Intro to Ansible Automation Lab

Documentation for F5’s released Ansible modules can be found [here](http://docs.ansible.com/ansible/latest/list_of_network_modules.html#f5).

Ansible utilizes YAML files to automate configurations of devices.

Variables can be of following types, alpha-numeric, boolean, lists and dictionaries.

Variables can be defined with-in playbooks, in external files or passed on the command line with the -e argument.

Examples of variables:

Scalars:

name: Bob

vs\_port: 80

useSSL: yes

provisionASM: false

Array/List

pools: [ ‘app1’, ‘app2’]

virtuals: ['app1\_vs','app2\_vs']

Hash/Dictionary

pools:

* { name: app1, lbMethod: ‘round-robin’, svcDownAction: reset, monitor: http }
* { name: app2, lbMethod: ‘ratio’, svcDownAction: none, monitor: tcp }

Run a playbook referencing a variable on commandline

ansible-playbook <playbookName> -e hosts=bigip-a

### Prerequisite for lab

Install Ansible on Linux JumpHost

1. Click on **Root Terminal** to open Linux terminal session
2. Run apt-get update

**apt-get update**

1. Install Python Package Manger Pip

**apt-get install python-pip**

1. Install the ansible, f5-sdk, bigsuds and deepdiff python modules

**pip install ansible==2.4.2 f5-sdk==3.0.9 bigsuds deepdiff**

1. Create ansible folders

**mkdir ansible**

**mkdir ansible/playbooks**

**mkdir ansible/tmp**

|  |
| --- |
| root@LinuxJumphost:~# pwd  /home/ubuntu  root@LinuxJumphost:~# mkdir ansible  root@LinuxJumphost:~# mkdir ansible/playbooks  root@LinuxJumphost:~# mkdir ansible/tmp  root@LinuxJumphost:~# vim ansible/hosts |

1. Create ansible config file (ansible/ansible.cfg)

**vim ansible/ansible.cfg**

|  |
| --- |
| [defaults]  inventory = ./hosts  roles\_path = .playbooks/roles  library = ./library/  retry\_files\_enabled = False  host\_key\_checking = False  remote\_tmp = /tmp/  [ssh\_connection]  ssh\_executable = /usr/bin/ssh  control\_path\_dir=/tmp/ |

1. Create inventor file (ansible/hosts)

**vim ansible/hosts**

|  |
| --- |
| [bigip-a]  10.1.1.10  [bigip-b]  10.1.1.11  [bigip-ab]  10.1.1.10  10.1.1.11 |

1. Change directory to ansible

**cd ansible**

## Lab1 – Create a node

This lab will create playbook which will have a single task that creates a node ‘Server1’ on IP 10.1.10.100 .utilizing the [bigip\_node](http://docs.ansible.com/ansible/latest/bigip_node_module.html)  ansible module.

Sample playbook:

|  |
| --- |
| ---  - name: Lab1 - Create a node  hosts: bigip-a  connection: local  tasks:  - name: Create node using bigip\_node module  bigip\_node:  server: "{{ inventory\_hostname }}"  user: admin  password: admin  validate\_certs: no  address: "10.1.10.100"  name: "Server1"  delegate\_to: localhost |

Steps:

1. Open editor and create lab1.yml in the playbooks directory

**vim playbooks/lab1.yml**

1. Copy and paste contents of the sample playbook above into editor
2. Save file and exit
3. Check syntax using –syntax-check argument

**ansible-playbook playbooks/lab1.yml --syntax-check**

1. Run playbook

**ansible-playbook playbooks/lab1.yml**

1. Verify new node Server1 was created on [BIG-IP](https://10.1.1.10/) (Local Traffic -> Nodes).

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab1.yml --syntax-check  playbook: playbooks/lab1.yml  root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab1.yml  PLAY [Lab1 Create a node] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create a node using bigip\_node module] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |

## Lab 2 – Using internal variables

Variables can be utilized with-in modules parameters using the "{{ <variable name> }}" format.

The sample playbook from the previous lab uses the inventory\_hostname variable for the server parameter.

Existing Lab1 playbook:

|  |
| --- |
| ---  - name: Lab1 - Create a node  hosts: bigip-a  connection: local  tasks:  - name: Create node using bigip\_node module  bigip\_node:  server: "{{ inventory\_hostname }}"  user: admin  password: admin  validate\_certs: no  address: "10.1.10.100"  name: "Server1"  delegate\_to: localhost |

Create variables for user, password parameters.

Use the playbook created for lab1 (lab1.yml).

Steps:

1. Open lab1.yml
2. Modify lab1 playbook as shown in the sample playbook below

Sample playbook:

|  |
| --- |
| ---  - name: Lab2 - Delete a node using internal variables  hosts: bigip-a  connection: local  vars:  user: admin  password: admin  tasks:  - name: Delete node using bigip\_node module  bigip\_node:  server: "{{ inventory\_hostname }}"  user: "{{ user }}"  password: "{{ password }}"  validate\_certs: no  state: absent  name: "Server1"  delegate\_to: localhost |

1. Save playbook as **playbooks/lab2.yml**
2. Verify you have valid syntax using the –syntax-check argument (*should see error as shown below*)

ansible-playbook playbooks/lab2.yml --syntax-check

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab2.yml --syntax-check  ERROR! Syntax Error while loading YAML.  The error appears to have been in '/home/ubuntu/ansible/playbooks/lab2.yml': line 21, column 6, but may  be elsewhere in the file depending on the exact syntax problem.  The offending line appears to be:  name: "Server1"  delegate\_to: localhost  ^ here  exception type: <class 'yaml.parser.ParserError'>  exception: while parsing a block collection  in "<unicode string>", line 13, column 5  did not find expected '-' indicator  in "<unicode string>", line 21, column 6 |
|  |

1. Modify the lab2.yml playbook and remove the extra space from in front of delegate\_to on line 21.
2. Verify the syntax of the playbook (**ansible-playbook playbooks/lab2.yml --syntax-check**)
3. Run playbook (**ansible-playbook playbooks/lab2.yml**)

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab2.yml  PLAY [Lab2 - Delete a node using internal variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10]  TASK [Delete node using bigip\_node module] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=2 changed=1 unreachable=0 failed=0 |

1. Let's re-create Server1
2. Copy lab2 to lab2\_1.yml

cp playbooks/lab2.yml playbooks/lab2\_1.yml

1. Modify the lab2\_1.yml playbook so it matches the one in the table below

|  |
| --- |
| ---  - name: Lab2 - Create a node using internal variables  hosts: bigip-a  connection: local  vars:  user: admin  password: admin  tasks:  - name: Create node using bigip\_node module  bigip\_node:  server: "{{ inventory\_hostname }}"  user: "{{ user }}"  password: "{{ password }}"  validate\_certs: no  name: "Server1"  address: "10.1.10.100"  state: present  delegate\_to: localhost |

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab2.yml  PLAY [Lab2 - Create node using internal variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create node] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |

## Lab 3 – Looping

with\_items provides a mechanism for ansible to perform a task with multiple input values. In this lab we will utilize [with\_items](http://docs.ansible.com/ansible/latest/playbooks_loops.html#standard-loops) to add multiple nodes with-in a single task.

|  |
| --- |
| ---  - name: Lab 3 - Create nodes using bigip\_node module and with\_items  hosts: bigip-a  connection: local  gather\_facts: no  vars:  user: admin  password: admin  servers:  - { name: Server1, address: "10.1.10.100", sessionState: "enabled" }  - { name: Server2, address: "10.1.10.101", sessionState: "disabled" }  tasks:  - name: Create nodes using with\_items  bigip\_node:  user: "{{ user }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  session\_state: "{{ item.sessionState }}"  validate\_certs: no  name: "{{ item.name }}"  address: "{{ item.address }}"  delegate\_to: localhost  with\_items: "{{ servers }}" |

Steps:

1. Open lab2.yml with your editor.
2. Modify lab2 playbook as shown in the sample playbook above
3. Save the modified playbook as **playbooks/lab3.yml**
4. Verify the syntax of the playbook (**ansible-playbook playbooks/lab3.yml --syntax-check**)
5. Run playbook (**ansible-playbook playbooks/lab3.yml**)

The output should look similar to the below table entry

|  |
| --- |
| ansible-playbook lab3.yml  PLAY [Lab 3 - Create nodes using bigip\_node module and with\_items] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create nodes using with\_items] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost] => (item={u'sessionState': u'enabled', u'name': u'Server1', u'address': u'10.1.10.100'})  changed: [10.1.1.10 -> localhost] => (item={u'sessionState': u'disabled', u'name': u'Server2', u'address': u'10.1.10.101'})  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=2 unreachable=0 failed=0 |

1. Extra Credit: Create a lab3\_1.yml file from your existing lab3.yml file to create nodes

(server3 -10.1.10.102, and server4 - 10.1.10.103)

## Lab 4 – Vault

Ansible Vault is an ansible feature that allows sensitive data such as user credentials to be encrypted. Variables can be encrypted in place and decryption will happen at runtime. The vault password can be passed as an argument

Steps:

1. Create password variable file (**playbooks/password.yml**)
2. Add entries for userName and password variables as shown below

|  |
| --- |
| ---  userName: admin  password: admin |

1. To encrypt the password file use ansible-vault as shown below

|  |
| --- |
| **ansible-vault encrypt playbooks/password.yml --ask-vault-pass**  New Vault password:  Confirm New Vault password:  Encryption successful |

1. Verify password file is encrypted

|  |
| --- |
| **cat playbooks/password.yml**  $ANSIBLE\_VAULT;1.1;AES256  383835383665343531383037366635656439396462623062353666313162386432316237663833663938663261333862323934383434323461616639653965630a393239363432343831363561373932363462636231663161323737393465336465646435336634653063303162633037323265656661663461383263376432620a383830613462613939646239613563346462623332623466303934663162346162373134613461336638303366363562343937333532613633613537303231623538316266373736646234366538646530306464333331323637636132333034 |

1. Copy lab3.yml and save it as lab4.yml
2. Modify lab4.yml as shown below

|  |
| --- |
| ---  - name: Lab 4 - Create nodes using bigip\_node module and with\_items and encrypted variables  connection: local  gather\_facts: no  hosts: bigip-a  vars:  servers:  - { name: Server1, address: "10.1.10.100", sessionState: "enabled" }  - { name: Server2, address: "10.1.10.101", sessionState: "disabled" }  vars\_files:  - "password.yml"  tasks:  - name: Create nodes using with\_items  bigip\_node:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  session\_state: "{{ item.sessionState }}"  validate\_certs: no  name: "{{ item.name }}"  address: "{{ item.address }}"  delegate\_to: localhost  with\_items: "{{ servers }}" |

1. To run the playbook and decrypt the password file at runtime, add --ask-vault-pass argument as shown below

|  |
| --- |
| **ansible-playbook lab4.yml --ask-vault-pass**  Vault password:  PLAY [Lab 4 - Create nodes using bigip\_node module and with\_items and encrypted variables] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create nodes using with\_items] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost] => (item={u'sessionState': u'enabled', u'name': u'server1', u'address': u'10.1.10.100'})  changed: [10.1.1.10 -> localhost] => (item={u'sessionState': u'disabled', u'name': u'server2', u'address': u'10.1.10.101'})  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |

1. Vault password can also be stored in a file and reference to this file can be added to the ansible.cfg file. This will allow for use of encrypted variables without user interaction.
2. Create vault password file

vim .vault\_file

1. Enter the password you gave to the password.yml file and save the file
2. Edit the ansible.cfg file and add the following line after remote\_tmp in the [defaults] section

|  |
| --- |
| vault\_password\_file **= .vault\_file** |

1. Re-run the lab4 playbook

**ansible-playbook lab4.yml**

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab4.yml  PLAY [Lab 4 - Create nodes using bigip\_node module and with\_items and encrypted variables] \*\*\*  TASK [Create nodes using with\_items] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost] => (item={u'sessionState': u'enabled', u'name': u'Server1', u'address': u'10.1.10.100'})  changed: [10.1.1.10 -> localhost] => (item={u'sessionState': u'disabled', u'name': u'Server2', u'address': u'10.1.10.101'})  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |

## Lab 5 – Creating application stack

This lab will utilize the bigip\_node, bigip\_pool, bigip\_pool\_members, bigip\_irule and bigip\_virtual\_server modules.

1. Create **maint\_pg.irule** in **ansible/tmp** directory

|  |
| --- |
| when HTTP\_REQUEST {  if {[active\_members [LB::server pool] ] < 1 } {  HTTP::respond 200 content "<html> <title>Maintenance Page</title><body> We are currently down for maintenance. Please try back later </body></html>"  }  } |

1. Create lab5.yml and enter/paste contents of the following table into it.

|  |
| --- |
| ---  - name: Lab 5 - Create application  connection: local  gather\_facts: no  hosts: bigip-a  vars:  servers:  - { name: Server1, address: "10.1.10.100", sessionState: "enabled" }  - { name: Server2, address: "10.1.10.101", sessionState: "disabled" }  userName: admin  password: admin  tasks:  - name: Create nodes using with\_items  bigip\_node:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  session\_state: "{{ item.sessionState }}"  validate\_certs: no  name: "{{ item.name }}"  address: "{{ item.address }}"  delegate\_to: localhost  with\_items: "{{ servers }}"  - name: Create pool  bigip\_pool:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  validate\_certs: no  name: red\_pool  delegate\_to: localhost  - name: Create pool members  bigip\_pool\_member:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  validate\_certs: no  pool: red\_pool  address: "{{ item.name }}"  port: 80  delegate\_to: localhost  with\_items: "{{ servers }}"  - name: Install maintenance page iRule  bigip\_irule:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  validate\_certs: no  module: ltm  name: maint\_pg\_irule  content: "{{ lookup('file', '/home/ubuntu/ansible/tmp/maint\_pg.irule') }}"  delegate\_to: localhost    - name: Create virtual server  bigip\_virtual\_server:  server: "{{inventory\_hostname}}"  user: "{{ userName }}"  password: "{{ password }}"  validate\_certs: no  partition: Common  name: red\_vs  destination: "{{ vsAddr }}"  port: "{{ vsPort }}"  all\_profiles:  - http  all\_rules:  - maint\_pg\_irule  delegate\_to: localhost |

1. Verify the syntax before running the lab5 playbook
2. Run the playbook

|  |
| --- |
| root@LinuxJumphost:~/ansible# **ansible-playbook playbooks/lab5.yml**  PLAY [Lab 5 - Create application] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create nodes using with\_items] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost] => (item={u'address': u'10.1.10.100', u'name': u'Server1', u'sessionState': u'enabled'})  changed: [10.1.1.10 -> localhost] => (item={u'address': u'10.1.10.101', u'name': u'Server2', u'sessionState': u'disabled'})  TASK [Create pool] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost]  TASK [Create pool members] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost] => (item={u'address': u'10.1.10.100', u'name': u'Server1', u'sessionState': u'enabled'})  ok: [10.1.1.10 -> localhost] => (item={u'address': u'10.1.10.101', u'name': u'Server2', u'sessionState': u'disabled'})  TASK [Install maintenance page iRule] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost]  TASK [Create virtual server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=5 changed=1 unreachable=0 failed=0 |

## Lab5.1 - Disable pool member

1. Create playbooks/lab5\_1.yml and paste in the contents of table below

|  |
| --- |
| ---  - name: Lab 5.1 - Toggle pool member session state  connection: local  gather\_facts: no  hosts: bigip-a  vars\_files:  - "password.yml"  tasks:  - name: Toggle pool member state  bigip\_pool\_member:  user: "{{ userName }}"  password: "{{ password }}"  server: "{{ inventory\_hostname }}"  validate\_certs: no  pool: "{{ poolName }}"  address: "{{ serverName }}"  port: "{{ serverPort }}"  session\_state: "{{ sessionState }}"  delegate\_to: localhost |

1. Run playbook and provide extra variables for poolName, serverName, serverPort and sessionState on the commandline

**ansible-playbook playbooks/lab5\_1.yml -e "poolName=red\_pool serverName=Server1 serverPort=80 sessionState=disabled"**

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab5\_1.yml -e "poolName=red\_pool serverName=Server1 serverPort=80 sessionState=disabled"  PLAY [Lab 5.1 - Toggle pool member session state] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Toggle pool member state] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |

1. Verify the pool member session state was toggled as expected

## Lab 6 – Declarative interface using bigip\_iapp\_services

1. Install latest HTTP iApp
2. Create lab6.yml and paste contents from the table below. This playbook will install the latest HTTP iApp template on BIG-IP device.
3. Verify your playbook syntax

**ansible-playbook playbooks/lab6.yml --syntax-check**

1. Run lab6 playbook to install the HTTP iApp template

**ansible-playbook playbooks/lab6.yml**

|  |
| --- |
| ---  - name: Install latest HTTP iApp template  hosts: bigip-a  connection: local  gather\_facts: no  vars:  userName: admin  password: admin  tasks:  - name: Install HTTP iApp  bigip\_iapp\_template:  content: "{{ lookup('file', '/home/ubuntu/ansible/tmp/f5.http.v1.2.0.tmpl') }}"  user: "{{ userName }}"  server: "{{ inventory\_hostname }}"  state: present  password: "{{ password }}"  validate\_certs: no  delegate\_to: localhost |

1. We will utilize the bigip\_iapp\_service module to create an iApp based application
2. Create **lab6\_1.yml** and copy the configuration from the table below. This will create an application blue\_vs and create two virtual servers blue\_vs and blue\_redir\_vs.

|  |
| --- |
| ---  - name: Create HTTP application using the install HTTP iApp  hosts: bigip-a  connection: local  gather\_facts: no  vars:  userName: admin  password: admin  templateName: "f5.http.v1.2.0"  vsName: blue\_vs  vsAddr: 10.1.20.121  appMonitor: http\_head\_f5  lbMethod: observed-member  sslKey: "default.key"  sslCert: "default.crt"  servicePort: "8002"  node1: Server1  node2: Server2  tasks:  - name: Create HTTP Application  bigip\_iapp\_service:  user: "{{ userName }}"  server: "{{ inventory\_hostname }}"  state: present  password: "{{ password }}"  validate\_certs: no  name: "{{ vsName }}"  template: "{{ templateName }}"  parameters:  variables:  - name: client\_\_http\_compression  encrypted: 'no'  value: "/#create\_new#"  - name: monitor\_\_monitor  encrypted: 'no'  value: "{{ appMonitor }}"  - name: net\_\_client\_mode  encrypted: 'no'  value: wan  - name: net\_\_server\_mode  encrypted: 'no'  value: lan  - name: pool\_\_addr  encrypted: 'no'  value: "{{ vsAddr }}"  - name: pool\_\_lb\_method  encrypted: 'no'  value: "{{ lbMethod }}"  - name: pool\_\_pool\_to\_use  encrypted: 'no'  value: "/#create\_new#"  - name: pool\_\_port\_secure  encrypted: 'no'  value: "443"  - name: ssl\_\_cert  encrypted: 'no'  value: "{{ sslCert }}"  - name: ssl\_\_key  encrypted: 'no'  value: "{{ sslKey }}"  - name: ssl\_\_mode  encrypted: 'no'  value: "client\_ssl"  - name: ssl\_\_server\_ssl\_profile  encrypted: 'no'  value: "/#default#"  - name: ssl\_encryption\_questions\_\_advanced  encrypted: 'no'  value: 'no'  - name: ssl\_encryption\_questions\_\_help  encrypted: 'no'  value: hide  tables:  - name: basic\_\_snatpool\_members  - name: net\_\_snatpool\_members  - name: optimizations\_\_hosts  - name: pool\_\_hosts  columnNames:  - name  rows:  - row:  - "{{ vsName }}"  - name: pool\_\_members  columnNames:  - addr  - port  - connection\_limit  rows:  - row:  - "{{ node1 }}"  - "{{ servicePort }}"  - "0"  - row:  - "{{ node2 }}"  - "{{ servicePort }}"  - "0"  - name: server\_pools\_\_servers  delegate\_to: localhost |

1. Verify the syntax before running the playbook

**ansible-playbook playbooks/lab6\_1.yml --syntax-check**

1. Run the playbook

**ansible-playbook playbooks/lab6\_1.yml**

|  |
| --- |
| root@LinuxJumphost:~/ansible# ansible-playbook playbooks/lab6\_1.yml  PLAY [Create HTTP application using the install HTTP iApp] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Create HTTP Application] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [10.1.1.10 -> localhost]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  10.1.1.10 : ok=1 changed=1 unreachable=0 failed=0 |