Monitor Catalyst 9800 WLC via SNMP with OIDs

Contents

Introduction

Prerequisites

- Requirements
- Components Used

SNMP vs Telemetry

Configure SNMP on WLC

- Through Web Interface
- Through Command Line

Object Names and Object IDs (OIDs)

- What are Object Names and OIDs?
- MIBs and a List of all Object Names and IDs on Cisco WLCs

Use OIDs to Monitor the State of WLC

- Monitor via Snmpwalk
- Monitor via Python3 And pysnmp Library
- Integration with Third-Party Software (Grafana+Prometheus/PRTG Network Monitor/SolarWinds)
- Integration With CUCM

Table of Most Commonly Monitored OIDs

Monitor Standby WLC in HA

- Monitor Standby WLC Directly
- Monitor Standby WLC through Active WLC

Introduction

This document describes how to configure Simple Network Management Protocol (SNMP) to monitor Cisco 9800 Wireless LAN Controller (WLC).

Prerequisites

Requirements

- SNMP Server/Tool
- Basic Knowledge of 9800 WLC and SNMP Protocol(s)

Components Used

All tests were performed on MacOS 10.14 and a 9800-CL WLC with image version 17.5.1. Some of the OIDs mentioned in this article do not exist on older image versions.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

SNMP vs Telemetry

Older AireOS WLCs rely on SNMP as the main protocol to monitor. The majority of relevant information like client count, number of joined access points, processor and memory usage can be obtained via SNMP queries from the tool that monitors to the WLC.

With the 9800 WLC, the focus has been placed on telemetry. Telemetry works in a "push" model where the WLC sends out relevant information to the server without the requirement to be queried. The Catalyst 9800 still offers SNMP for legacy purposes. Some information can be exclusive to telemetry and some of the OIDs previously available on AireOS are not yet available on 9800.

Configure SNMP on WLC

From Cisco IOS XE® Bengaluru 17.6.1, Ethernet Service Port (Management Interface VRF/GigabitEthernet 0) is supported in Cisco Catalyst 9800 Series Wireless Controller.

Prior to this release, the Catalyst 9800 WLC can only be monitored with SNMP via its Wireless Management Interface or via Redundancy Management Interface (in case of a standby WLC in HA cluster on versions 17.5.1 and higher).

Through Web Interface

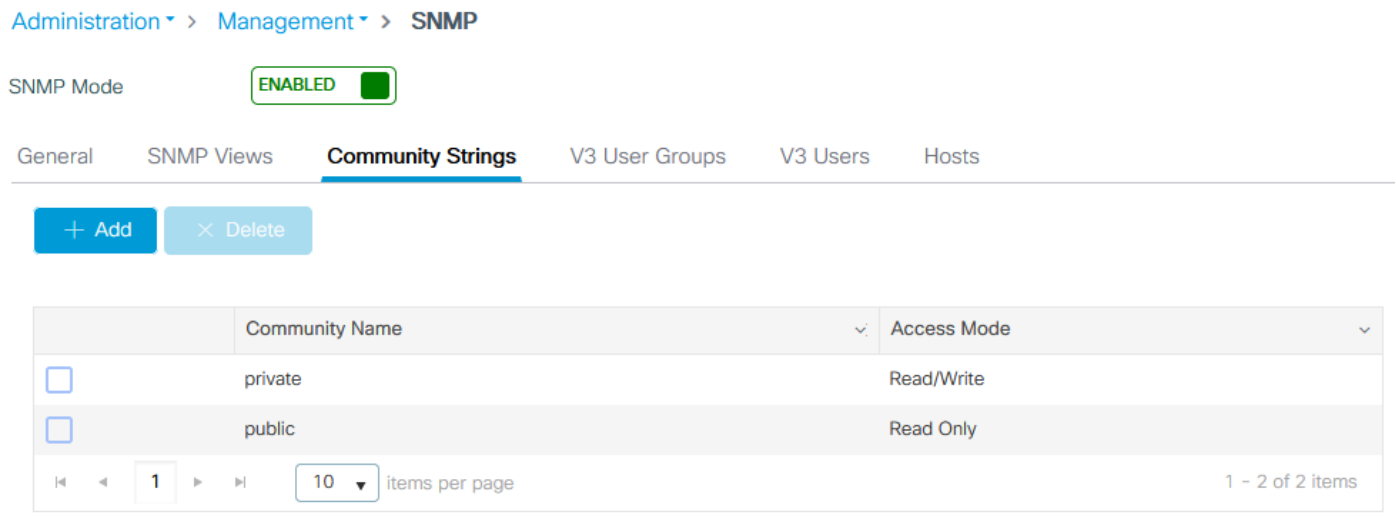
SNMPv2c is a community-based version of SNMP and all communication between the devices is in clear text. SNMPv3 is the most secure version that offers message integrity checks, authentication, and encryption of the packets. SNMPv1 is quite outdated, but still exists to provide legacy software compatibility. It is not mentioned in this article.

 **Important:** SNMPv2c is enabled by default with community “private” with read + write privileges

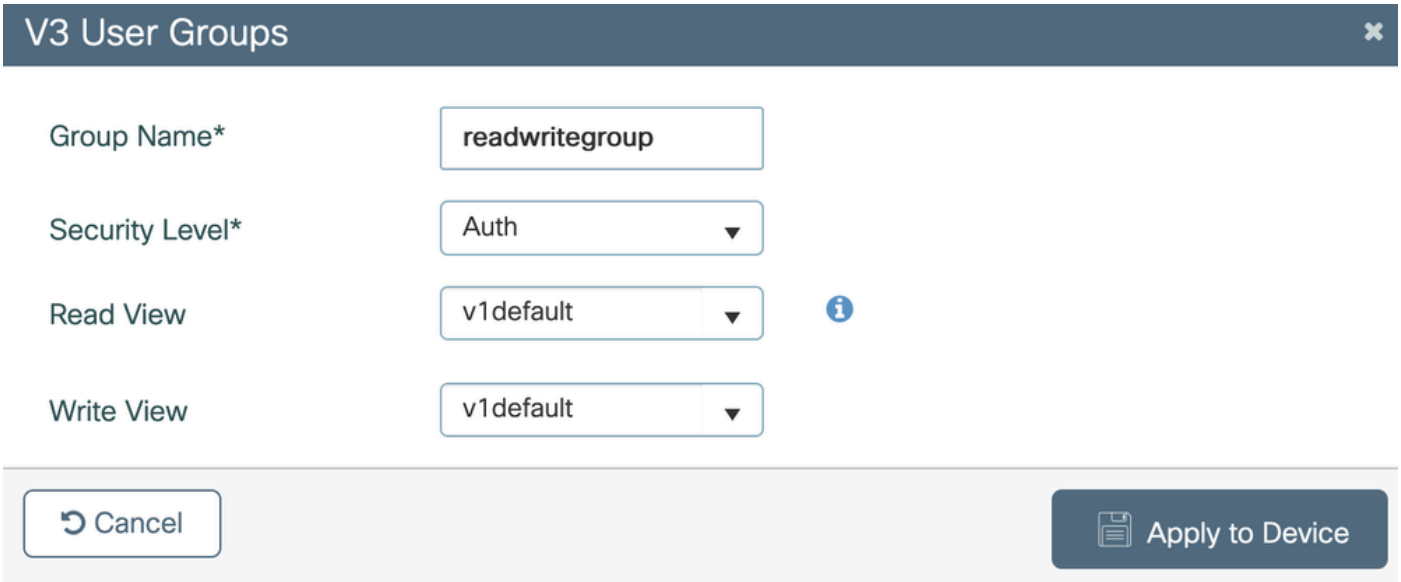
---

and community “public” with read-only privileges. It is recommended to remove these and create a new community with a different name.

1. Log into the web interface of the 9800 WLC. Under **Administration > Management > SNMP** and ensure SNMP is globally enabled. Under **Community Strings**, all currently configured communities and their permission level are displayed:



2. Before an SNMP V3 user is created, it is required to define an SNMP V3 group. To create a group of users with read + write permission, set the **Read View** and **Write View** to **v1default**. Leave the read only group **Write View** field box as empty:



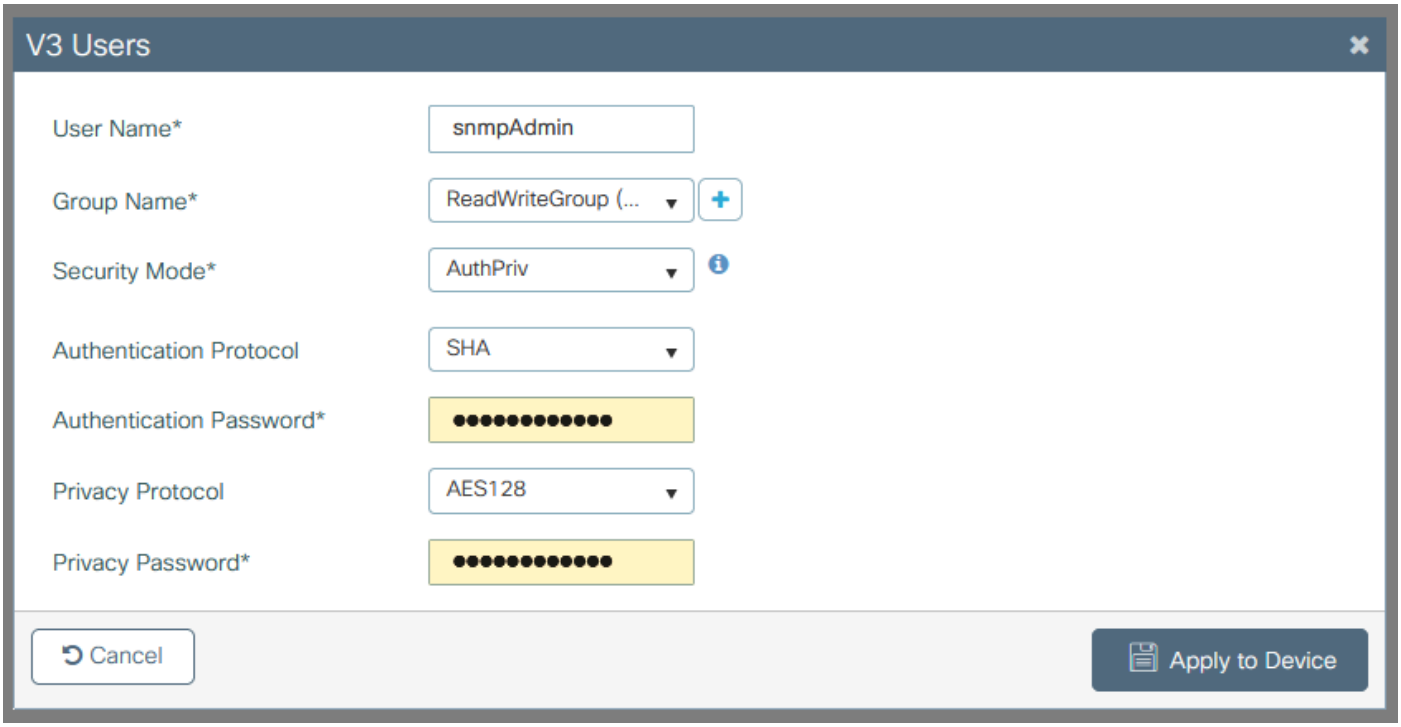
3. Under the **SNMP v3 Users** tab, you can see all configured users, and their privileges and protocols used for authentication and encryption. The **New** button allows you to create a new user.

There are 3 security modes available:

- **AuthPriv**: Messages are authenticated and encrypted.
- **AuthNoPriv**: Messages are authenticated but not encrypted.
- **NoAuthNoPriv**: No security applied to messages.

---

4. Select SHA as the authentication protocol and at least AES-128 as the privacy protocol is recommended.



Through Command Line

SNMP can also be configured through Command Line Interface (CLI). CLI offers additional configuration parameters, like the ability to assign an access list to a v2 community or v3 user. An example configuration of a v2 read + write community, v3 read + write group, and v3 user that belongs to this group:

```
snmp-server manager
snmp-server community <community-name> RW <access-list-number>
snmp-server community <community-name> RO <access-list-number>
snmp-server group <v3-group-name> v3 auth write v1default
snmp-server user <v3-user-name> <v3-group-name> v3 auth sha <auth-password> priv aes 128 <encryption-pa
```

This is an example of an access list that only allows the device on an IP address of: 192.168.10.10. To query the WLC v2 community named "ReadWriteCommunity":

```
ip access-list standard 50
 10 permit 192.168.10.10
 20 deny   any
snmp-server manager
snmp-server community ReadWriteCommunity RW 50
```

Note: At the time of this document, only standard ACLs are supported. Extended ACLs can be assigned, but they do not work.

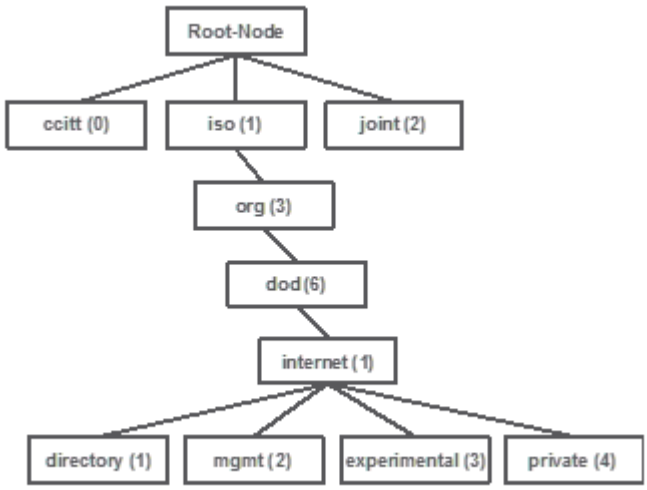
Object Names and Object IDs (OIDs)

What are Object Names and OIDs?

Object IDs, or OIDs for short, are unique identifiers that represent a certain variable or object. For example, the current processor usage is considered variable, which values can be retrieved with the call upon their object ID. Each OID is unique and no two can be the same, this is similar to a MAC address.

These identifiers align with a tree hierarchy, and each OID can be tracked down back to its root. Each vendor has its own branch with a common root. An analogy can be a home address, where the root is the country or state, with their city, zip code, street, and finally the home number.

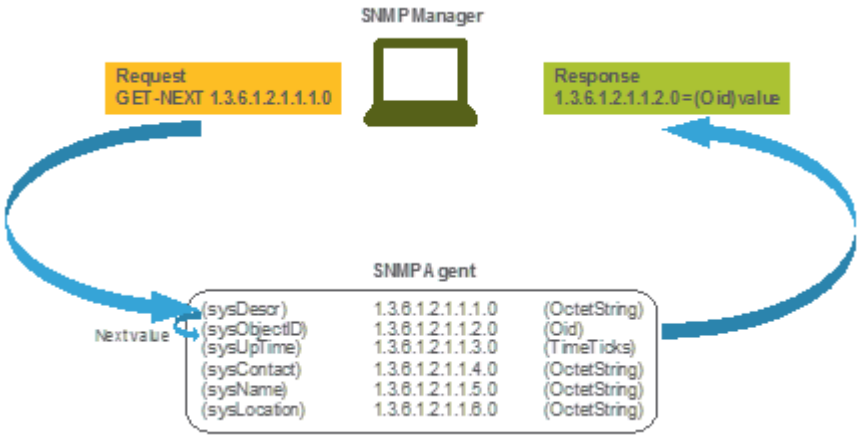
The numbers trailed by a dot represent each step it takes to get to a certain point in that tree or branch.



All these values are stored in a Management Information Base, or MIB for short in each network device. Each identifier has name and a definition (range of possible values, type, and so on).

It is not required to load an MIB on your SNMP monitoring tool to use SNMP to query a device. As long as a valid OID is known, the device responds with the value stored in the variable the OID represents. However, if you load the MIB on your query tool, it provides the benefit of translating the object names to their IDs and explains their description.

In this example, the SNMP tool queries the SNMP agent of a device for its system description with the use of OID 1.3.6.1.2.1.1.1.0.



MIBs and a List of all Object Names and IDs on Cisco WLCs

Cisco offers MIBs for 9800 WLCs. It is not easily readable, however, MIB contains all available object names and their descriptions.

All 9800 models (9800-80, 9800-40, 9800-L, 9800-CL, EWC) use the same MIB which can be downloaded at the Cisco Feature Navigator. The most up-to date one is with the most recent date, not the one with the higher code version name.

Downloaded archive files contain multiple **.my** text files that can either be imported into any third-party SNMP server or simply opened with a text editor. To find the OID of a specific object name, you must locate the exact file that contains it first.

For example, all objects related to monitoring the devices physical state (like CPU and memory) are located inside a MIB called CISCO-PROCESS-MIB.my.

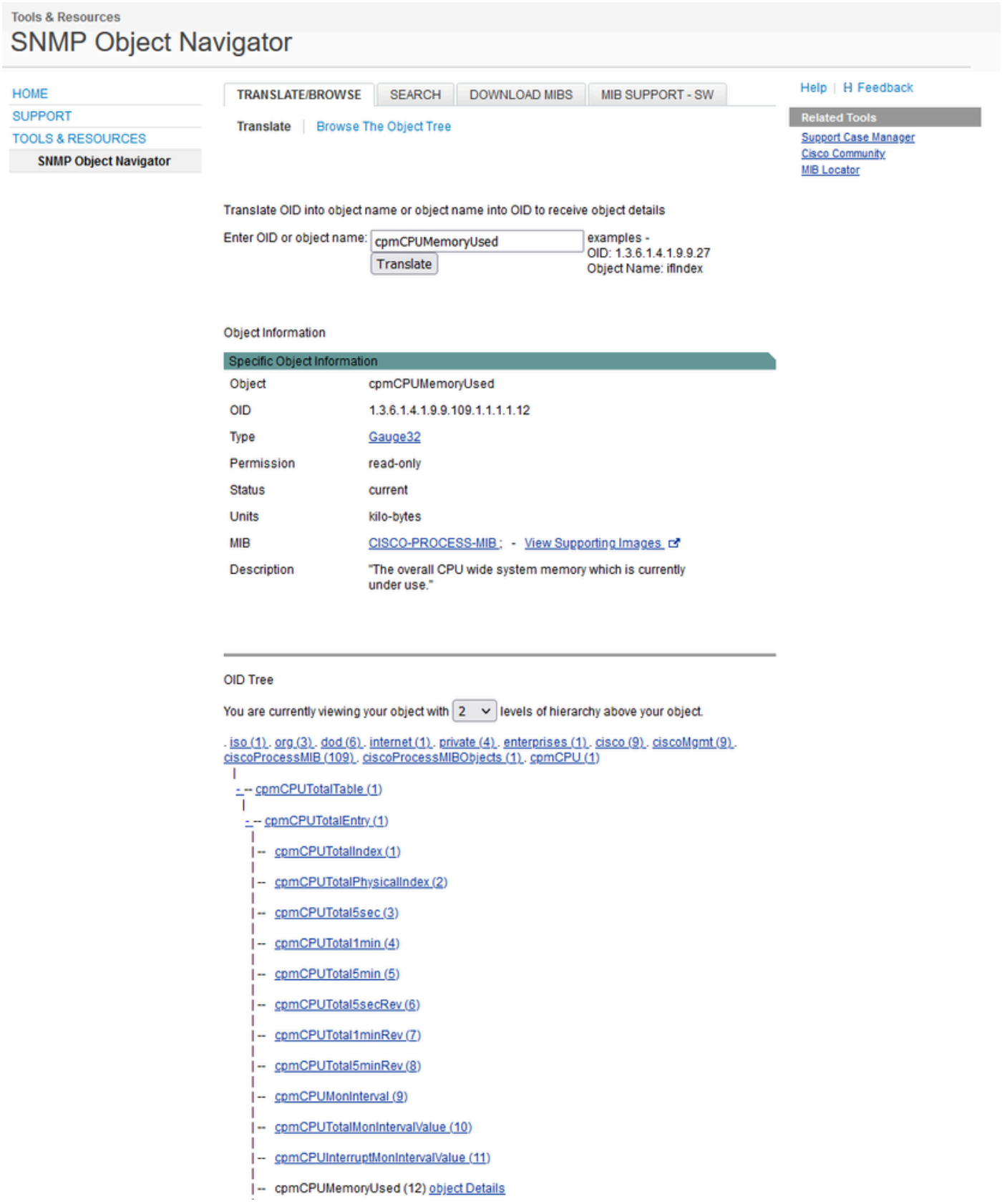
The “cpmCPUMemoryUsed” is the object name that is used to provide the amount of memory used by the WLC in bytes. MIB files all align with similar syntax. Information on the used memory object looks like this:

```
cpmCPUMemoryUsed OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "kilo-bytes"
```

```
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "The overall CPU wide system memory which is currently
    under use."
::= { cpmCPUTotalEntry 12 }
```

Most the third-party software to monitor relies on OIDs, and not object names. Translation between object name and object ID can be done with the usage of the Cisco SNMP Object Navigator Tool.

To utilize this tool, you enter the object name into the search bar and the output provides the OID and a short description. Additionally, the same tool can be used to find the object name of the provided OID.



Use OIDs to Monitor the State of WLC

After the acquisition, the OID of the object that must be monitored, the first SNMP query can be executed. Examples in this chapter showcase how to acquire a WLC free memory (OID =

1.3.6.1.4.1.9.9.48.1.1.1.5) for SNMPv2 community private and SNMPv3 user snmpadmin with SHA Auth password: **Cisco123#** and AES Privacy password: **Cisco123#**. The controller management interface is located on: 10.48.39.133.

Monitor via Snmpwalk

Snmpwalk is an SNMP application that uses SNMP GETNEXT requests to query a network entity for a tree of information. It is present by default on MacOS and most Linux distributions. For SNMPv2c, the command complies with this syntax:

```
snmpwalk -v2c -c <community_name> <WLC_management_interface_ip> <OID>
```

This is another example:

```
<#root>

VAPEROVI:~ vaperovi$

snmpwalk -v2c -c private 10.48.39.133 1.3.6.1.4.1.9.9.109.1.1.1.1.12



SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.2 =

3783236

<-------- Free Memory in Bytes
```

If SNMPv3 is used, the command adheres to the syntax:

```
snmpwalk -v3 -l authPriv -u <username> -a [MD5|SHA] -A <auth_password> -x [AES|DES] -X <priv_password>
```

Select MD5/SHA and AES/DES based on how you created the SNMPv3 user on the controller. This is another example:

```
<#root>

VAPEROVI:~ vaperovi$ snmpwalk -v3  -l authPriv -u snmpadmin -a SHA -A Cisco123# -x AES -X Cisco123# 10.
SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.2 =
```

```
3783236

<-------- Free Memory in Bytes
#snmpwalk output still shows v2 even though v3 is used
```

Monitor via Python3 And pysnmp Library

Code snippets are written for Python 3.9 and utilize the **pysnmp** module (pip install pysnmp) to run SNMP queries for memory utilization of Catalyst 9800-CL WLC. These examples use the same SNMPv2 community and SNMPv3 user created in the previous chapters. Simply replace the variable values and integrate the code within your own custom scripts.

SNMPv2 Example:

```
from pysnmp.hlapi import *

communityName = 'private'
ipAddress = '10.48.39.133'
OID = '1.3.6.1.4.1.9.9.109.1.1.1.1.12'

for (errorIndication,
     errorStatus,
     errorIndex,
     varBinds) in nextCmd(SnmpEngine(),
                          CommunityData(communityName),
                          UdpTransportTarget((ipAddress, 161)),
                          ContextData(),
                          ObjectType(ObjectIdentity(OID)),
                          lexicographicMode=False):

    if errorIndication:
        print(errorIndication)
    elif errorStatus:
        print('%s at %s' % (errorStatus.prettyPrint(),
                            errorIndex and varBinds[int(errorIndex) - 1][0] or '?'))
    else:
        for varBind in varBinds:
            print(' = '.join([x.prettyPrint() for x in varBind]))
```

Output prints out:

```
SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.2 = 3783236
```

SNMPv3 Example:

```
from pysnmp.hlapi import *

username = 'snmpadmin'
ipAddress = '10.48.39.133'
OID = '1.3.6.1.4.1.9.9.109.1.1.1.1.12'
authKey = 'Cisco123#'
privKey = 'Cisco123#'

for (errorIndication,
     errorStatus,
     errorIndex,
     varBinds) in nextCmd(SnmpEngine(),
                          UsmUserData(username, authKey, privKey,
                                      authProtocol=usmHMACSHAAuthProtocol,
                                      privProtocol=usmAesCfb128Protocol),
                          UdpTransportTarget((ipAddress, 161)),
                          ContextData(),
                          ObjectType(ObjectIdentity(OID)),
                          lexicographicMode=False):

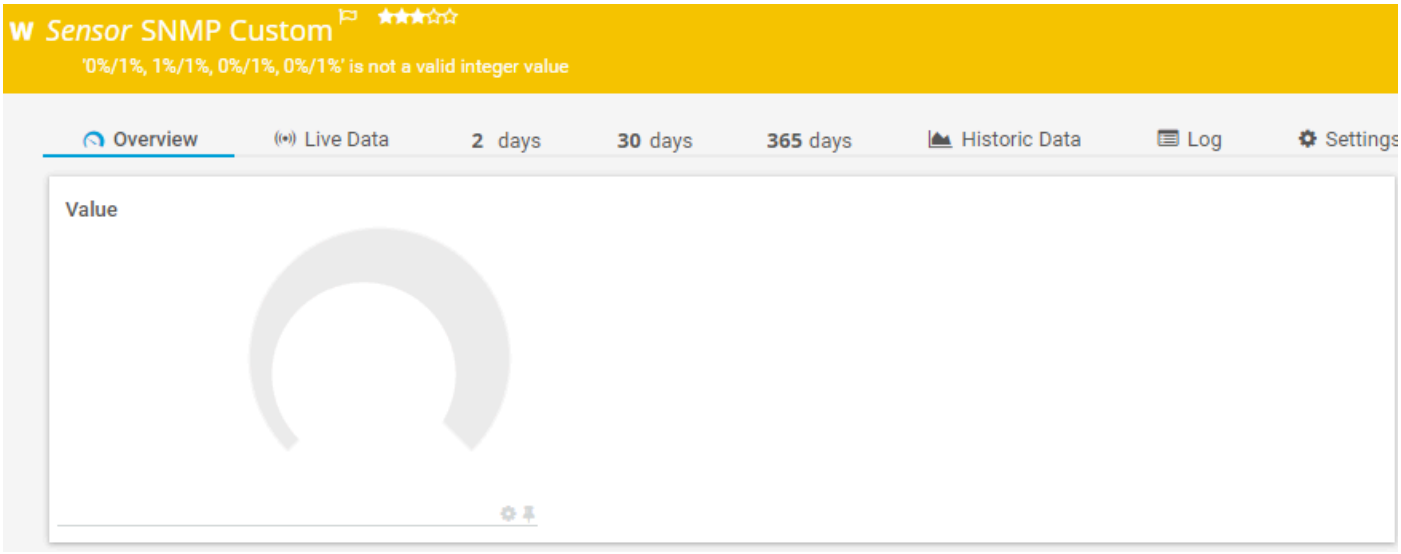
    if errorIndication:
        print(errorIndication)
    elif errorStatus:
        print('%s at %s' % (errorStatus.prettyPrint(),
                            errorIndex and varBinds[int(errorIndex) - 1][0] or '?'))
    else:
        for varBind in varBinds:
            print(' = '.join([x.prettyPrint() for x in varBind]))
```

Integration with Third-Party Software (Grafana+Prometheus/PRTG Network Monitor/SolarWinds)

Cisco Prime Infrastructure offers the ability to easily monitor and configure multiple network devices, which includes wireless controllers. Prime Infrastructure comes preloaded with all the OIDs and integrations with WLC, and consists of the addition of the WLC credentials to Prime. With 9800 WLCs, Prime mostly relies on Telemetry to collect the majority of details from the WLC, while the small portion of information is obtained through SNMP.

On the other hand, Cisco WLC can also be integrated with multiple third-party solutions for monitoring, as long as the OIDs are known. Programs like Grafana+Prometheus, PRTG Network Monitor and SolarWinds servers allow the MIBs or OIDs to be imported and values to display in a user-friendly graph.

This integration can require some tweaks on the SNMP server side. In this example, the PRTG monitor server is provided with the per core CPU utilization OID, which returns the string “0%/1%, 1%/1%, 0%/1%, 0%/1%”. PRTG expects an integer value and raises an error.



Integration With CUCM

Cisco Unified Communications Manager (CUCM) has a Wireless Endpoint Tracking Feature that allows it to approximately track client location based on the AP the client is connected to. For this feature to function, the CUCM must pull information from the WLC via SNMP queries.

Important: Many CUCM releases are affected by Cisco bug ID: CSCvv07486 - Unable to Sync Access Points in WLC due to SNMP requests too large. This issue is triggered in situations where CUCM runs an affected release and the WLC has more than 10 APs. Due to the incorrect way the CUCM queries large amount of OIDs in a single bulk request, the WLC either refuses to answer or answers with a **tooBig** response. The tooBig response is not sent immediately and can be delayed. SNMP debugs on

Table of Most Commonly Monitored OIDs

The table includes some of the most common object names and their OIDs, with the consideration that MIBs present the data in non-user-friendly syntax:

Note: Running the **show snmp mib | in <Object name>** command can be used to verify if a specific object name is available on 9800 WLC.

Description	Object Name	OID	Expected Response
Overall CPU usage in % last 5 sec (equivalent to "show process	cpmCPUTotal5sec	1.3.6.1.4.1.9.9.109.1.1.1.1.3	INTEGER: 5

cpu platform")			
Overall CPU usage in % last 1 min	cpmCPUTotal1min	1.3.6.1.4.1.9.9.109.1.1.1.1.4	INTEGER: 5
Overall CPU usage in % last 5 min	cpmCPUTotal5min	1.3.6.1.4.1.9.9.109.1.1.1.1.5	INTEGER: 5
CPU usage in % last 5 min, per core	cpmCore5min	.1.3.6.1.4.1.9.9.109.1.1.2.1.5.5 (.0 to .X depending on number of cores)	INTEGER: 5
Current used memory in bytes	cpmCPUMemoryUsed	1.3.6.1.4.1.9.9.109.1.1.1.1.12	INTEGER: 3783236
Current free memory in bytes	cpmCPUMemoryFree	1.3.6.1.4.1.9.9.109.1.1.1.1.13	INTEGER: 4263578
Lowest amount of free memory since last boot in bytes	cpmCPUMemoryLowest	1.3.6.1.4.1.9.9.109.1.1.1.1.15	INTEGER: 4251212
Last reload reason	whyReload	1.3.6.1.4.1.9.2.1.2	STRING: "reload"
Software image of all joined APs	bsnAPSoftwareVersion	1.3.6.1.4.1.14179.2.2.1.1.8	STRING: "17.5.1.12"
Model number of all joined APs	bsnAPModel	1.3.6.1.4.1.14179.2.2.1.1.16	STRING: "AIR-AP1840I-E-K9"
Number of clients	cLApActiveClientCount	.1.3.6.1.4.1.9.9.513.1.1.1.1.72	Integer
Number of joined APs	cLApGlobalAPConnectCount	1.3.6.1.4.1.9.9.513.1.3.35	Gauge32: 1
Power Supply Unit State	ciscoEnvMonSupplyStatusTable	.1.3.6.1.4.1.9.9.13.1.5(added through CSCwb73163)	table

Fan State	ciscoEnvMonFanStatusTable	.1.3.6.1.4.1.9.9.13.1.4 (added through CSCwb73163)	table

Currently, there are enhancement requests open to support OIDs total number of clients and number of joined access points under Cisco bug ID: CSCvu26309 - SNMP OID for Client Count not Present on 9800.

State of the Power Supply Unit (PSU) and fan state are not supported at the time of this article. An enhancement request has been opened under
Cisco bug ID: CSCwa23598 - 9800 WLC Enhancement / Support for PSU and Fan State SNMP OID (1.3.6.1.4.1.9.9.13).

Monitor Standby WLC in HA

You can monitor a standby WLC in High Availability cluster from release 17.5.1. The standby WLC can either be monitored directly through RMI or with querying the active WLC.

Monitor Standby WLC Directly

Standby WLC can be monitored directly only for WLCs that run in RMI + RP HA type. This is done through the standby Redundancy Management Interface (RMI) IP address of the standby WLC.

In this scenario, only OIDs from **IF-MIB** are supported, which makes it possible to monitor the state of all interfaces on the standby WLC. An example output from 9800-CL WLC:

Description	Object Name	OID	Expected Response
Interface name	ifDescr	1.3.6.1.2.1.2.2.1.2	SNMPv2-SMI::mib-2.2.2.1.2.1 = GigabitEthernet1 SNMPv2-SMI::mib-2.2.2.1.2.2 = GigabitEthernet2 SNMPv2-SMI::mib-2.2.2.1.2.3 = GigabitEthernet3 SNMPv2-SMI::mib-2.2.2.1.2.4 = VoIP-Null0 SNMPv2-SMI::mib-2.2.2.1.2.5 = Null0 SNMPv2-SMI::mib-2.2.2.1.2.6 = Vlan1 SNMPv2-SMI::mib-2.2.2.1.2.7 = Vlan39

Interface operational state (1=up, 2=down)	ifOperStatus	1.3.6.1.2.1.2.2.1.8	SNMPv2-SMI::mib-2.2.2.1.8.1 = 2 SNMPv2-SMI::mib-2.2.2.1.8.2 = 2 SNMPv2-SMI::mib-2.2.2.1.8.3 = 1 SNMPv2-SMI::mib-2.2.2.1.8.4 = 2 SNMPv2-SMI::mib-2.2.2.1.8.5 = 2 SNMPv2-SMI::mib-2.2.2.1.8.6 = 1

Note: It is expected the standby WLC Gig 2 port (trunk port used for traffic switching) is in shutdown state. Once failover occurs, the Gig 2 port on the standby WLC populates. A similar situation occurs with TenGigabit ports on 9800 physical appliances (9800-80, 9800-40 and 9800-CL).

Monitor Standby WLC through Active WLC

The standby WLC state can also be monitored with the query to the active WLC. Only **CISCO-LWAPP-HA-MIB** and **CISCO-PROCESS-MIB** MIBs are officially supported. When an active WLC in HA is queried, the first response represents the value of the active WLC, while the second response represents the value of the standby WLC.

Description	Object Name	OID	Expected Response
Overall CPU usage in % last 5 sec	cpmCPUTotal5sec	1.3.6.1.4.1.9.9.109.1.1.1.1.3	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.3.5 = **3** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.3.6 = **7**
Overall CPU usage in % last 1 min	cpmCPUTotal1min	1.3.6.1.4.1.9.9.109.1.1.1.1.4	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.4.5 = **8** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.4.6 = **6**
Overall CPU usage in % last 5 min	cpmCPUTotal5min	1.3.6.1.4.1.9.9.109.1.1.1.1.5	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.5.5 = **10** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.5.6 = **15**

Current used memory in bytes	cpmCPUMemoryUsed	1.3.6.1.4.1.9.9.109.1.1.1.1.12	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.5 = **4318980** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.6 = **3950332**
Current free memory in bytes	cpmCPUMemoryFree	1.3.6.1.4.1.9.9.109.1.1.1.1.13	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.5 = **4318739** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.12.6 = **3950738**
Lowest amount of free memory since last boot in bytes	cpmCPUMemoryLowest	1.3.6.1.4.1.9.9.109.1.1.1.1.15	SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.15.5 = **3763868** SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.15.6 = **4132588**
State of the standby WLC (1=up, 0=down)	cLHaPeerHotStandbyEvent	1.3.6.1.4.1.9.9.843.1.3.4	SNMPv2-SMI::enterprises.9.9.843.1.3.4.0 = **1**