redhat.

ANSIBLE

Introduction à Ansible - Gestion F5

Michael Lessard
Architecte principal de solutions
mlessard@redhat.com
michaellessard

AVERTISSEMENT

CECI EST UNE FORMATION D'INTRODUCTION GRATUITE OFFERTE PAR RED HAT

ELLE N'A AUCUN LIEN AVEC NOTRE GROUPE GLS

CET ATELIER A ÉTÉ ÉLABORÉ PAR QUELQUES ENTHOUSIASTES ARCHITECTES DE SOLUTIONS AU CANADA

redhat.

ORDRE DU JOUR

Formation Ansible F5

1 **Introduction à Ansible**

2 **Commande Ansible**
+ LABO

3 **Playbook Ansible**
+ LABO

4 **Playbook Ansible avancé**

5 **Use cases F5 Ansible**
+ LABO

6 **Ansible Tower**

INTRODUCTION À ANSIBLE

SIMPLE

Automatisation facile
Pas besoin d'être programmeur
Les tâches sont exécuter en ordre
Utilisable par tous
Devener productif rapidement

PUISSANT

Déploiement d'application
Gestion de configuration
Orchestration de workflow
Automatisation des réseaux
Orchestrer le cycle de vie complet

SANS AGENT

Sans agent
Utilise OpenSSH & WinRM
Pas d'agent à exploiter ou maintenir
Démarrer immédiatement
Plus efficace, plus sécure

Introduction à Ansible

Michael DeHaan (créateur de Cobbler et de Func)
https://www.ansible.com/blog/2013/12/08/the-origins-of-ansible

Simple
AUTOMATISE TOUT
Peut gérer presque n'importe lequel *IX par le biais d'un protocole SSH

nécessite Python

Windows (PowerShell, module WinRM Python)
Composants de nuage, virtualisation, conteneur et réseau

« Ansible doit une grande partie de ses origines au temps que j'ai passé au sein du groupe des technologies émergentes de Red Hat, qui était une unité de R D sous la direction du CTO de Red Hat. »

- Michael DeHaan

«...parce que Puppet était trop déclaratif, vous ne pouviez pas l'utiliser pour faire des choses comme réinitialiser des serveurs ou effectuer toutes les tâches ad hoc qui devaient être faites entretemps…»

- Michael DeHaan


Une **ansible** est un dispositif **théorique** permettant de réaliser des communications à une vitesse supraluminique. Elle peut envoyer et recevoir des messages en provenance et en direction du périphérique correspondant sur n'importe quelle distance sans aucun délai. Les **ansibles** sont une composante emblématique de la littérature de **science-fiction**.
-- Wikipédia
redhat.

28,000+
Stars on GitHub

1450+
Ansible modules

500,000+
Downloads a month

STARS	TECHNO	CONTRIBUTEURS
29,014	Ansible	3328
16,357	Vagrant	814
11,421	Terraform	1195
8,680	Salt	2021
5,255	Chef	550
4,915	Puppet	491


EN 30 ANS, LA GESTION DES RÉSEAUX
N’A PAS CHANGÉE.

POURQUOI ANSIBLE + RÉSEAU?

« Lorsqu'on leur a demandé ce qui selon eux était le composant le plus immature en gestion du nuage, 76 % ont dit que c'était le réseau; 15 % ont mentionné le traitement et 9 % le stockage. »

Rapport de gestion nuagique SDx 2015 : OpenStack and More sdxcentral.com

networktocode
andrius

All Threads

CHANNELS (59)
ansible
netdevops-survey
general

DIRECT MESSAGES
slackbot
andrius (you)
dgarros
plumbis

#ansible
594 | 4 | Ansibl...

Search

joined #ansible

Pinned by that1guy15

cidrblock 9:41 AM
good morning all. We're starting to realize how handy ansible is not only for conf mgmt but for health and state checking before and after maintenance. We've got a hw upgrade this weekend, ansible is gonna make easy work of collecting every the config and state of every pool member virtual server etc. I forked the internal repo and posted it here in case you may need something similar. https://github.com/cidrblock/f5_health_check

> GitHub
> **cidrblock/f5_health_check**
> f5_health_check - Automated health check, configuration and state collection using ansible for bigip

💯 3 ❤️ 1 😎 1

plumbis 9:42 AM
Big fan of using Ansible for post-change validation

cidrblock 9:42 AM
@plumbis yeah, it's just starting to 'click' at work for some people...

Message #ansible

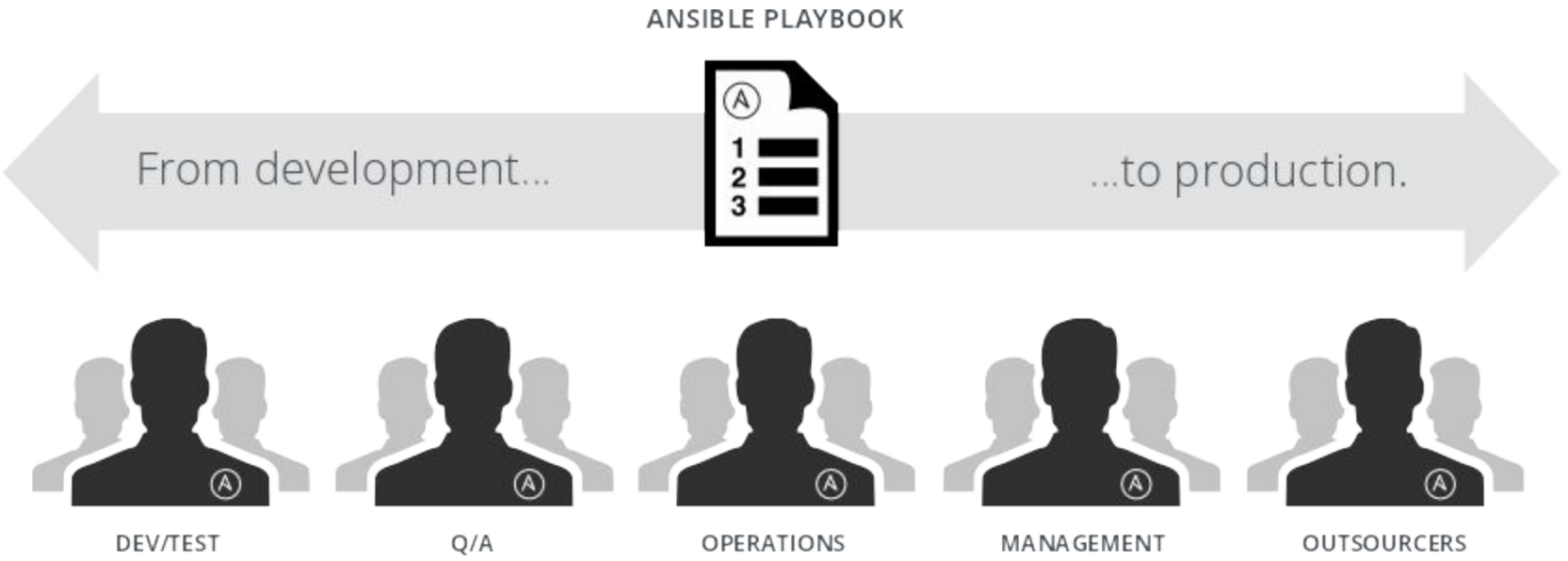
← **"A-ha" Moment (Feb. 2, 2017)**

AVANTAGES

Pourquoi Ansible est-il populaire?

- **Efficace :** sans agent, installation minimale, état désiré (aucun changement non nécessaire), architecture basée sur la technologie de diffusion personnalisée, ciblage facile basé sur des faits
- **Rapide :** Facile à apprendre/à se rappeler, langage déclaratif simple
- **Évolutif :** Peut gérer des milliers de noeuds, architecture modulaire extensible
- **Securitaire :** Transport au travers SSH
- **Vaste communauté** : des milliers de rôles sur Ansible Galaxy

ANSIBLE - LE LANGAGE DE DEVOPS



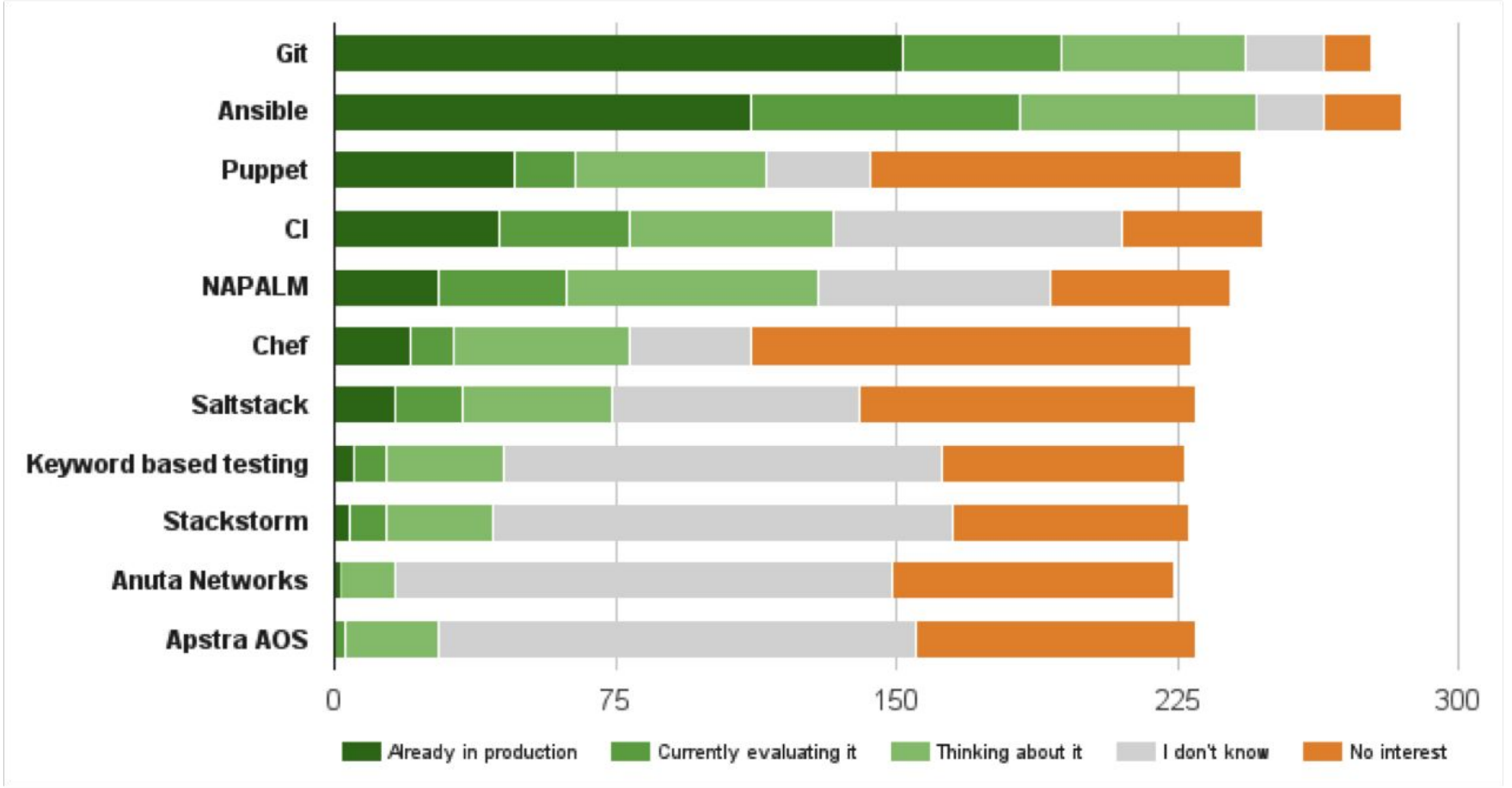
COMMUNICATION IS THE KEY TO DEVOPS.

Ansible is the first **automation language** that can be read and written across IT.

Ansible is the only **automation engine** that can automate the entire **application lifecycle** and **continuous delivery** pipeline.

DU RÉSEAU AU CODE – SONDAGE NETDEVOPS (NOV. 2016)

« *Lesquels des outils suivants vous intéressent ou avez-vous déployés?* »



PRET ?

COMPOSANTS CLÉS

Comprendre les termes d'Ansible

- ★ **Playbook (Plan)**
- ★ **Plays**
- ★ **Tasks**
- ★ **Modules (Tools)**
- ★ **Inventory**

INSTALLATION D'ANSIBLE

Mode d'emploi

https://access.redhat.com/articles/3174981

```
# CENTOS
# INSTALLER LE REPO EPEL
yum install epel-release

# RHEL
# ACTIVER LE REPOS ANSIBLE
subscription-manager repos --enable=rhel-7-server-ansible-VERSION-rpms

# INSTALLER ANSIBLE
yum install ansible
```

Est-ce que Red Hat offre du soutien pour les services de base d'Ansible?

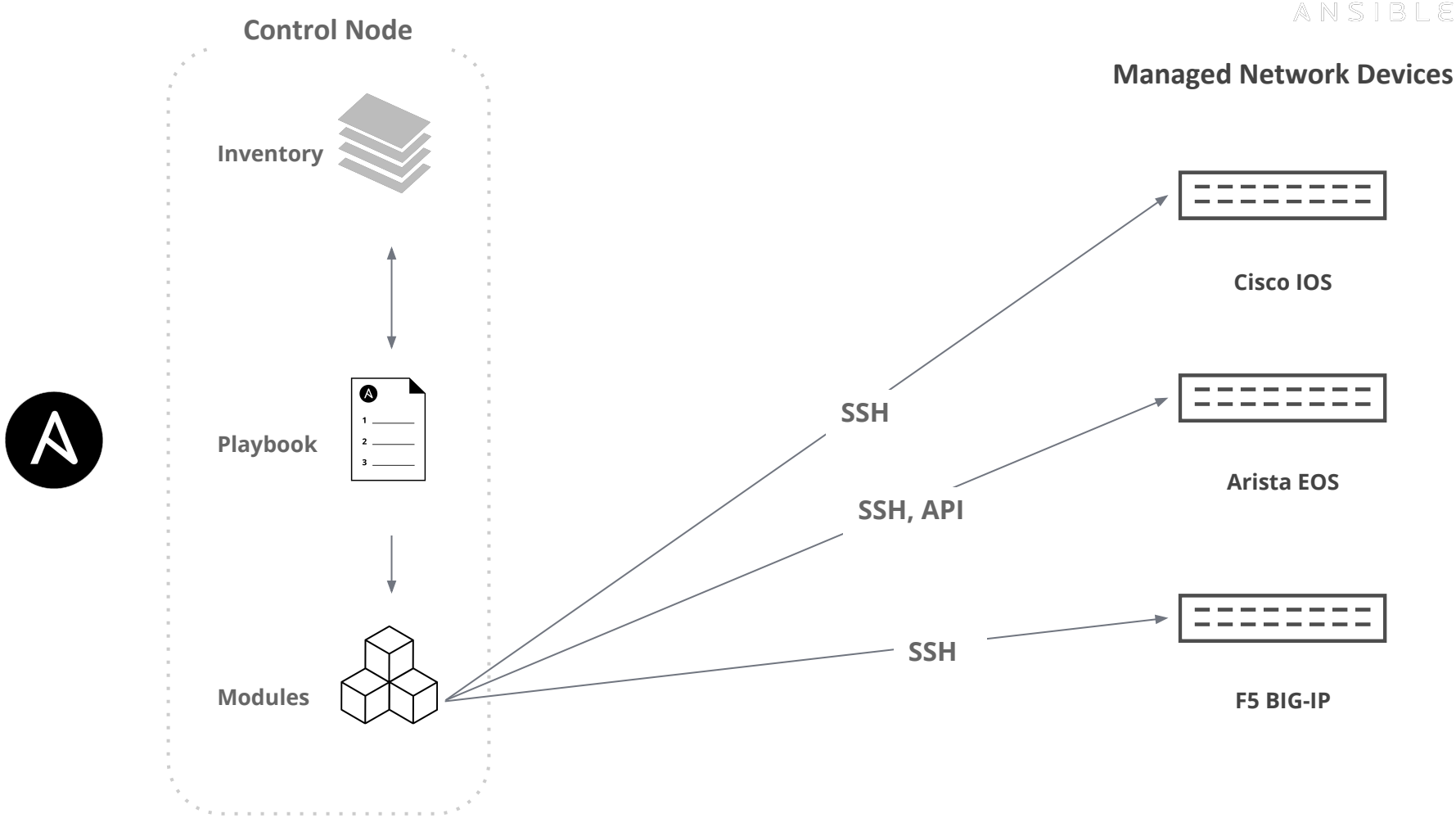
https://access.redhat.com/articles/2271461

MODULES

En quoi ça consiste?

Bouts de code copiés sur le système cible.
Exécutés pour satisfaire à la déclaration de tâche.
Personnalisables.

Les modules qui expédient avec Ansible sont tous écrit en Python, mais les modules peuvent être écrits en n'importe quel langage.



Modules:
Handles execution of remote system commands

Control Node:
Any client system (server, laptop, VM) running Linux or Mac OSX

Managed Nodes (Inventory):
A collection of endpoints being managed via SSH or API.

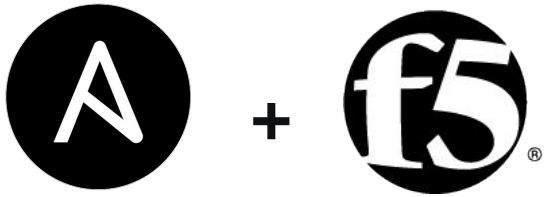
MODULES

Vaste choix / force secrète d'Ansible...

- ➔ **Modules de nuage**
- ➔ **Modules de grappes**
- ➔ **Modules de commandements**
- ➔ **Modules Crypto**
- ➔ **Modules de bases de données**
- ➔ **Modules de fichiers**
- ➔ **Modules d'identités**
- ➔ **Modules d'inventaire**
- ➔ **Modules de messages**
- ➔ **Modules de surveillance**
- ➔ **Modules de réseaux**
- ➔ **Modules de notification**
- ➔ **Modules de gestion à distance**
- ➔ **Modules d'intégration**
- ➔ **Modules de contrôle à la source**
- ➔ **Modules de stockage**
- ➔ **Modules de système**
- ➔ **Modules de logiciels utilitaires**
- ➔ **Modules d'infrastructures Web**
- ➔ **Modules Windows**

F5 ANSIBLE MODULES

ANSIBLE



bigip_pool - Manages F5 BIG-IP LTM pools
bigip_pool_member - Manages F5 BIG-IP LTM pool members
bigip_profile_client_ssl - Manages client SSL profiles on a BIG-IP
bigip_provision - Manage BIG-IP module provisioning
bigip_qkview - Manage qkviews on the device
bigip_remote_syslog - Manipulate remote syslog settings on a BIG-IP
bigip_routedomain - Manage route domains on a BIG-IP
bigip_security_address_list - Manage address lists on BIG-IP AFM
bigip_security_port_list - Manage port lists on BIG-IP AFM
bigip_selfip - Manage Self-IPs on a BIG-IP system
bigip_snat_pool - Manage SNAT pools on a BIG-IP
bigip_snmp - Manipulate general SNMP settings on a BIG-IP
bigip_snmp_trap - Manipulate SNMP trap information on a BIG-IP
bigip_software_update - Manage the software update settings of a BIG-IP
bigip_ssl_certificate - Import/Delete certificates from BIG-IP
bigip_ssl_key - Import/Delete SSL keys from BIG-IP
bigip_static_route - Manipulate static routes on a BIG-IP
bigip_sys_db - Manage BIG-IP system database variables
bigip_sys_global - Manage BIG-IP global settings
bigip_traffic_group - Manages traffic groups on BIG-IP
bigip_ucs - Manage upload, installation and removal of UCS files
bigip_ucs_fetch - Fetches a UCS file from remote nodes
bigip_user - Manage user accounts and user attributes on a BIG-IP
bigip_vcmp_guest - Manages vCMP guests on a BIG-IP
bigip_virtual_address - Manage LTM virtual addresses on a BIG-IP
bigip_virtual_server - Manage LTM virtual servers on a BIG-IP
bigip_vlan - Manage VLANs on a BIG-IP system
bigip_wait - Wait for a BIG-IP condition before continuing
bigiq_regkey_license - Manages licenses in a BIG-IQ registration key pool
bigiq_regkey_pool - Manages registration key pools on BIG-IQ

bigip_asm_policy - Manage BIG-IP ASM policies
bigip_command - Run arbitrary command on F5 devices
bigip_config - Manage BIG-IP configuration sections
bigip_configsync_action - Perform different actions related to config-sync
bigip_device_connectivity - Manages device IP configuration settings for HA on a BIG-IP
bigip_device_dns - Manage BIG-IP device DNS settings
bigip_device_group - Manage device groups on a BIG-IP
bigip_device_group_member - Manages members in a device group
bigip_device_httpd - Manage HTTPD related settings on BIG-IP
bigip_device_ntp - Manage NTP servers on a BIG-IP
bigip_device_sshd - Manage the SSHD settings of a BIG-IP
bigip_device_trust - Manage the trust relationships between BIG-IPs
bigip_facts - Collect facts from F5 BIG-IP devices
bigip_gtm_datacenter - Manage Datacenter configuration in BIG-IP
bigip_gtm_facts - Collect facts from F5 BIG-IP GTM devices
bigip_gtm_pool - Manages F5 BIG-IP GTM pools
bigip_gtm_server - Manages F5 BIG-IP GTM servers
bigip_gtm_virtual_server - Manages F5 BIG-IP GTM virtual servers
bigip_gtm_wide_ip - Manages F5 BIG-IP GTM wide ip
bigip_hostname - Manage the hostname of a BIG-IP
bigip_iapp_service - Manages TCL iApp services on a BIG-IP
bigip_iapp_template - Manages TCL iApp templates on a BIG-IP
bigip_iapplx_package - Manages Javascript iApp packages on a BIG-IP
bigip_irule - Manage iRules across different modules on a BIG-IP
bigip_monitor_http - Manages F5 BIG-IP LTM http monitors
bigip_monitor_https - Manages F5 BIG-IP LTM https monitors
bigip_monitor_snmp_dca - Manages BIG-IP SNMP data collecting agent (DCA) monitors
bigip_monitor_tcp - Manages F5 BIG-IP LTM tcp monitors
bigip_monitor_tcp_echo - Manages F5 BIG-IP LTM tcp echo monitors
bigip_monitor_tcp_half_open - Manages F5 BIG-IP LTM tcp half-open monitors
bigip_monitor_udp - Manages F5 BIG-IP LTM udp monitors
bigip_node - Manages F5 BIG-IP LTM nodes
bigip_partition - Manage BIG-IP partitions
bigip_policy - Manage general policy configuration on a BIG-IP
bigip_policy_rule - Manage LTM policy rules on a BIG-IP

redhat.

MODULES

Documentation

```
# AFFICHE TOUS LES MODULES
ansible-doc -l


# ACCÉDER À LA DOCUMENTATION D'UN MODULE
ansible-doc <module_name>
```

IDEMPO-QUOI?

« En mathématiques et en informatique, le concept d’idempotence signifie essentiellement qu'une opération a le même effet qu'on l'applique une ou plusieurs fois, ou encore qu'en le réappliquant on ne modifiera pas le résultat. »

« Lorsqu’il est soigneusement écrit, un scénario Ansible peut être idempotent afin de prévenir les effets secondaires imprévus sur les systèmes gérés. »

– Wikipédia

LES COMMANDES ANSIBLE

INVENTAIRE

Pour utiliser l'inventaire par défaut (/etc/ansible/hosts) ou créer un fichier inventaire

```
[centos@centos1 ~]$ mkdir ansible ; cd ansible
[centos@centos1 ~]$ vim inventory

[bigip]
192.168.124.11
```

LAB # 1

Inventaire

Objectifs

Créer votre fichier inventaire

1. Validez l'accès à votre intance BIG-IP. Connectez vous via le web à votre appliance. IP et credentials vous serons fournis par l'instructeur
2. # vim /etc/ansible/hosts
3. ajoutez ce-ci au fichier (prendre l'ip qui vous est assigné)

   [bigip]

   192.168.124.11

PLAYBOOK ANSIBLE

YAML

1. Principalement conçu pour la représentation des structures de données
2. Facile à écrire, format pouvant être lu par les humains
3. Objectif de la conception : abandonner la syntaxe traditionnelle “cloisonée”

ÉVITEZ D’UTILISER LE COPIER-COLLER!!!

Programme de validation YAML : yamllint.com

EXEMPLE DE PLAYBOOK

```
---
- name: This is a Play
  hosts: bigip
  gather_facts: no
  vars:
    username: admin
    password: 1qaz2wsX

  tasks:
    - name: Manage NTP setting on BIG-IP
      bigip_device_ntp:
        server: "{{ inventory_hostname }}"
        user: "{{ username }}"
        password: "{{ password }}"
        ntp_servers:  0.centos.pool.ntp.org
        validate_certs: False
      delegate_to: localhost
```

PLAYS

Nommage

```
- name: This is a Play
```

PLAYS

Sélection des hôtes

```
- name: This is a Play
  hosts: bigip

- name: this is a play 2
  hosts: all
```

PLAYS

Arguments

```
- name: This is a Play
  hosts: bigip
  gather_facts: no
```

FAITS

Recueille les faits au sujet de l'hôte distant

- **Ansible fournit automatiquement de nombreux faits au sujet des systèmes contactés**
- **Fournit par le module setup**
- **Mettre gather_facts à no lorsqu'utilisé pour des composantes réseautique.**
- **Les faits des composantes réseaux sont accessible via des modules spécifiques**

http://docs.ansible.com/ansible/setup_module.html

PLAYS

Variables et tâches

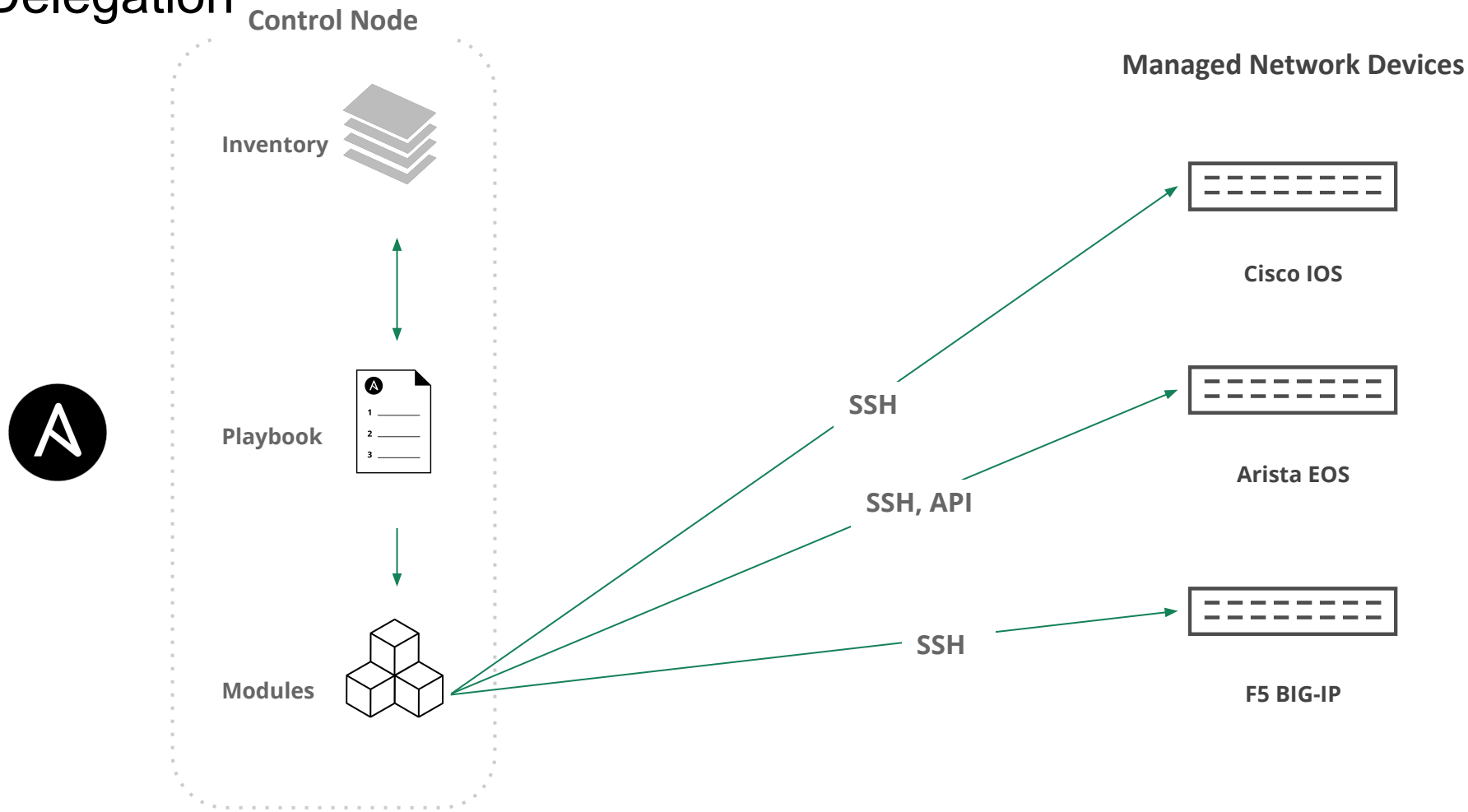
```
---
- name: This is a Play
  hosts: bigip
  gather_facts: no
  vars:
    username: admin
    password: 1qaz2wsX


  tasks:
     - name: Manage NTP setting on BIG-IP
       bigip_device_ntp:                        // module appelé
         server: "{{ inventory_hostname }}"     // paramètres du module
         user: "{{ username }}"
         password: "{{ password }}"
         ntp_servers: 0.centos.pool.ntp.org
         validate_certs: False
       delegate_to: localhost
```

**** Lorsqu'une variable est utilisée comme premier élément pour commencer une valeur, les guillemets sont obligatoires.

DÉLEGATION

```
---
- name: This is a Play
  hosts: bigip
  gather_facts: no
  vars:
    username: admin
    password: 1qaz2wsX


  tasks:
     - name: Manage NTP setting on BIG-IP
       bigip_device_ntp:
         server: "{{ inventory_hostname }}"
         user: "{{ username }}"
         password: "{{ password }}"
         ntp_servers: 0.centos.pool.ntp.org
         validate_certs: False
       delegate_to: localhost
```

Délégation



Modules:
Handles execution of remote system commands

Control Node:
Any client system (server, laptop, VM) running Linux or Mac OSX

Managed Nodes (Inventory):
A collection of endpoints being managed via SSH or API.

EXÉCUTER UN PLAYBOOK ANSIBLE

```
[centos@centos7-1 ansible]$ ansible-playbook play.yml
```

EXÉCUTER UN PLAYBOOK ANSIBLE

En mode vérification uniquement

```
[centos@centos7-1 ansible]$ ansible-playbook play.yml --check
```

RESULT

Enregistre les résultats de la tâche pour le débogage ou d'autres fins

```
- shell: /usr/bin/uptime
  register: result

- debug: var=result    (module debug : va afficher le résultat)
```

Exemple playbook

```
---
- name: This is a Play
  hosts: bigip
  gather_facts: no
  vars:
    username: admin
    password: 1qaz2wsX


  tasks:
     - name: Manage NTP setting on BIG-IP
       bigip_device_ntp:
         server: "{{ inventory_hostname }}"
         user: "{{ username }}"
         password: "{{ password }}"
         ntp_servers: 0.centos.pool.ntp.org
         validate_certs: False
       delegate_to: localhost
```

LAB # 2

Premier playbook

Objectifs

Créer votre premier playbook

1. En utilisant la documentation ansible des modules bigip, et basé sur l'information transmise, créer votre premier playbook qui appel le module bigip_command et retourne la version du BIG IP

LAB #2 - SOLUTION

```
---
- name: F5 Command
  gather_facts: false
  hosts: bigip
  vars:
      username: admin
      password: admin

  tasks:
  - name: show sys version
      bigip_command:
        commands: show sys version
        server: "{{ inventory_hostname }}"
        user: "{{ username }}"
        password: "{{ password }}"
        validate_certs: False
      delegate_to: localhost
      register: result

  - debug:
      var: result

[centos@centos1 ansible]$ ansible-playbook -i inventory lab2.yaml
```

PLAYBOOK ANSIBLE -- AVANCÉ

PLAYS

Boucles ***

```
- name: F5 Initial setup
  gather_facts: false
  hosts: bigip
  vars:
      username: admin
      password: admin
  tasks:
    - name: Add http node to web-pool
      bigip_pool_member:
        description: "HTTP Webserver-1"
        host: "{{ item.host }}"
        name: "{{ item.name }}"
        user: "{{ username }}"
        password: "{{ password }}"
        pool: "web-pool"
        port: "80"
        server: "{{ inventory_hostname }}"
       validate_certs: False
      with_items:
        - host: "192.168.168.140"
          name: "web01.internal"
        - host: "192.168.68.141"
          name: "web02.internal"
      delegate_to: localhost
```

BOUCLES

Plusieurs types de boucles générales et à usage déterminé

- **with_nested**
- **with_dict**
- **with_fileglob**
- **with_together**
- **with_sequence**
- **until**
- **with_random_choice**
- **with_first_found**
- **with_indexed_items**
- **with_lines**

http://docs.ansible.com/ansible/playbooks_loops.html

HANDLERS

Exécuter seulement si la tâche a un statut « modifié »

```
---
- name: This is a Play
  hosts: web

  tasks:
    - yum: name={{ item }} state=installed
      with_items:
        - httpd
        - memcached
      notify: Restart Apache

    - template: src=templates/web.conf.j2
dest=/etc/httpd/conf.d/web.conf
      notify: Restart Apache

  handlers:
    - name: Restart Apache
      service: name=httpd state=restarted
```

TAG

Exemple d’utilisation d’un tag

```
tasks:

    - yum: name={{ item }} state=installed
      with_items:
         - httpd
         - memcached
      tags:
         - packages

    - template: src=templates/src.j2 dest=/etc/foo.conf
      tags:
         - configuration
```

TAGS

Exécuter avec des tags

```
ansible-playbook example.yml --tags “configuration”

ansible-playbook example.yml --skip-tags "notification"
```

TÂCHES CONDITIONNELLES

Seulement exécuter sur la machine dont le système d'exploitation est Red Hat

```
- name: This is a Play
  hosts: web
  remote_user: centos
  become: sudo

  tasks:
    - name: install Apache
      yum: name=httpd state=installed
      when: ansible_os_family == "RedHat"
```

BLOCS

Applique une condition à plusieurs tâches à la fois

```
tasks:

  - block:
    - yum: name={{ item }} state=installed
      with_items:
        - httpd
        - memcached
    - template: src=templates/web.conf.j2 dest=/etc/httpd/conf.d/web.conf
    - service: name=bar state=started enabled=True
    when: ansible_distribution == 'CentOS'
```

ERREURS

Ignore les erreurs

Par défaut, Ansible s'arrête aux erreurs. Ajoutez le paramètre ignore_error pour sauter les erreurs possibles.

```
- name: ping host
  command: ping -c1 www.foobar.com
  ignore_errors: yes
```

ERREURS

Gérer les erreurs à l'aide des blocs

```
tasks:

 - block:
     - debug: msg='i execute normally'
     - command: /bin/false
     - debug: msg='i never execute, cause ERROR!'
   rescue:
     - debug: msg='I caught an error'
     - command: /bin/false
     - debug: msg='I also never execute :-('
   always:
     - debug: msg="this always executes"
```

LINEINFILE

Pour ajouter, enlever ou mettre à jour une ligne en particulier

```
-   lineinfile: dest=/etc/selinux/config regexp=^SELINUX=
    line=SELINUX=enforcing

-   lineinfile: dest=/etc/httpd/conf/httpd.conf regexp="^Listen "
    insertafter="^#Listen " line="Listen 8080"
```

Vous trouverez ci-dessous un très bon exemple :
https://relativkreativ.at/articles/how-to-use-ansibles-lineinfile-module-in-a-bulletproof-way

Remarque : L'utilisation d'un template ou d'un module dédié est plus efficace

ANSIBLE CONFIG

Pour configurer ansible, le fichier par défaut est ansible.cfg

Par exemple, pour retirer les information de DEPRECATION WARNING

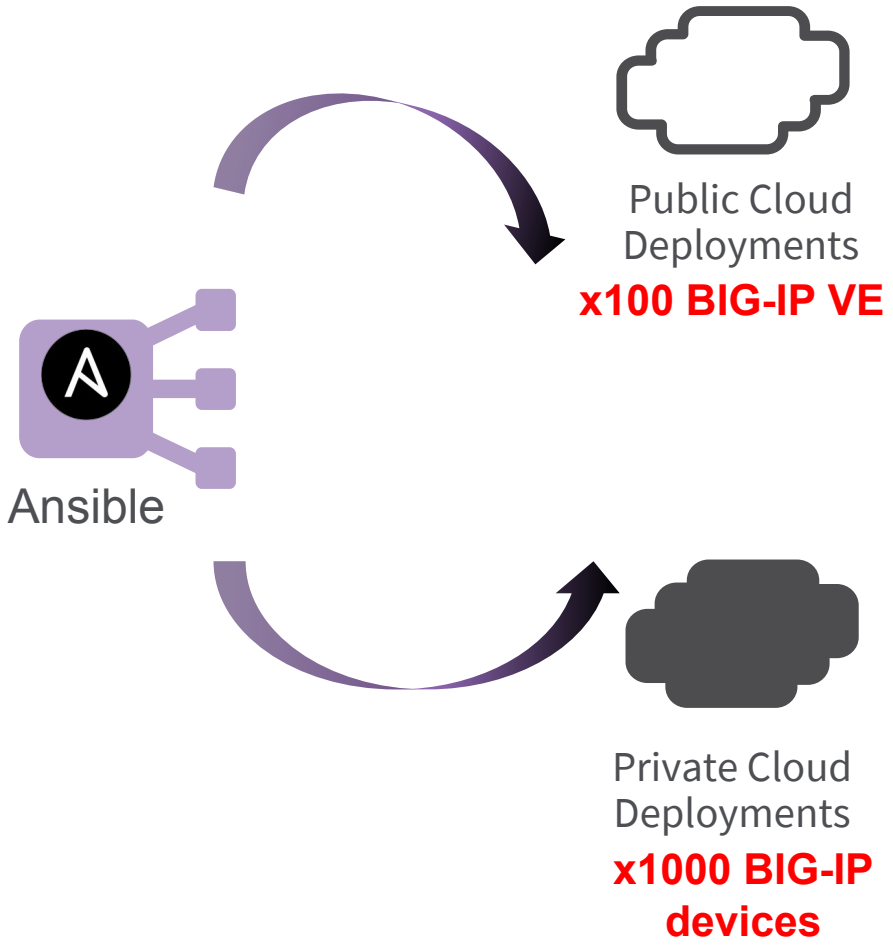
```
TASK [show sys version] ********************************************************************
[DEPRECATION WARNING]: Param 'server' is deprecated. See the module docs for more
information. This feature will be removed in version 2.9. Deprecation warnings can be
disabled by setting deprecation_warnings=False in ansible.cfg.
```

```
vim ansible.cfg

[defaults]
warnings = False
stdout_callback = debug
deprecation_warnings = False
```

USE CASE #1 - Onboarding

Onboard the F5 BIG-IP with initial configurations for application deployment using Ansible Playbooks

- Configure new infrastructure without sysadmins needing to scour documentation to remember how to do it.
- Bootstrap & provision large BIG-IP infrastructures using Ansible playbooks (write once, run over and over)



LAB #3

Configuration initiale de BIGIP

Objectifs

À l'aide d'un playbook Ansible :

1. Configurer les éléments suivant à l'aide d'un playbook ansible
    a. Serveurs ntp (bigip_device_ntp):
        i. 0.centos.pool.ntp.org
        ii. 1.centos.pool.ntp.org
    b. SSH (bigip_device_ssh)
        i. Activer la bannière ssh
        ii. Bannière : "---- Bienvenue sur BIGIP F5 Ansible workshop ---"
    c. DNS (bigip??)
        i. Activez la configuration DNS
        ii. Indiquez comme serveur de nom : 8.8.8.8

LAB #3 - SOLUTION

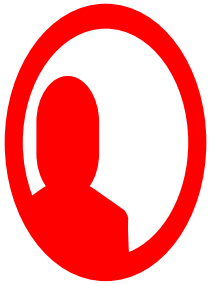
```
---
- name: F5 initial setup
  gather_facts: false
  hosts: bigip
  vars:
    username: admin
    password: 1qaz2wsX
    ntp_servers:
      - '0.centos.pool.ntp.org'
      - '1.centos.pool.ntp.org'
    banner_text: "---- Bienvenue sur BIGIP F5 Ansible workshop ---"

  tasks:
  - name: Configure NTP server on BIG-IP
    bigip_device_ntp:
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      ntp_servers: "{{ ntp_servers }}"
      validate_certs: False
    delegate_to: localhost

  - name: Manage SSHD setting on BIG-IP
    bigip_device_sshd:
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      banner: "enabled"
      banner_text: " {{ banner_text }}"
      validate_certs: False
    delegate_to: localhost

  - name: Set the DNS setting on the BIG-IP
    bigip_device_dns:
      name_servers:
        - 8.8.8.8
      search:
        - localdomain
      state: present
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      validate_certs: False
    delegate_to: localhost
```

USE CASE #2 SaaS & Multi-tenancy

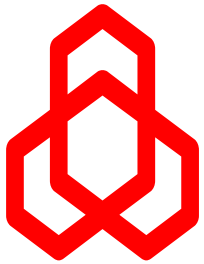
Scale up or scale down BIG-IP objects based on your tenants need or your consumer needs

- Add/Remove Applications on the F5 BIG-IP automatically through iApps
- Add/remove new virtual servers & pool members on the F5 BIG-IP
- Deploy completely new VE's and configure application

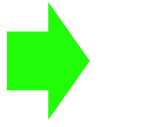
With out Automation: days through a ticketing system + 2 hours of manual tasks
With automation: 10 mins

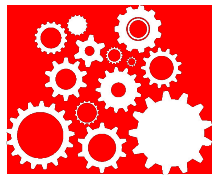


Application Owner



Service Portal (E.g. Jenkins, Service Now)





Automation & Orchestration using Ansible

Faster delivery

LAB #4

Déploiement d'une application

Objectifs

1. Configurer les éléments suivant à l'aide d'un playbook ansible
   a. Créer deux noeuds (ip : xx.xx.xx.xx et xx.xx.xx.xx)
   b. Créer un webpool qui contient ces deux noeuds
   c. Créer un serveur virtuel assigné à ce pool

LAB #4 - SOLUTION 1

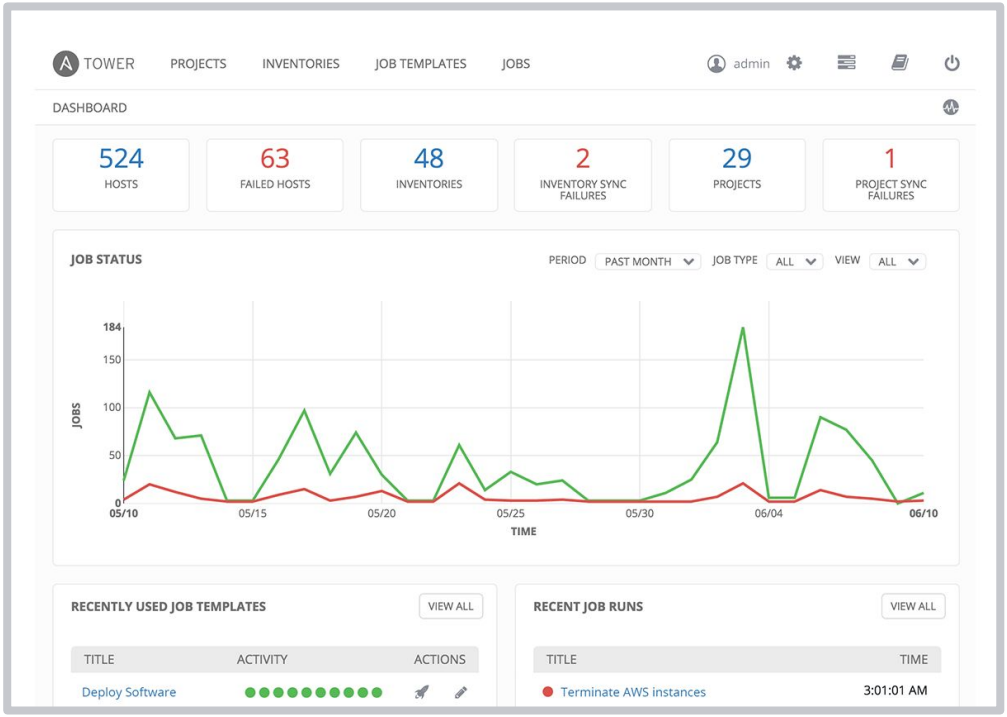
```
---

- name: F5 Initial setup
  gather_facts: false
  hosts: bigip
  vars:
       username: admin
       password: admin

  tasks:
  - name: Create a web01.internal node             //Creating Node1
    bigip_node:
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      host: "192.168.68.140"
      name: "web01.internal"
      validate_certs: False
    delegate_to: localhost

  - name: Create a web02.internal node             //Creating Node2
    bigip_node:
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      host: "192.168.68.141"
      name: "web02.internal"
      validate_certs: False
      delegate_to: localhost

  - name: Create a web-pool                        //Creating a pool
    bigip_pool:
      server: "{{ inventory_hostname }}"
      user: "{{ username }}"
      password: "{{ password }}"
      lb_method: "ratio_member"
      monitors:
        - '/Common/gateway_icmp'
      monitor_type: 'and_list'
      name: "web-pool"
      validate_certs: False
    delegate_to: localhost
```

LAB #4 - SOLUTION 2

```
- name: Add http node to web-pool                //Assigning members to a pool
  bigip_pool_member:
    description: "HTTP Webserver-1"
    host: "{{ item.host }}"
    name: "{{ item.name }}"
    user: "{{ username }}"
    password: "{{ password }}"
    pool: "web-pool"
    port: "80"
    server: "{{ inventory_hostname }}"
    validate_certs: False
  with_items:
    - host: "192.168.168.140"
      name: "web01.internal"
    - host: "192.168.68.141"
      name: "web02.internal"
  delegate_to: localhost

- name: Create a virtual server                 //Create a HTTPS Virtual Server
  bigip_virtual_server:
    description: "Secure web application"
    server: "{{ inventory_hostname }}"
    user: "{{ username }}"
    password: "{{ password }}"
    name: "https_vs"
    destination: "10.10.20.120"
    port: 443
    snat: "Automap"
    all_profiles:
      - http
      - clientssl
    pool: "web-pool"
    validate_certs: False
  delegate_to: localhost
```

WHAT IS ANSIBLE TOWER?

ANSIBLE



Ansible tower is an **enterprise framework** for controlling, securing and managing your Ansible automation – with a **UI and restful API.**

- **Encrypted access** management centralized (ssh, password, api access, etc ...)
- **Role-based access control** keeps environments secure, and teams efficient.
- All Ansible automations are centrally logged**,** ensuring **complete auditability and compliance**.
- Non-privileged users can **safely deploy** entire applications with **push-button deployment** access.
- **Integrates** with the api, Callback provisioning
- **Schedule** playbook execution

redhat.

Configuration F5 credential Tower

Avec Tower, pour l'instant, il n'y a pas de type de credential pour F5, on doit en créer un.

```
INPUT CONFIGURATION
fields:
  - type: string
    id: user
    label: User
  - secret: true
    type: string
    id: password
    label: Password
required:
  - user
  - password

INJECTOR CONFIGURATION
extra_vars:
  F5_PASSWORD: '{{password}}'
  F5_USER: '{{user}}'
  password: '{{password}}'
  username: '{{user}}'
```

LAB #5

Integration avec Tower

Lab guidée

redhat.

MERCI

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos