



# *Cisco Developers Connect*

May 21<sup>th</sup> 2024, Seoul Korea

# Ansible 활용 네트워크 점검 자동화 및 리포팅

허재 프로, Technical Solutions Specialist

jaheo@cisco.com

2024.02.21



# Agenda

- 무엇을 모니터링 할까요?
  - Spanning Tree Protocol
- 데이터 파싱 & Webex 연동 실습
  - Mission #1
- Jinja2 템플릿을 통한 리포트 생성
  - Mission #2



# Agenda

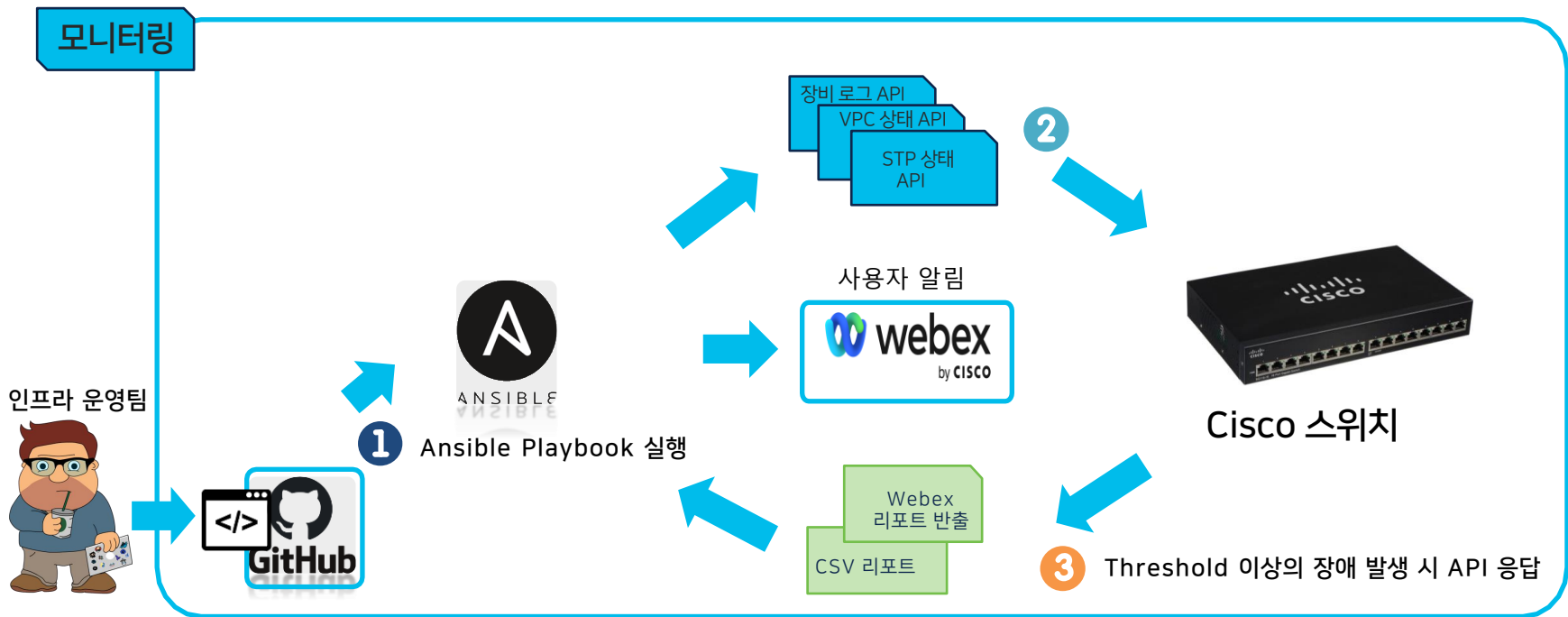
- 무엇을 모니터링 할까요?
  - Spanning Tree Protocol 예시 소개

장비에 직접 접속하지 않고, 스위치에 작업을 하려면 백대역의 작업을 한번에 할 수 있을까?



인프라 운영팀

# Ansible을 활용한 스위치 점검 자동화 Workflow



➡ 네트워크 점검 시 수백대의 스위치 정보를 단시간에 수집 및 모니터링 리포트 생성



# Spanning Tree Protocol (STP)

Port Up까지  
30초 소요!

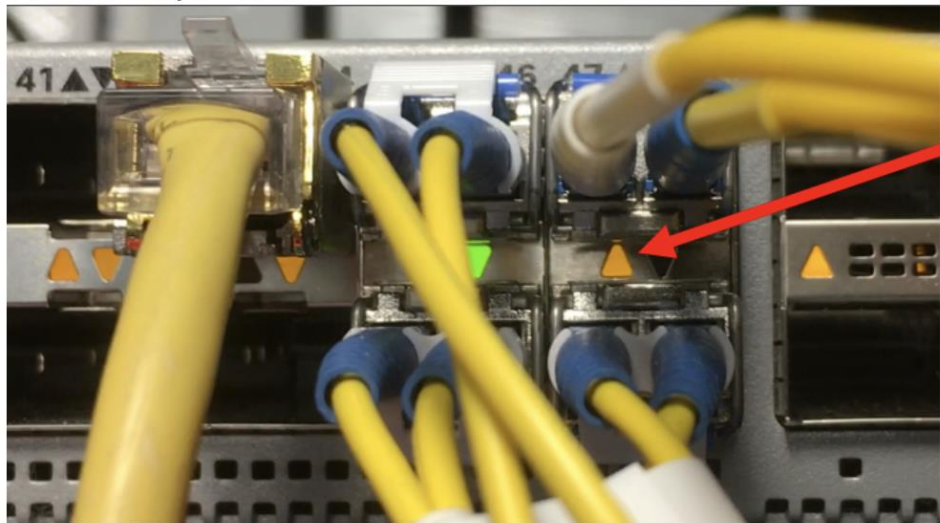


```
*Jun 29 06:14:48.768: set portid: VLAN0001 Et0/1: new port id 8002
*Jun 29 06:14:48.768: STP: VLAN0001 Et0/1 -> listening
*Jun 29 06:14:49.174: STP: VLAN0001 new root port Et0/1, cost 100
*Jun 29 06:14:49.174: STP: VLAN0001 Et0/2 -> blocking
SW3(config-if)#
*Jun 29 06:14:50.770: %LINK-3-UPDOWN: Interface Ethernet0/1, changed state t
*Jun 29 06:14:51.776: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethern
l, changed state to up
SW3(config-if)#
*Jun 29 06:15:03.773: STP: VLAN0001 Et0/1 -> learning
SW3(config-if)#
*Jun 29 06:15:18.779: STP[1]: Generating TC trap for port Ethernet0/1
*Jun 29 06:15:18.779: STP: VLAN0001 sent Topology Change Notice on Et0/1
*Jun 29 06:15:18.779: STP: VLAN0001 Et0/1 -> forwarding
```

# Spanning Tree Protocol (STP)

시스템 LED의 상태

| LED의 상태  | 시스템 상태               |
|----------|----------------------|
| 꺼짐       | 전원이 공급되고 있지 않은 꺼진 상태 |
| 초록색      | 시스템이 정상적으로 동작 중      |
| 주황색(노란색) | 전원은 공급되나 비정상적으로 작동 중 |

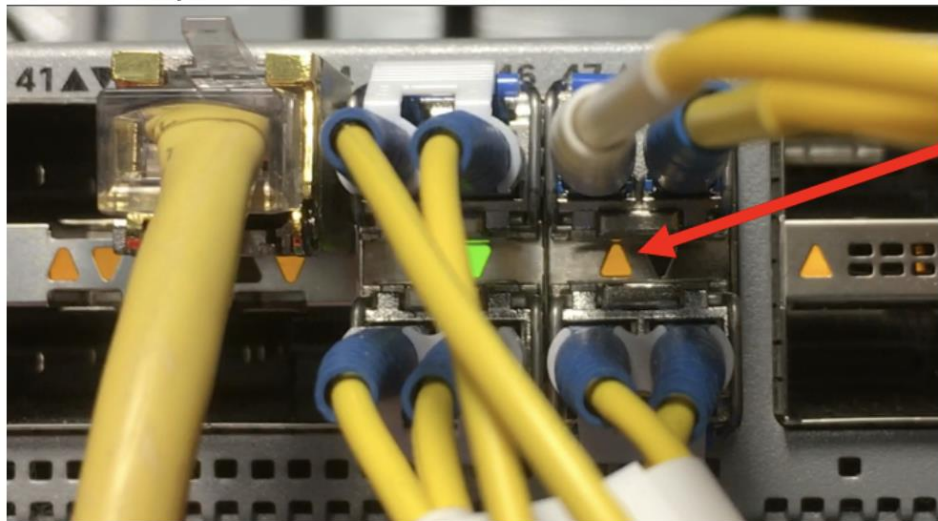


너무 느려요! 포트  
Learning 시간을  
줄일 좋은 방법이  
없을까?



인프라 운영팀

# Spanning Tree Protocol (STP)



주황색 -> 초록색 LED 변경까지 최대 50초 소요

STP 동작원리:

Listening 15초 + Learning: 15초  
= 최대 30초 소요 (직접 링크)

Blocking 20초 + Listening 15초 + Learning: 15초  
= 최대 50초 소요 (간접 링크)



# Portfast



- 사내 유저 Access Port 설정
- Portfast 설정



```
cat9kv1(config)#interface GigabitEthernet 1/0/5
```

```
cat9kv1(config-if)#spanning-tree portfast
```

```
%Warning: portfast should only be enabled on ports connected to a single  
host. Connecting hubs, concentrators, switches, bridges, etc... to this  
interface when portfast is enabled, can cause temporary bridging loops.  
Use with CAUTION
```

```
%Portfast has been configured on GigabitEthernet1/0/5 but will only  
have effect when the interface is in a non-trunking mode.
```

```
cat9kv1#show spanning-tree interface GigabitEthernet 1/0/5 detail  
Port 5 (GigabitEthernet1/0/5) of VLAN0001 is designated forwarding  
Port path cost 4, Port priority 128, Port Identifier 128.5.  
Designated root has priority 32769, address 5254.001b.293f  
Designated bridge has priority 32769, address 5254.001b.293f  
Designated port id is 128.5, designated path cost 0  
Timers: message age 0, forward delay 0, hold 0  
Number of transitions to forwarding state: 1  
The port is in the portfast mode  
Link type is point-to-point by default  
BPDU: sent 213324, received 0
```

Portfast  
설정으로  
더 이상 이슈가  
없겠지?



인프라 운영팀

# Portfast



- 신규 관리자가 Portfast 설정된 포트에 새로운 네트워크 장비 연결
- 해당 포트는 Portfast로 동작, Loop 발생으로 인해 네트워크 Down 발생

네트워크 변경으로  
새로운 스위치를  
연결해야 합니다



신규 인프라 관리자

# BPDUGuard & BPDUGuard Filtering

## 1) BPDUGuard

- BPDUGuard는 스위치 포트에 BPDUGuard를 수신하면, 포트 상태를 Err-Disable로 전환하여 스위치 포트를 비활성화 시키는 기능을 수행
- 사용
  - ① 스위치가 절대로 연결되면 안되는 구간 또는 포트
  - ② 공격자에 의해 생성된 BPDUGuard를 수신하여 스위치 부하 현상을 방지하기 위해서 사용
  - ③ 인가 받지 않은 스위치가 로컬 스위치에 연결되는 것을 방지하기 위해서 사용



## BPDUGuard

특정 포트를 통해 BPDUGuard를 수신했을 때, 해당 포트를 자동으로 shutdown 시키는 기능이다.

스위치가 아닌 장비들이 BPDUGuard를 보낼일이 없기 때문에, 일반적으로 PC나 서버와 같은 종단 장치가 접속된 포트에 설정한다.

종단장치가 접속된 포트가 BPDUGuard를 수신한다는 것은 일반 사용자들이 무단으로 스위치에 개인의 스위치나 허브를 접속했거나, STP 관련 해킹 공격을 받고 있음을 의미하는 경우가 많다.

## Configuration

@ Global configuration Mode

```
Switch(config)# spanning-tree portfast bpduguard default
```

@ Interface configuration Mode

```
Switch(config)# interface F0/13
```

```
Switch(config-if)# spanning-tree bpduguard enable
```

# Ansible을 통한 BPDU 관리 및 설정



신규 인프라 관리자

# Step 1. 장비에서 Interface 정보 가져오기

```
cat9kv1#show running-config | section ^interface
interface GigabitEthernet0/0
 vrf forwarding Mgmt-vrf
 ip address 192.168.14.14 255.255.255.0
 negotiation auto
interface GigabitEthernet1/0/1
interface GigabitEthernet1/0/2
interface GigabitEthernet1/0/3
interface GigabitEthernet1/0/4
 spanning-tree portfast
interface GigabitEthernet1/0/5
 spanning-tree portfast
 spanning-tree bpduguard enable
interface GigabitEthernet1/0/6
interface GigabitEthernet1/0/7
interface GigabitEthernet1/0/8
interface Vlan1
 no ip address
 shutdown
```

| 인터페이스                 | Portfast | BPDUGuard      |
|-----------------------|----------|----------------|
| GigabitEthernet 1/0/4 | Enable   | <u>Disable</u> |
| GigabitEthernet 1/0/5 | Enable   | Enable         |
| GigabitEthernet 1/0/6 | Enable   | <u>Disable</u> |

GigEth1/0/4,6에  
BPDU 관련 설정이  
안되어 있어서,  
스위치 연결 시  
루프가 발생 우려.



신규 인프라 관리자

→ 찾아올 데이터와 가장 근접한 command 찾기  
"show running-config | section ^ interface"

# Quick Summary: Ansible YAML 문법

Ansible Playbook의 표현방식: YAML

1. YAML 문서의 시작은 (---)로 시작한다.

---

- name: DevNet 2024 First Event!

2. 정의한 변수를 호출하는 경우, 이중 중괄호 {{ }} 사용하여 참조.

{{ Meraki }}



3. Ansible Task의 출력 결과 저장 시 "register 변수" 생성 후 debug로 출력 가능

```
tasks:
  - name: Spanning Tree Protocol # 작업의 이름을 정의합니다.
    pyats_parse_command:
      command: show running-config | section ^interface
    register: stp # 명령의 결과를 'stp' 변수에 저장합니다.
```

```
- name: 디버깅을 통한 stp 변수 출력
  debug:
    msg:
      - "{{ stp }}"
```

# Quick Summary: Ansible YAML 문법 – list

List – YAML의 배열과 동일되는 개념

1. YAML에 lists를 사용하여 여러 값으로 변수를 정의 가능. 대쉬 뒤에 공백 필요.

STP:

- Portfast
- BPDUGUARD
- BPDUFilter

2. "STP " 라는 변수를 호출하는 경우, 이중 중괄호 {{ }} 사용하여 참조.

{{ stp }} 혹은 {{ stp[0] }}과 같이 배열의 표현식도 가능

3. 위 내용을 출력하면 아래와 같습니다.

```
- name: 디버깅을 통한 stp 변수 출력
  debug:
    msg:
      - " {{ stp }}"
```



# Quick Summary: Ansible YAML 문법 –dictionary

Ansible Playbook의 표현방식인 YAML에서 사용하는 **Key/value** 쌍의 데이터 표현 형식

1. List와는 다르게, 변수를 Key-Value 형태로 저장. 콜론 뒤에 공백 필요.

Cisco:

Nexus: DataCenter

Catalyst: Campus

UCS: Server

2. Dictionary/List를 조합하여 더 복잡한 데이터 구조가 가능

- Cisco:

Location: San Jose

Skills:

- Network

- Computing





# Step 1. 장비에서 Interface 정보 가져오기

PyATS를 활용하여 데이터 가져오기

```
1  ---
2  - name: Gather interface facts from Cisco IOSXE devices
3    hosts: switches
4    gather_facts: no
5    roles:
6      - ../../roles/ansible-pyats
7    tasks:
8      - name: Spanning Tree Protocol # 작업의 이름을 정의합니다.
9        pyats_parse_command:
10          command: show running-config | section ^interface
11          register: stp # 명령의 결과를 'stp' 변수에 저장합니다.
```



Interface 관련된 running-config만을 출력, 전체 running-config 출력 시, 복잡성 증가.

## Step 2. STP 관련 설정정보 Parsing 하기

PyATS를 활용하여 데이터 가져오기

```
9 pyats_parse_command:
10     command: show running-config | section ^interface
11     register: stp # 명령의 결과를 'stp' 변수에 저장합니다.
14 - debug: # debug 모듈을 사용하여 디버깅 메시지를 출력합니다.
15     msg: #
16     - "stp: {{ stp.structured }}"
```



```
"msg": [
  "Here is your structured data: {'interfaces': {'GigabitEthernet0/0': {'vrf': 'Mgmt-vrf', 'ipv4': {'ip': '192.168
.14.14', 'netmask': '255.255.255.0'}, 'negotiation_auto': True}, 'GigabitEthernet1/0/1': {}, 'GigabitEthernet1/0/2': {},
'GigabitEthernet1/0/3': {}, 'GigabitEthernet1/0/4': {'spanning_tree_portfast': True, 'spanning_tree_bpduguard': 'disabl
e'}, 'GigabitEthernet1/0/5': {'spanning_tree_portfast': True, 'spanning_tree_bpdufilter': 'enable', 'spanning_tree_bpduguard': 'enable'}, 'GigabitEthernet1/0/6': {}, 'GigabitEthernet1/0/7': {}, 'GigabitEthernet1/0/8': {}, 'Vlan1': {'shutdow
n': True}}}"
```

- Portfast = true, bpdugard = disable인 인터페이스를 찾아서 출력!

## Step 2. STP 관련 설정정보 Parsing 하기

PyATS를 활용하여 데이터 가져오기

```
9 pyats_parse_command:
10     command: show running-config | section ^interface
11     register: stp # 명령의 결과를 'stp' 변수에 저장합니다.
14 - debug: # debug 모듈을 사용하여 디버깅 메시지를 출력합니다.
15     msg: #
16     - "stp: {{ stp.structured }}"
```



```
"msg": [
  "Here is your structured data: {'interfaces': {'GigabitEthernet0/0': {'vrf': 'Mgmt-vrf', 'ipv4': {'ip': '192.168
.14.14', 'netmask': '255.255.255.0'}, 'negotiation_auto': True}, 'GigabitEthernet1/0/1': {}, 'GigabitEthernet1/0/2': {},
'GigabitEthernet1/0/3': {}, 'GigabitEthernet1/0/4': {'spanning_tree_portfast': True, 'spanning_tree_bpduguard': 'disabl
e'}, 'GigabitEthernet1/0/5': {'spanning_tree_portfast': True, 'spanning_tree_bpdufilter': 'enable', 'spanning_tree_bpduguard': 'enable'}, 'GigabitEthernet1/0/6': {}, 'GigabitEthernet1/0/7': {}, 'GigabitEthernet1/0/8': {}, 'Vlan1': {'shutdow
n': True}}}"
```

- Portfast = true, bpdugard = disable인 인터페이스를 찾아서 출력!

# Quick Summary: Ansible 문법 – dict2items

Key-Value 형태로 저장되는 객체인 Dictionary를 개별적인 형태로 사용할 수 있는 item으로 변환시켜주는 키워드

1. 만약 변수가 아래와 같이 있다고 가정합니다.

```
dict_variable : {'key1': 'value1', 'key2': 'value2'}
```

2. 사용은 아래와 같이 playbook에 선언 가능합니다.

```
dict_variable | dict2items
```

3. 위 내용을 출력하면 아래와 같습니다.

```
[{'key': 'key1', 'value': 'value1'}, {'key': 'key2', 'value': 'value2'}]
```



## Step 2. STP 관련 설정정보 Parsing 하기

dict2items로 키워드 데이터 만들기

```
{{ stp.structured }}
```

```
"{{ stp.structured.interfaces | dict2items }}"
```

```
TASK [pyATS를 통해 구조화된 데이터 출력] *****
*****
*****ok: [cat9k-1] => {
  "msg": [
    {
      "interfaces": {
        'GigabitEthernet0/0': {'vrf': 'Mgmt-vrf', 'ipv4': {'ip': '192.168.14.14', 'netmask': '255.255.255.0'}, 'negotiation_auto': True},
        'GigabitEthernet1/0/1': {}, 'GigabitEthernet1/0/2': {}, 'GigabitEthernet1/0/3': {}, 'GigabitEthernet1/0/4': {'spanning_tree_portfast': True, 'spanning_tree_bpduguard': 'disable'}, 'GigabitEthernet1/0/5': {'spanning_tree_portfast': True, 'spanning_tree_bpdufilter': 'enable', 'spanning_tree_bpduguard': 'enable'}, 'GigabitEthernet1/0/6': {'spanning_tree_portfast': True, 'spanning_tree_bpduguard': 'disable'}, 'GigabitEthernet1/0/7': {}, 'GigabitEthernet1/0/8': {}, 'Vlan1': {'shutdown': True}}}}
  ]
}
```

```
TASK [dict2items 활용 출력] *****
*****
{
  "key": "GigabitEthernet1/0/4",
  "value": {
    "spanning_tree_bpduguard": "disable",
    "spanning_tree_portfast": true
  },
  "key": "GigabitEthernet1/0/5",
  "value": {
    "spanning_tree_bpdufilter": "enable",
    "spanning_tree_bpduguard": "enable",
    "spanning_tree_portfast": true
  },
  "key": "GigabitEthernet1/0/6",
  "value": {
    "spanning_tree_bpduguard": "disable",
    "spanning_tree_portfast": true
  }
}
```

- Dict2items 키워드 활용, 데이터를 Parsing 가능한 형태로 변형

## • Step 3. Portfast 인터페이스 리스트 출력

Running-config에서 Portfast 설정된 인터페이스 이름 리스트를 출력합니다.

- name: portfast enable된 인터페이스의 이름 출력하기

debug:

```
msg: "{{ stp.structured.interfaces | dict2items | json_query('[?value.spanning_tree_portfast==`true`'].{interfacename: key}') }}"
```



```
TASK [portfast enable된 인터페이스의 이름 출력하기] ***
ok: [cat9k-1] => {
  "msg": [
    {
      "interfacename": "GigabitEthernet1/0/4"
    },
    {
      "interfacename": "GigabitEthernet1/0/5"
    },
    {
      "interfacename": "GigabitEthernet1/0/6"
    }
  ]
}
```

## Step 4. Portfast 인터페이스 리스트와 설정 출력

Running-config에서 Portfast 설정된 인터페이스 이름과 Running-config 리스트를 출력합니다

name: portfast enable된 인터페이스의 이름과 러닝컨피그 함께 출력하기

debug:

```
msg: "{ { stp.structured.interfaces dict2items | json_query('[?value.spanning_tree_portfast==`true`]', {interfacename: key, configcontext: value})
```

```
TASK [portfast enable된 인터페이스의 이름과 러닝컨피그 함께 출력하기]
ok: [cat9k-1] => {
  "msg": [
    {
      "configcontext": {
        "spanning_tree_bpduguard": "disable",
        "spanning_tree_portfast": true
      },
      "interfacename": "GigabitEthernet1/0/4"
    },
    {
      "configcontext": {
        "spanning_tree_bpdufilter": "enable",
        "spanning_tree_bpduguard": "enable",
        "spanning_tree_portfast": true
      },
      "interfacename": "GigabitEthernet1/0/5"
    },
    {
      "configcontext": {
        "spanning_tree_bpduguard": "disable",
        "spanning_tree_portfast": true
      },
      "interfacename": "GigabitEthernet1/0/6"
    }
  ]
}
```


## Step 5. Portfast & BPDU Guard Disable 된 인터페이스 리스트와 설정 출력

Running-config에서 Portfast 설정된 인터페이스 이름과 Running-config 리스트를 출력합니다

```
name: portfast enable && bpduguard disable list 출력
```

```
debug:
```

```
msg: "{{ stp.structured.interfaces | dict2items | json_query(['?value.spanning_tree_portfast==`true` && value.spanning_tree_bpduguard==`disable`'].{interface: key, value: value}) }}"
```

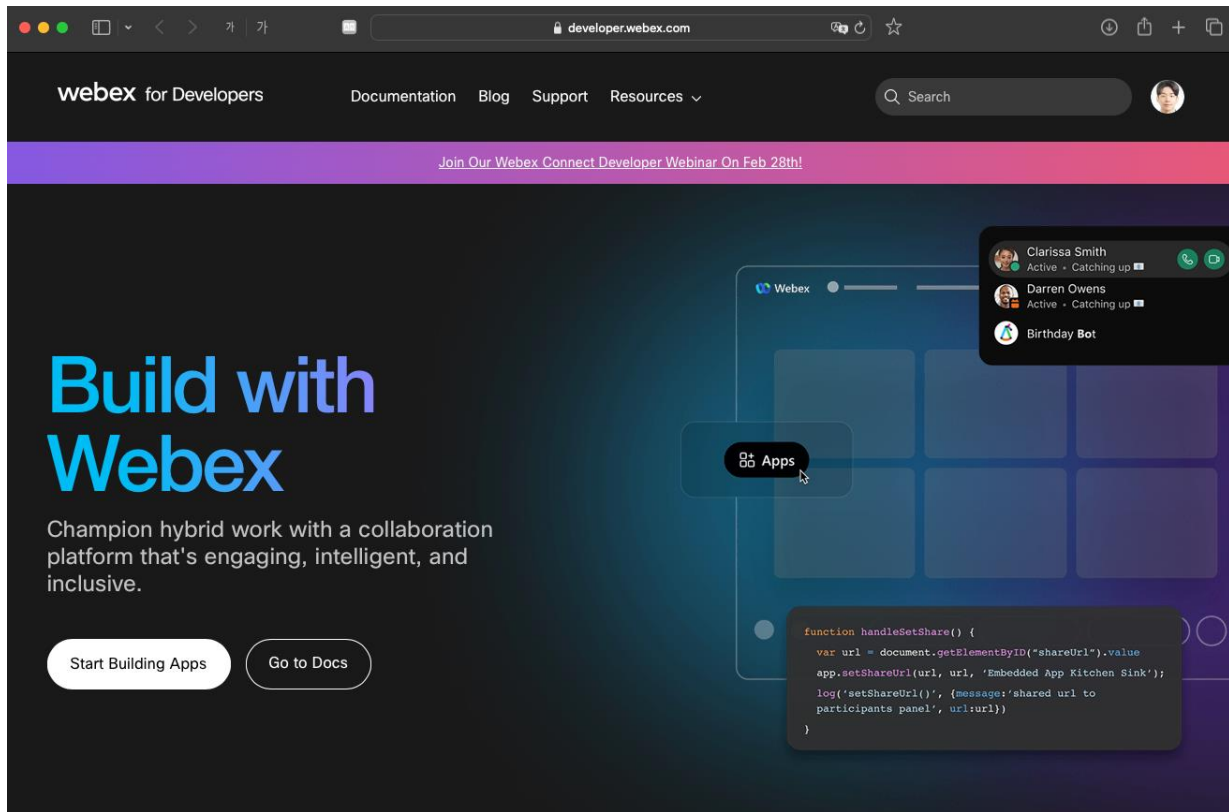


```
TASK [portfast enable && bpduguard disable list 출력] *
ok: [cat9k-1] => {
  "msg": [
    {
      "interface": "GigabitEthernet1/0/4",
      "value": {
        "spanning_tree_bpduguard": "disable",
        "spanning_tree_portfast": true
      }
    },
    {
      "interface": "GigabitEthernet1/0/6",
      "value": {
        "spanning_tree_bpduguard": "disable",
        "spanning_tree_portfast": true
      }
    }
  ]
}
```



# Step 6. Webex API 연동 – Step 1. Webex 로그인

<https://developer.webex.com> 회원가입 및 로그인 진행



# Step 6. Webex API 연동 – Step 2. 계정 Token 복사

<https://developer.webex.com/docs/getting-started - accounts-and-authentication> 로그인 후 Bearer Token 복사

webex for Developers

Documentation Blog Support Resources

Q Search

FAQs

APIs

API Behavior Changes

Partners API Guide

XML API Deprecation

Access the API

REST API Basics

Compliance

Webhooks

Webex APIs

+ Admin

+ Webex Calling Beta

+ Webex Calling

Accounts and Authentication

What's possible with the Webex APIs?

Accounts and Authentication

Personal Access Tokens

Methods & Content Types

Next Steps

Support Policy

Getting Help

## Access the Webex API

The Webex APIs give you easy access to the Webex Platform to build [Bots](#), [Integrations](#), or [Guest Issuer](#) apps. If you're ready to start using the Webex APIs, keep reading.

### Your Personal Access Token

Bearer \*\*\*\*\*

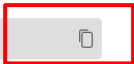
*This limited-duration personal access token is hidden for your security.*

What's possible with the Webex APIs?

The Webex APIs provide your applications with direct access to the Cisco Webex Platform, giving you the ability to:

- [Create a Webex space and invite people](#)
- [Search for people in your company](#)
- [Post messages in a Webex space](#)
- [Get Webex space history](#) or be [notified in real-time](#) when new messages are posted by others
- [Execute a command on a Webex RoomOS device](#)
- and much more!

클릭



# Step 6. Webex API – Webex Room list API 실행

<https://developer.webex.com/docs/api/v1/rooms/list-rooms> 에 접속 후 GET Method 실행 후 Room list 획득

The screenshot displays the Webex Developer Portal interface. On the left, a sidebar lists various API categories like Recordings, Report Templates, Reports, and Rooms. The 'Rooms' category is selected, showing a list of endpoints including 'GET List Rooms'. The main content area provides detailed information about the 'List Rooms' endpoint, including its purpose, default behavior, and known limitations. Below this, the 'Query Parameters' section lists parameters such as teamId, type, orgPublicSpaces, from, and max. On the right, a configuration panel allows users to set these parameters and authorize the request. A red box highlights the 'Run' button at the bottom of this panel.

**webex for Developers** Documentation Blog Support Resources

Search

Recordings  
Report Templates  
Reports  
Reports: Detailed Call History  
Resource Group Memberships  
Resource Groups  
Roles  
Room Tabs  
Rooms

**GET** List Rooms  
**POST** Create a Room  
**GET** Get Room Details  
**GET** Get Room Meeting Details  
**PUT** Update a Room  
**DELETE** Delete a Room

SCIM 2 Bulk  
SCIM 2 Groups  
SCIM 2 Users  
Security Audit Events  
Session Types  
Site

### List Rooms

List rooms.

The title of the room for 1:1 rooms will be the display name of the other person. When a Compliance Officer lists 1:1 rooms, the "other" person cannot be determined. This means that the room's title may not be filled in. Please use the [memberships API](#) to list the people in the space.

By default, lists rooms to which the authenticated user belongs.

Long result sets will be split into [pages](#).

Known Limitations: The underlying database does not support natural sorting by `lastactivity` and will only sort on limited set of results, which are pulled from the database in order of `roomId`. For users or bots in more than 3000 spaces this can result in anomalies such as spaces that have had recent activity not being returned in the results when sorting by `lastactivity`.

**GET** /v1/rooms

#### Query Parameters

**teamId**  
string  
List rooms associated with a team, by ID. Cannot be set in combination with `orgPublicSpaces`.

**type**  
string  
List rooms by type. Cannot be set in combination with `orgPublicSpaces`.  
Possible values: `direct`, `group`

**orgPublicSpaces**  
boolean  
Shows the org's public spaces joined and unjoined. When set the result list is sorted by the `madePublic` timestamp.

**from**  
string  
Filters rooms, that were made public after this time. See `madePublic` timestamp

**Header**

Authorization ☒ Use personal access token

**Bearer** \*\*\*\*\*  
*This limited-duration personal access token is hidden for your security.*

#### Query Parameters

**teamId** e.g. Y2IzY29zcGFyaXovL3VzL1JPT00v

**type** e.g. group

**orgPublicSpaces** e.g. true

**from** e.g. 2022-10-10T17:00:00.000Z

**to** e.g. 2022-10-11T17:00:00.000Z

**sortBy** e.g. id

**max** e.g. 100

**Run**

클릭 후 Room의 ID를 복사 후 저장

## Step 6. Webex API – 메시지 연동

```
33 - name: Cisco Webex Teams - Text Message to a Room
34   community.general.cisco_webex: # community.general.cisco_webex 모듈을 사용합니다.
35     recipient_type: roomId # 수신자 유형은 roomId로 설정합니다.
36     recipient_id: "Y2lzY29zcGFyazovL3VzL1JPT00vZWY0MmE0MTAtN2NkZS0xMWVlLTg3MGQtMDM0MTYwZmMyMmEx" # room ID로 설정.
37     msg_type: text # 메시지 유형을 text로 설정합니다.
38     personal_token: "NjhiMTU0YzQtYWU3Ni00NDc0LTljOGQtZDNjOGM2OWVhYmZkM2QxZjA2ZTctMzNh_Pf84_1eb65fdf-9643-417f-9974-ad72cae0e10f" #본인의 Bearer Token을 설정합니다.
39     msg:
40       - "Portfast가 설정된 interface 리스트 입니다"
41       - "{{ stp.structured.interfaces | dict2items | json_query('[?value.spanning_tree_portfast==`true`'].{interfacename: key}')}}"
42
```

코드 주석 해제 후, 토큰과 Room ID를 코드에 입력

# Step 6. Webex API – Webex 연동 결과

코드 수정 후 `ansible-playbook main.yml -i hosts` 실행

```
devnet@devnet-virtual-machine: ~/DevNet_Korea/03_Reporting/STP$ ansible-playbook main.yml -i hosts_
```



본인 오후 8:01

['Portfast가 설정된 interface 리스트 입니다', [{'interfacename': 'GigabitEthernet1/0/4'}, {'interfacename': 'GigabitEthernet1/0/5'}, {'interfacename': 'GigabitEthernet1/0/6'}]]

['Portfast가 설정된 interface 리스트와 컨피그 내용입니다', [{'interfacename': 'GigabitEthernet1/0/4', 'configcontext': {'spanning\_tree\_portfast': True, 'spanning\_tree\_bpduguard': 'disable'}}, {'interfacename': 'GigabitEthernet1/0/5', 'configcontext': {'spanning\_tree\_portfast': True, 'spanning\_tree\_bpdufilter': 'enable', 'spanning\_tree\_bpduguard': 'enable'}}, {'interfacename': 'GigabitEthernet1/0/6', 'configcontext': {'spanning\_tree\_portfast': True, 'spanning\_tree\_bpduguard': 'disable'}}]]

['Portfast가 설정되어있지만, BPDUguard가 설정되어있지 않아, 설정이 필요한 인터페이스 리스트입니다.', [{'interface': 'GigabitEthernet1/0/4', 'value': {'spanning\_tree\_portfast': True, 'spanning\_tree\_bpduguard': 'disable'}}, {'interface': 'GigabitEthernet1/0/6', 'value': {'spanning\_tree\_portfast': True, 'spanning\_tree\_bpduguard': 'disable'}}]]



Shift + Enter를 눌러 새로운 줄 생성

The Mosaic에 메시지 쓰기



# Mission Time!

# Mission 1:

## 이 미션을 위한 폴더 접속 방법

```
devnet@devnet-virtual-machine:~$ cd DevNet_Korea/03_Reporting/STP  
devnet@devnet-virtual-machine:~/DevNet_Korea/03_Reporting/STP$ ansible-playbook main.yml -i hosts
```

1. `cd DevNet_Korea/03_Reporting/STP`
2. `Ansible-playbook main.yml -i hosts`

# Mission 1:

## 코드 수정하여 Webex로 메시지 보내기

- Main.yml을 수정하여 미션을 수행합니다.
- Bpdufilter가 설정된 인터페이스를 Parsing하여 Webex 메시지로 전송  
정답 예시: ["Bpdufilter가 설정된 interface 리스트 입니다", [{"interfacename": "GigabitEthernet1/0/5"}]]

- Hint:

BPDUfilter의 데이터 형식: spanning\_tree\_bpdufilter



# Agenda

- Jinja2 템플릿을 통한 CSV 리포트 생성
  - Mission #2

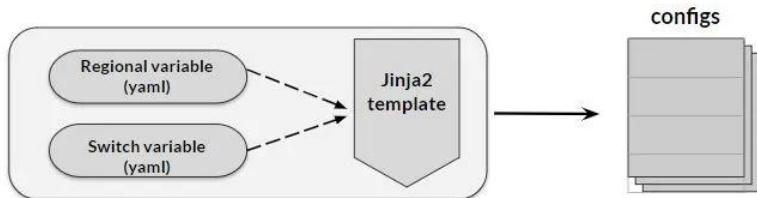
수집한 데이터들을만  
정형화된 파일로만  
들 수 있을까?



인프라 운영팀

# Why Jinja2 Template in Ansible?

## Ansible Configuration Template



Playbook-YAML에서  
정의된 변수들을  
새롭게 정의하고  
정렬하는데 용이

```
- name: Generate CSV file
  template:
    src: templates/interface_info.j2
    dest: interface_info.csv
```

# Ansible Jinja2 템플릿을 사용한 모니터링 리포트 생성 과정

1

## 템플릿 작성

Ansible과 Jinja2를 활용하여 리포트 템플릿을 작성합니다.

2

## 데이터 수집

넥서스 스위치에서 필요한 데이터를 **Playbook**을 통해 수집하여 템플릿에 적용합니다.

3

## CSV 리포팅

완성된 리포트를 생성하여 관리자나 팀원들에게 제공합니다.

# Jinja2 기초

- Jinja2 Template이란?
- 파이썬의 **j2 (jinja2)** 를 통해 앤서블의 템플릿을 만듦 (파이썬에서 템플릿을 위해 정의된 엔진)
- 확장자는 .j2
- 일반 텍스트 선언 가능, csv 헤더 작성 시 각 열의 이름을 쉼표로 구분하여 작성.
- 변수는 {{ }} 와 같이 중괄호 사용을 통해 선언
- YAML Playbook에 선언된 변수명을 j2 템플릿에서 그대로 사용 가능

```
admin_state,oper_status,link_state,description
{{ interface_info.admin_state }},{{ interface_info.oper_status }},{{ interface_info.link_state }},
{{ interface_info.description }}
```

# Jinja2 기초 - 반복문

- Jinja2 기본 반복문 구조

```
{% 반복문 %}  
반복할 변수 입력  
{% endfor %}
```

예시:

```
name,age                #CSV의 헤더가 될 부분입니다.  
{% for person in people %}    #people list의 각 항목에 대해 1행 작성  
{{ person.name }},{{ person.age }}  #각 변수는 쉼표로 구분합니다  
{% endfor %}                #반복문을 마치는 부분에 입력합니다.
```

```
- name: Generate CSV file  
  template:  
    src: templates/interface_info.j2  
    dest: interface_info.csv
```

J2파일 작성  
후, dest를  
csv파일로  
입력시 CSV  
파일 생성

# Jinja2 기초 – interface\_info.j2

```
ansible > pyATS+Ansible > templates > 🏠 interface_info.j2
1  admin_state,oper_status,link_state,description
2  {{ interface_info.admin_state }},
3  {{ interface_info.oper_status }},
4  {{ interface_info.link_state }},
5  {{ interface_info.description }}
```

# Jinja2 기초 – interface\_info.j2

```
92 - name: Interface information
93   pyats_parse_command:
94     command: show interface
95     register: output # 'output' 변수에 'show interface' 명령의 결과를 저장합니다.
96     # 'Ethernet1/1' 인터페이스의 정보를 'interface_info' 변수에 저장합니다.
97 - name: Save interface information to a variable (1개 인터페이스)
98   set_fact:
99     interface_info: "{{ output.structured['Ethernet1/1'] | to_nxos1 }}"
```

```
TASK [debug] *****
*****
```

```
ok: [nxos1] => {
  "interface_info": {
    "admin_state": "down",
    "auto_mdix": "on",
    "auto_negotiate": true,
    "bandwidth": 10000000,
    "beacon": "off",
    "counters": {
      "in_bad_etype_drop": 0,
      "in_broadcast_pkts": 0,
      "in_crc_errors": 0,
      "in_discard": 0,
      "in_errors": 0,
      "in_if_down_drop": 0,
      "in_ignored": 0,
```

# Mission Time!



## Mission 2:

### 이 미션을 위한 폴더 접속 방법

```
devnet@devnet-virtual-machine:~$ cd DevNet_Korea/03_Reporting/  
devnet@devnet-virtual-machine:~/DevNet_Korea/03_Reporting$ ansible-playbook main.yml -i hosts_
```

1. `cd DevNet_Korea/03_Reporting`
2. `Ansible-playbook main.yml -i hosts`

## Mission 2:

### 코드 수정하여 CSV 파일 만들기

- DevNet\_Korea/03\_Reporting/templates/interface\_info.j2을 수정하여 미션을 수행합니다
- Interface\_info의 bandwidth를 추가하는 내용을 Jinja2 템플릿에 추가합니다.

*정답 예시:* Interface\_info.csv의 내용을 확인합니다.

```
devnet@devnet-virtual-machine:~/DevNet_Korea/03_Reporting/templates$ cat interface_info.csv
admin_state,oper_status,link_state,description,bandwidth
up,up,up,devnet-ansible-event, 1000000
```

- Hint:

Bandwidth의 형식: interface\_info.bandwidth

# 네트워크 점검 템플릿 소개 – multi\_interface\_info.j2

ansible > pyATS+Ansible > templates > multi\_interface\_info.j2

```
/ansible [Interface,admin_state,oper_status,link_state,mac_address,in_crc_errors
2  {% for item in range(1, 65) %}
3  Ethernet1/{{ item }},{{ output.structured['Ethernet1/' + item|string].admin_state }},
4  {{ output.structured['Ethernet1/' + item|string].oper_status }},
5  {{ output.structured['Ethernet1/' + item|string].link_state }},
6  {{ output.structured['Ethernet1/' + item|string].mac_address }},
7  {{ output.structured['Ethernet1/' + item|string].counters.in_crc_errors }}
8  {% endfor %}
9
10  Hostname: {{ version.structured.platform.hardware.device_name }}
11  Version: {{ version.structured.platform.software.system_version }}
12  Model: {{ version.structured.platform.hardware.model }}
13  OS: {{ version.structured.platform.os }}
14
15  CPU Usage (last 5 minutes): {{ cpu.structured.five_min_cpu }}%
16  Memory Total: {{ memory.structured.memory_usage.memory_usage_total_kb }}KB
17  Available Memory: {{ memory.structured.memory_usage.memory_usage_free_kb }}KB
18  Unavailable Memory: {{ memory.structured.memory_usage.memory_usage_used_kb }}KB
19
20  Redundancy State: {{ redundancy.structured.redundancy_mode.operational }}
21  Active Supervisor Time: {{ redundancy.structured.active_supervisor_time }}
22
23  Routing total routes: {{ route.structured.vrf.default.total_routes }}
```

# 네트워크 점검 템플릿 소개 – multi\_interface\_info.csv

```
root@22131cc9e8e1:/ansible/pyATS+Ansible# cat multi_interface_info.csv
Interface,admin_state,oper_status,link_state,mac_address,in_crc_errors
Ethernet1/1,down,down,down,bcd2.95a4.bc67,0
Ethernet1/2,down,down,down,bcd2.95a4.bc67,0
Ethernet1/3,down,down,down,bcd2.95a4.bc67,0
Ethernet1/4,down,down,down,bcd2.95a4.bc67,0
Ethernet1/5,down,down,down,bcd2.95a4.bc67,0
Ethernet1/6,down,down,down,bcd2.95a4.bc67,0
Ethernet1/7,down,down,down,bcd2.95a4.bc67,0
Ethernet1/8,down,down,down,bcd2.95a4.bc67,0
Ethernet1/9,down,down,down,bcd2.95a4.bc67,0
Ethernet1/10,down,down,down,bcd2.95a4.bc67,0
Ethernet1/11,down,down,down,bcd2.95a4.bc67,0
Ethernet1/12,down,down,down,bcd2.95a4.bc67,0
Ethernet1/13,down,down,down,bcd2.95a4.bc67,0
Ethernet1/14,down,down,down,bcd2.95a4.bc67,0
Ethernet1/15,down,down,down,bcd2.95a4.bc67,0
Ethernet1/16,down,down,down,bcd2.95a4.bc67,0
Ethernet1/17,down,down,down,bcd2.95a4.bc67,0
Ethernet1/18,down,down,down,bcd2.95a4.bc67,0
```

Hostname: switch

Version: 9.3(5)

Model: Nexus9000 C93108TC-EX

OS: NX-OS

CPU Usage (last 5 minutes): 5%

Memory Total: 24631956KB

Available Memory: 18486848KB

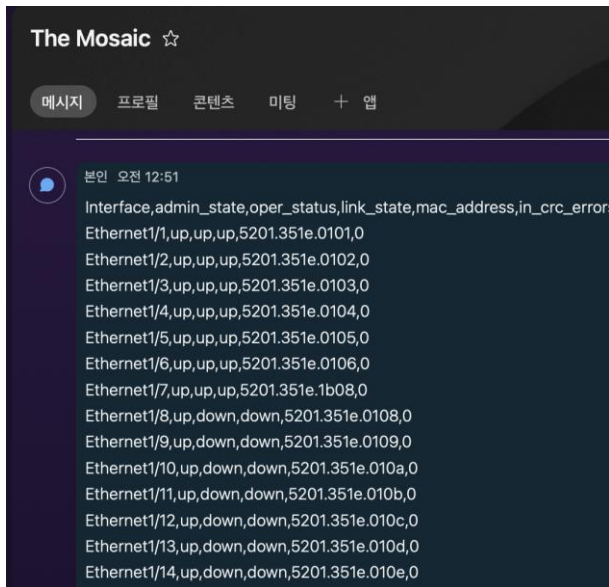
Unavailable Memory: 6145108KB

Redundancy State: None

Active Supervisor Time: 1 days, 17 hours, 57 minutes, 40 seconds

Routing total routes: 4

# 네트워크 점검 템플릿 소개 – multi\_interface\_info.csv to Webex



```
Hostname: N9Kv1
Version: 10.3(1) [Feature Release]
Model: Nexus9000 C9300v
OS: NX-OS

CPU Usage (last 5 minutes): 17%
Memory Total: 8135680KB
Available Memory: 2726912KB
Unavailable Memory: 5408768KB

Redundancy State: None
Active Supervisor Time: 6 days, 8 hours, 47 minutes, 10 seconds

Routing total routes: 3

VPC Peer Status: peer adjacency formed ok
VPC Keepalive Status: peer is alive
VPC Type 2 Consistency Status: success
```



Thank you

