

A10 Thunderforof AnsibleOf modulebookKikata

A10Networks Co., Ltd.

Solution Architect Kentaro Ishizuka

Ansible Night in Osaka

2018Year11Month28Day



Reliable Security Always
TM

CONFIDENTIAL | DO NOT DISTRIBUTE

A10 Networks Do you know?



A10 Networks, Inc. (NYSE: ATEN)
 Established: September 2004
 Representative: Lee Chen (CEO and Founder) Headquartered in San Jose, California, USA

A10 Network's IPO Ceremony Company

- Setting Standing: 2009 Year Four Month

Global company from Silicon Valley

Country of location

27

number of employees

900+ (Japan: 60+)

Adopted by world leaders

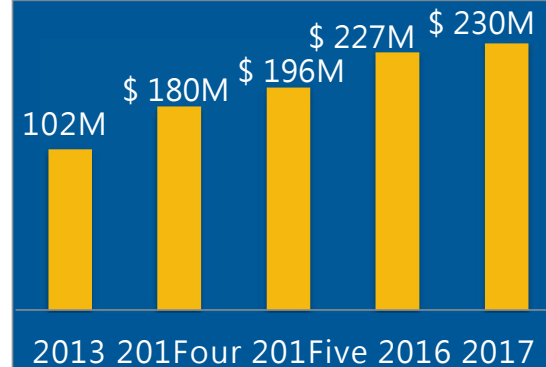
Product / support destination country

80+

Number of companies

6,200+

Stable growth



Product group by function



aGalaxy
(TPS Management of)



Harmony Controller
(Multi cloud, Multi-service
tigerFWWhKuVisibleConversionAnd tubeReason)

Security



DDoS defense
(Thunder
TPS)



TLS Communication
is possible
SightConversion
(Thunder
SSLi)



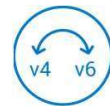
Centralized FW
(Thunder
CFW)



traffic control
(Thunder ADC,
Lightning ADC)

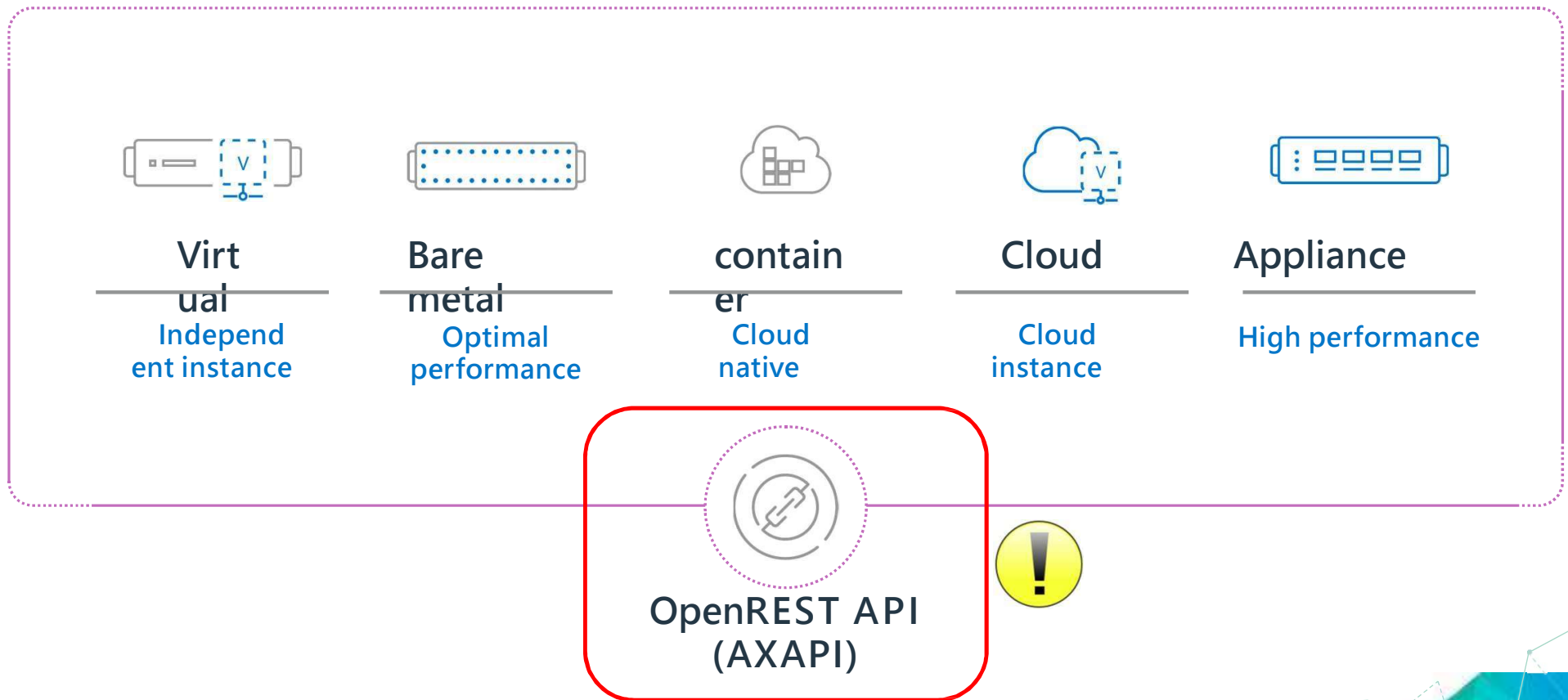


application distribution



Your career-DoNAT
(Thunder
CGN)

Flexible choice: Form factor

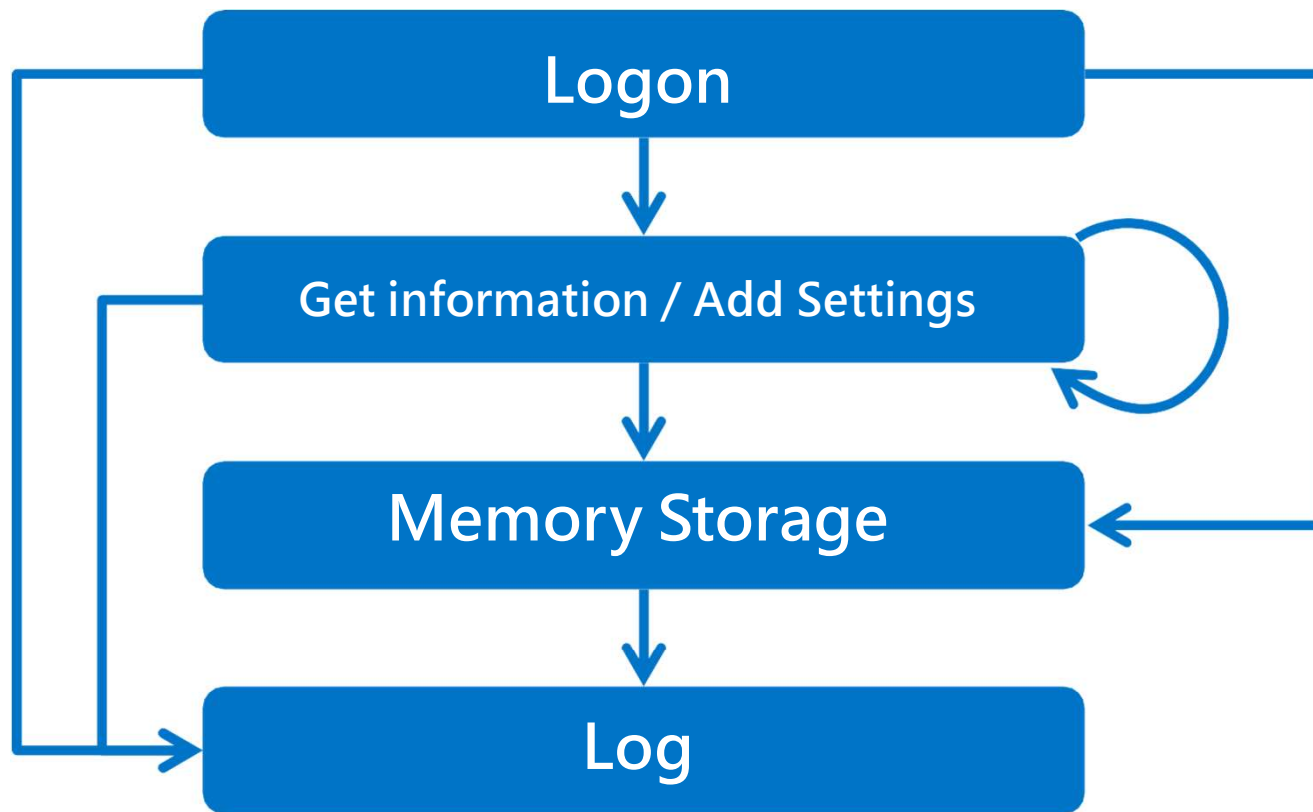


What is aXPI??

- REST API for acquiring and changing thunder information and settings Items that can be set in the CLI can be almost aXAPI
- Currently there are two main versions v2.1: ACOS 2.7.2 series v3.0: ACOS 4.x series
-



aXPI flow



Logon

【request】

```
curl -X POST --K
https://10.255.250.105/axapi/v3/auth
-H "Content-Type: application / json"
-d '{
    "credential": {
        "username": "admin",
        "password": "a10"
    }
}
```

With credentials
mgmtLog on to port

【response】

```
{
  "authresponse": {
    "signature": "6141de07ade039e1ef6010a6297c18",
    "description": "the signature should be set in
    Authorization
    header for following request. "
  }
}
```

GetsignatureUse
information in future
requests

Setting information acquisition

【request】

```
curl -k https://10.255.250.105/axapi/v3/slb
-H "Content-Type: application / json"
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18"
```

Got at logonsignatureSend the request with in the header

【response】

```
{
  "slb": {
    "common": {
      "extended-stats":
        0,
      "stats-data-disable": 0,
      "graceful-shutdown-enable":
        0,
    }
    ...
    "a10-url": "/ axapi / v3 / slb / service-
    group / sg80"
```

The specific of the object being setURIAlso get

Get a list of setting information according to request

Add setting

【request】

```
curl -X POST -k https://10.255.250.105/axapi/v3/slb/server  
-H "Content-Type: application / json"  
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18 "  
-d '{"server-list": [{"name": "svr3", "host":  
"192.168.1.3"}]}'
```

To the attribute to add the
setting POST ↑

Object to add JSON Described by

【response】

```
{  
  "server": {  
    "name":  
    "192.168.1.3",  
    ...  
    "action": "enable",  
    "extended-stats": 0,  
    "uuid": "432b96fc-c3b5-11e7-a858-  
  } 000c29395c26"  
}
```

Information of added
object is returned

Delete settings

【request】

```
curl -X DELETE -k https://10.255.250.105/axapi/v3/ s1b / server / svr3  
-H "Content-Type: application / json"  
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18 "
```

DELETE Specify the object to delete
with

【UUID When specifying DStrike】

```
curl -X DELETE -k https://10.255.250.105/axapi/v3/ uuid / 52f7e62a-c3b6-11e7-a858-000c29395c26  
-H "Content-Type: application / json"  
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18 "
```

Object UUID You can also specify and delete

【response】

```
{  
  "response": {  
    "status": "OK"  
  }  
}
```

If successful "OK" Returns

Save memory

【request】

```
curl -k -X POST https://10.255.250.105/axapi/v3/write /  
-H "Content-Type: application / json"  
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18"
```

POSTso/ write /
memorySpecify↑

【response】

```
{  
  "response": {  
    "status": "OK"  
  }  
}
```

} If successful "OK" Returns

log off

【request】

```
curl -X POST -k https://10.255.250.105/axapi/v3/logout  
-H "Content-Type: application / json"  
-H "Authorization: A10 6141de07ade039e1ef6010a6297c18"  
"
```

logoutSpecifyPOST

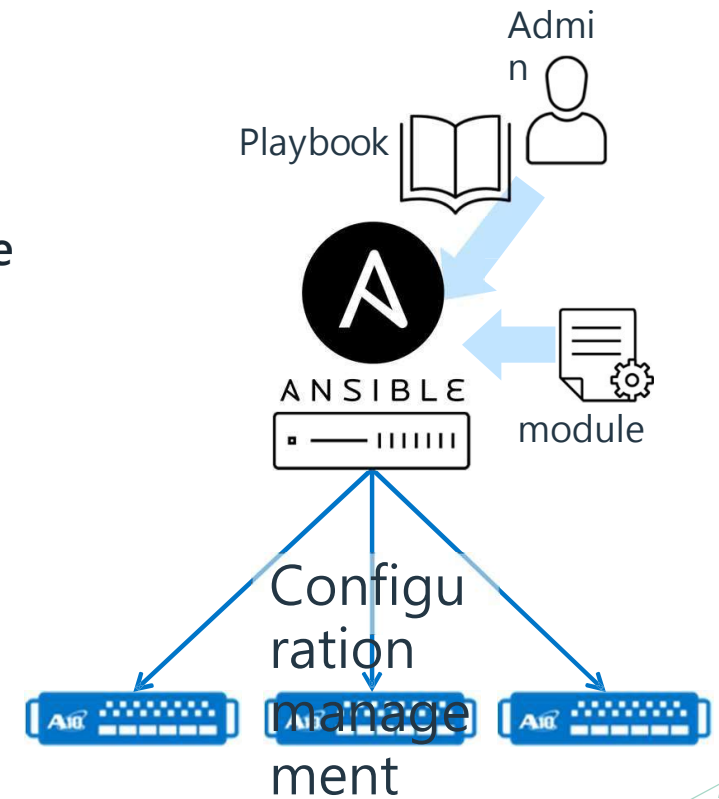
【response】

```
{  
  "response": {  
    "status": "OK"  
  }  
}
```

If successful "OK" is returned,
signature cannot be reused

AnsiblebyThunderConfiguration management

- AnsibleToYoRuThunderConfiguration oftubeMethod
 - Method①: a 1 0 _ * Using modulesPlaybookDescribe
 - Method②: u r i Using the modulePlaybookAt the inneraXAPIRun



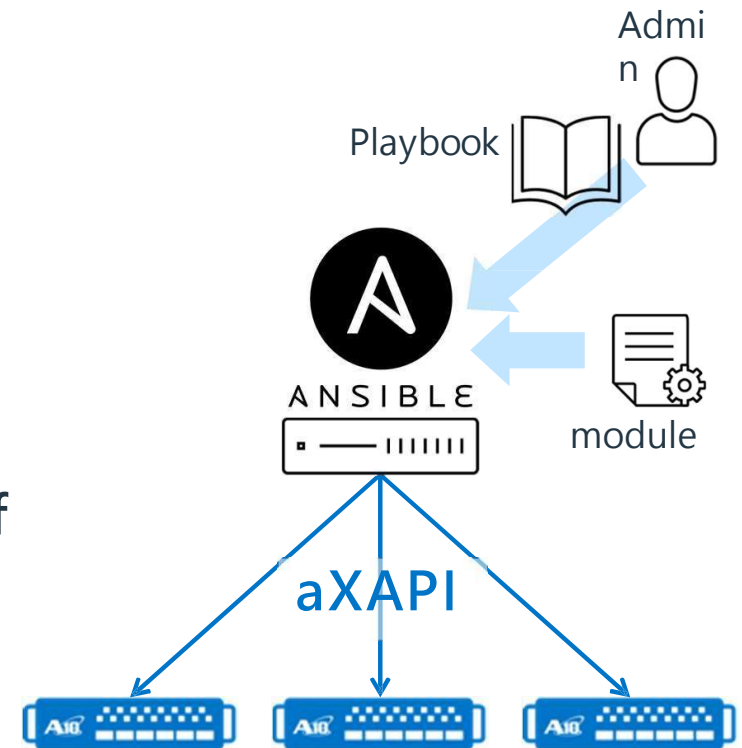
a10_* Basic operation when using a module

- A10abovePythonThere is no execution environment (It is easier to understand) For,

PracticallyAnsible CoreFrom a running server
ThunderAgainstaXAPIOperationProductRun

- PlaybookProcessing onversusUnderstand the elephantOrScoopMeTo ThunderHost ofI P ToIf you specifyGoodI

- connection: localSpecify the localLesoplaceReasonlineDo



A10ofAnsiblemodule

- officialGitHubIn the beta barTheRelease
<https://github.com/a10networks/a10-ansible>
- 1,453Module of
- 2,568ofPlaybookAn example
- Precautions when using
 - PythonofRequestsMoTheThelsMustEssential
 - In moduleofpartToRepairRuMustEssentialButAhIfButis there
 - The above moduleLelsaXAPIv3versuselephant(ACOS 4.xSinceDescending)
 - Guarantee of idempotencyIs part of Only/ Dry RunNot compatible/ PartitionOraVCSToNot yet versusResponse
- Automaticallywrite memoryNot done

But I want to be like this

- I want to ensure idempotency
- ListThe information given inReasonWant to
- Dry RunWant to support
- Partition (Virtual partition)OraVCS (VirtualShiTo)versusI want to respond
- Different in the same moduleaXAPIBa-Treat JohnIWant
- Write memoryI want to be able to specify whether to do



To write a module?

- First, the modules around here import

```
from ansible.module_utils.basic import AnsibleModule
from ansible.module_utils.urls import url_argument_spec, fetch_url
from ansible.module_utils.a10 import axapi_call_v3, axapi_call,
axapi_authenticate_v3, axapi_authenticate, axapi_failure
```

Basically,

- `AnsibleModule`
- `axapi_call_v3, axapi_call`
- `axapi_authenticate_v3, axapi_authenticate`
- `axapi_failure`

use

Basic process

```
def main ():
    module = AnsibleModule (
        argument_spec =get_argspec (),
        supports_check_mode =True,
        mutually_exclusive
        =MUTUALLY_EXCLUSIVE_SET
    )
    if module.check_mode:
        result = dry_run_command
(module) else:
        result = run_command (module)

    module.exit_json (**
```

} Dry
runCorresp
ondence

```
result) MUTUALLY_EXCLUSIVE_SET
```

```
=
```

```
[['tagged_eth_list', 'tagged_trunk_list', 'untagged_eth_list',  
'untagged_trunk_list', 'untagged_lif']]
```

19 CONFIDENTIAL - NO FORN DISCLOSURE

PlaybookRead arguments from(VLANSetting example)

```
def get_argspec ():
    rv = get_default_argspec
    () rv.update (      ↳ Read common
                        settings
                        dict(
                            name = dict (type = 'str', required = False), user_tag
                                = dict (type = 'str', required = False), vlan_num =
                                dict (type = 'int', required = False), ve = dict (type
                                = 'int', required = False), tagged_eth_list = dict
                                (type = 'list', required = False), untagged_eth_list =
                                dict (type = 'list', required = False),
...
                                untagged_lif = dict (type = 'int', required =
                                False)
                        )
    )
    return rv
```

Reading commonly used arguments

```
def get_default_argspec
(): rv = dict (
    a10_host = dict (type = 'str', required = True),
    a10_username = dict (type = 'str', required = True),
    a10_password = dict (type = 'str', required = True, no_log
= True), axapi_version = dict (type = 'str', required =
False, default = '3',
choises = ['2.1', '3']),
    partition = dict (type = 'str', required = False, default =
'shared'),
    device = dict (type = 'str', required = False), write_config
= dict (type = 'bool', required = False, default = 'yes',
choises = ['yes', 'no']),
    state = dict (type = 'str', required = True, choises =
['present', 'absent', 'current', 'statistics', 'operational'])
)
rv.update (url_argument_spec
()) return rv
```

PlaybookCommon setting items in

- `a10_host`: Be targeted Thunder / vThunder of mgmtPort address
- `a10_username`: Admin user account for the target device
- `a10_password`: Admin user password
- `validate_certs`: aXAPI Whether to validate the validity of the certificate of the port (Yes or no)
- `axapi_version`: aXAPI Version of (2.1 Or 3)
- `write_config`: Whether to save the config after setting (Yes or no)
- `partition`: Partition to be set (ADP) Designation of
- `state`: present (Existence), absent (absence)
- `device`: aVCS of device ID (device-context)

Command execution process

```
def run_command (module):  
...  
    signature = axapi_open_session (module) aXAPI Signature of  
...  
    if module.params ['partition']:  
        change_partition (module, signature) Move partition  
    if module.params ['device']:  
        change_device_context (module, signature) aVCS Moving devices  
    if module.params ['state'] == 'present':  
        result = present (module, signature, result)  
    elif module.params ['state'] == 'absent':  
        result = absent (module, signature, result) } Add / delete  
settings  
...  
    if write_config:  
        write_memory (module, signature) Save memory  
    axapi_close_session (module, signature) aXAPI Log off  
    return result
```

Get signature

```
def axapi_open_session
(module):
    if module.params ['axapi_version'] ==
        '3': axapi_auth_url =
'https: // {} /axapi/v3/auth/'.format (module.params ['
    a10_host ']) rv = axapi_authenticate_v3 (module,
module.params ['a10_username'], module.params ['a10_password'])
    elif module.params ['axapi_version'] ==
        '2.1': axapi_auth_url =
'https: // {} /services/rest/V2.1/'.format (module.params ['
    a10_host ']) rv = axapi_authenticate (module,
module.params ['a10_username'], module.params ['a10_password'])
    if axapi_failure (rv):
        module.fail_json (msg = "Failed to open aXAPI session ")
    return rv
```

aXAPIv3In
Signature

aXAPIv2.1In
Signature

Move partition

```
def change_partition (module, signature):  
  
    if module.params ['axapi_version'] ==  
        '3': axapi_base_url =  
'https: // {} /axapi/v3/'.format (module.params [' a10_host  
        ']) result = axapi_call_v3 (module, axapi_base_url +  
        'active-  
signature != signature)params [' partition', method = 'POST',  
body = '',  
    if axapi_failure  
    (result, axapi_close_session (module, signature)  
        module.fail_json (msg = "Failed to change  
partition. ")
```

Execute command to move partition

aVCSMoving devices

For commands that have no return value axapi_call () Does not move, so individual processing

```
def change_device_context (module, signature):  
  
    if module.params ['axapi_version'] == '3':  
        json_post = {"device-context": {"device-id":  
module.params ['device']}}  
        axapi_base_url = 'https: // {}  
/axapi/v3/'.format (module.params [' a10_host '])  
        headers = {'content-type': 'application / json',  
'Authorization': 'A10 % s' % signature}  
        rsp, info = fetch_url (module, axapi_base_url + 'device-  
context', method = 'POST', data = json.dumps (json_post), headers =  
headers)  
        if not rsp or info ['status'] >= 400: module.fail_json (msg  
            = "failed to connect (status code % s),  
error was% s " % (info ['status'], info.get (' msg ', ' no error given  
')))  
        rsp.close ()
```

Change settings(VLAN(For setting))

```
def present (module, signature, result):  
    differences, config_before, json_post = diff_config  
(module, signature, result, status = 'present')
```

With existing configurationPlaybookModule to

compare configurations of

```
    if differences: axapi_base_url = 'https: // {} /axapi/v3/'.format  
        (host) result_list = axapi_call_v3 (module,  
axapi_base_url + 'network / vlan /', method =  
POST=, json.dumps (json_post), signature = signature)  
        if axapi_failure (result_list):  
            axapi_close_session (module,  
signature)  
            module.fail_json (msg = "Failed to create  
VLAN. ") else:  
                result ["changed"] = True
```

JSONIf there is a
diff, send the content of
POST

```
    return result
```

Save memory

```
def write_memory (module, signature):  
  
    if module.params ['a10_host'] ==  
        '3': axapi_base_url =  
'https: // {} /axapi/v3/'.format (module.params [' a10_host '])  
        result = axapi_call_v3 (module, axapi_base_url + 'write /  
memory /', method = 'POST', body = '', signature = signature)  
  
    if axapi_failure (result): axapi_close_session  
        (module, signature) module.fail_json (msg =  
            "Failed to write config. ")
```

aXAPI Log off

```
def axapi_close_session (module, signature):  
  
    if module.params ['axapi_version'] ==  
        '3': axapi_logoff_url =  
'https: // {} /axapi/v3/logoff/'.format (module.params [' a10_host '])  
        result = axapi_call_v3 (module, axapi_logoff_url, method =  
'POST', body = '', signature = signature)  
    elif module.params ['axapi_version'] == '2.1':  
        axapi_logoff_url = signature + '& method =  
        session.close' result = axapi_call (module,  
        axapi_logoff_url)  
  
    if axapi_failure (result):  
        module.fail_json (msg = "Failed to close aXAPI session. ")
```

Dry run in the case of

```
def dry_run_command (module):  
...  
    differences, config_before, json_post = diff_config  
(module, signature, result, status = state)  
...  
    result ['diff'] ['before'] = config_before  
...  
    if state == 'present':  
...  
        result ['changed'] = True result  
        ['diff'] ['after'] = json_post  
...  
    return result
```

Playbook Before
and after execution
JSON The difference
between ['diff']
['before']
['diff'] ['after'] Put
in

PlaybookDescription example

```
---
- hosts: 10.255.211.25
  connection: local
  gather_facts: no

tasks:
- name: Set GSLB site
  a10_gslb_site:
    a10_host: 10.255.211.25
    a10_username: admin
    a10_password: a10
    validate_certs: no
    axapi_version: 3
    partition: test
    device: 1
    write_config: yes
    state: present
    site_name: local
    active_rdt:
      limit: 16300
      ignore-count: 6
    multiple_geo_locations:
      - geo-location: US
    - geo-location: JP
    slb_dev_list:
      - device-name: A1
        ip-address: 1.0.0.1
        vip-server:
          vip-server-name-list:
            - vip-name: VIP-HTTP
      - device-name: A3
        ip-address: 1.0.0.3
        vip-server:
          vip-server-name-list:
            - vip-name: VIP-HTTP3
    register: test_reg
- name: show JSON
  debug: msg = "{{ test_reg
    }}"
```

Specify the target host, connection: local Because it is not possible to specify the execution locally and to obtain information in advance by specifying gather_facts: no To

Specify the module to use (in this example, a10_gslb_site module)

Setting items common to all modules

Key-value Type of TermEye setting

dictType item

settings

listType item

setting is hierarchical listType-dictType item settings

Display of return value (if necessary)

Ensuring idempotency

```
# ansible-playbook test_network_vlan.yml -i hosts

PLAY [10.255.211.25] *****

TASK [Set VLAN] *****
changed: [10.255.211.25]

PLAY RECAP *****
10.255.211.25      : ok = 1  changed = 1  unreachable = 0 failed = 0

# ansible-playbook test_network_vlan.yml -i hosts

PLAY [10.255.211.25] *****

TASK [Set VLAN] *****
OK: [10.255.211.25]

PLAY RECAP *****
10.255.211.25      : ok = 1  changed = 0 unreachable = 0 failed = 0
```

Dry run (checkMode) and diff Response to


```
# ansible-playbook test_network_vlan.yml -i hosts --check --diff
PLAY [10.255.211.25] *****

***** TASK [Set VLAN] *****

*****
--- before
+++ after
@@ -1,7 +1,16 @@
 {
+   "vlanName": "testnamex",
+   "api": "/v3 / network / vlan / 110",
+   {
+     "untagged-ethernet-end": 7,
+     "untagged-ethernet-start": 7
+   }
+   ],
+   "user-tag": "jkl",
+   "uuid": "861056a6-8e5d-11e8-9be5-ed1e39a22969",
+   "ve": 110,
+   "vlan-num": 110
 }

changed: [10.255.211.25]

PLAY RECAP *****
***** 10.255.211.25 : ok = 1 changed = 1
unreachable = 0 failed = 0
```


An aerial photograph of the Tokyo skyline, featuring the Tokyo Tower prominently on the left. The image is overlaid with a semi-transparent blue geometric pattern of interconnected lines and polygons. The sky is filled with white clouds. The text "Thank you you" is written in white on the right side of the image.

Thank you
you