device communications required.
- **Error Handling**: Monitor operation status using the GET endpoint and implement retry logic for failed operations. Check device connectivity if operations remain pending for extended periods.
- **Security Considerations**: Validate device serial numbers before scheduling operations and ensure proper access controls are in place for sensitive configuration parameters.
- **Status Updates**: Use the `update_status=true` parameter when you need real-time status information, but be aware this may increase response times for large device inventories.

---

Revision #4
Created 2026-02-04 05:08:16 UTC by ipena@zequenze.com
Updated 2026-02-11 03:01:18 UTC by ipena@zequenze.com