Ansible Tips and Tricks

Mike Dahlgren (Mike D)

Chief Architect

miked@redhat.com

Red Hat

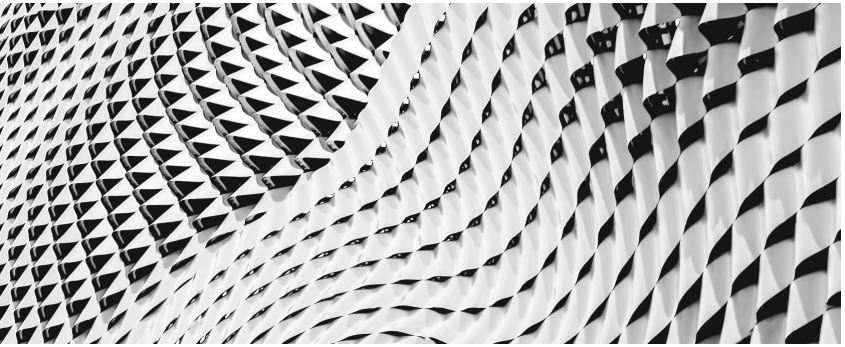
Agenda

What is Ansible

Stupid Ansible tips

Tips for Playbooks

What is Ansible?

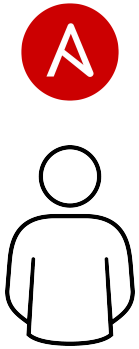


Technical

introduction

and overview

Red Hat



Automation happens when one person meets a problem they never want to solve again

WHY ANSIBLE?

SIMPLE

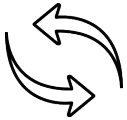
Human readable automation

No special coding skills needed

Tasks executed in order

Usable by every team

Get productive quickly



POWERFUL

App deployment

Configuration management

Workflow orchestration

Network automation

Orchestrate the app lifecycle

AGENTLESS

Agentless architecture

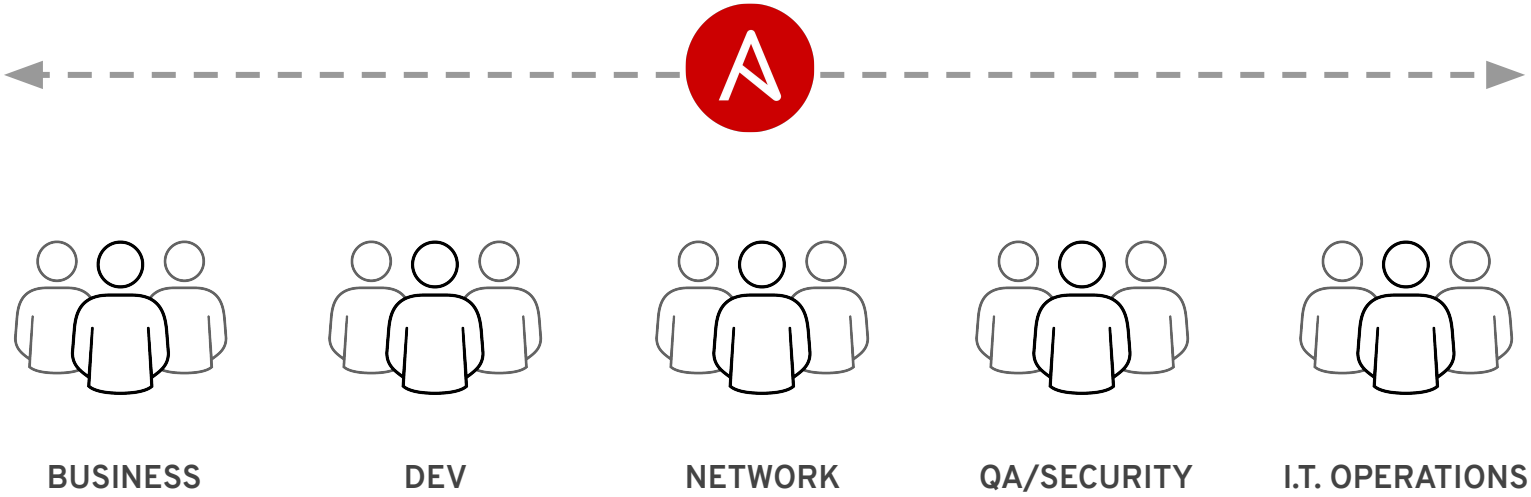
Uses OpenSSH & WinRM

No agents to exploit or update

Get started immediately

More efficient & more secure

ANSIBLE AUTOMATION WORKS ACROSS TEAMS



WHAT CAN I DO USING ANSIBLE?

Automate the deployment and management of your entire IT footprint.

Do this...

Orchestration	Configuration Management	Application Deployment	Provisioning	Continuous Delivery	Security and Compliance

On these...

Firewalls	Load Balancers	Applications	Containers	Clouds
Servers	Infrastructure	Storage	Network Devices	**And more...**

ANSIBLE AUTOMATES TECHNOLOGIES YOU USE

Time to automate is measured in minutes

CLOUD

AWS
Azure
Digital Ocean
Google
OpenStack
Rackspace
+more

OPERATING SYSTEMS

RHEL and Linux
UNIX
Windows
+more

VIRT & CONTAINER

Docker
VMware
RHV
OpenStack
OpenShift
+more

STORAGE

NetApp
Red Hat Storage
Infinidat
+more

WINDOWS

ACLs
Files
Packages
IIS
Regedits
Shares
Services
Configs
Users
Domains
+more

NETWORK

Arista
A10
Cumulus
Bigswitch
Cisco
Cumulus
Dell
F5
Juniper
Palo Alto
OpenSwitch
+more

DEVOPS

Jira
GitHub
Vagrant
Jenkins
Bamboo
Atlassian
Subversion
Slack
Hipchat
+more

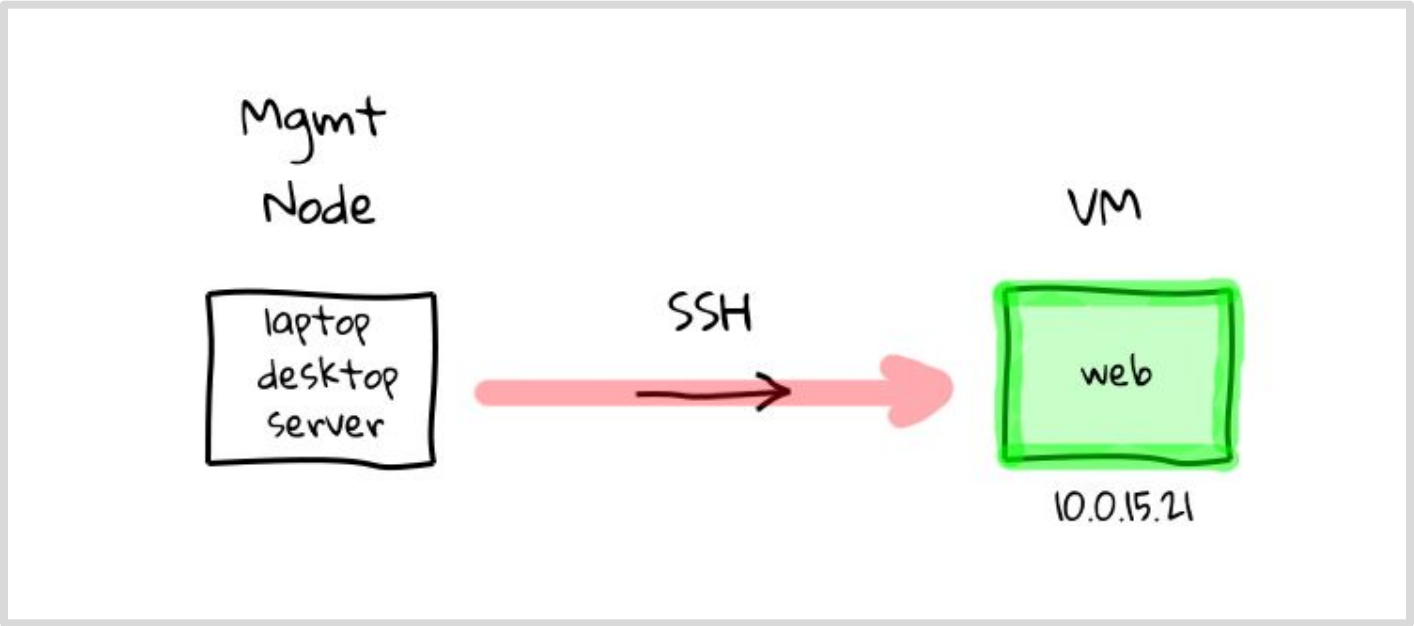
MONITORING

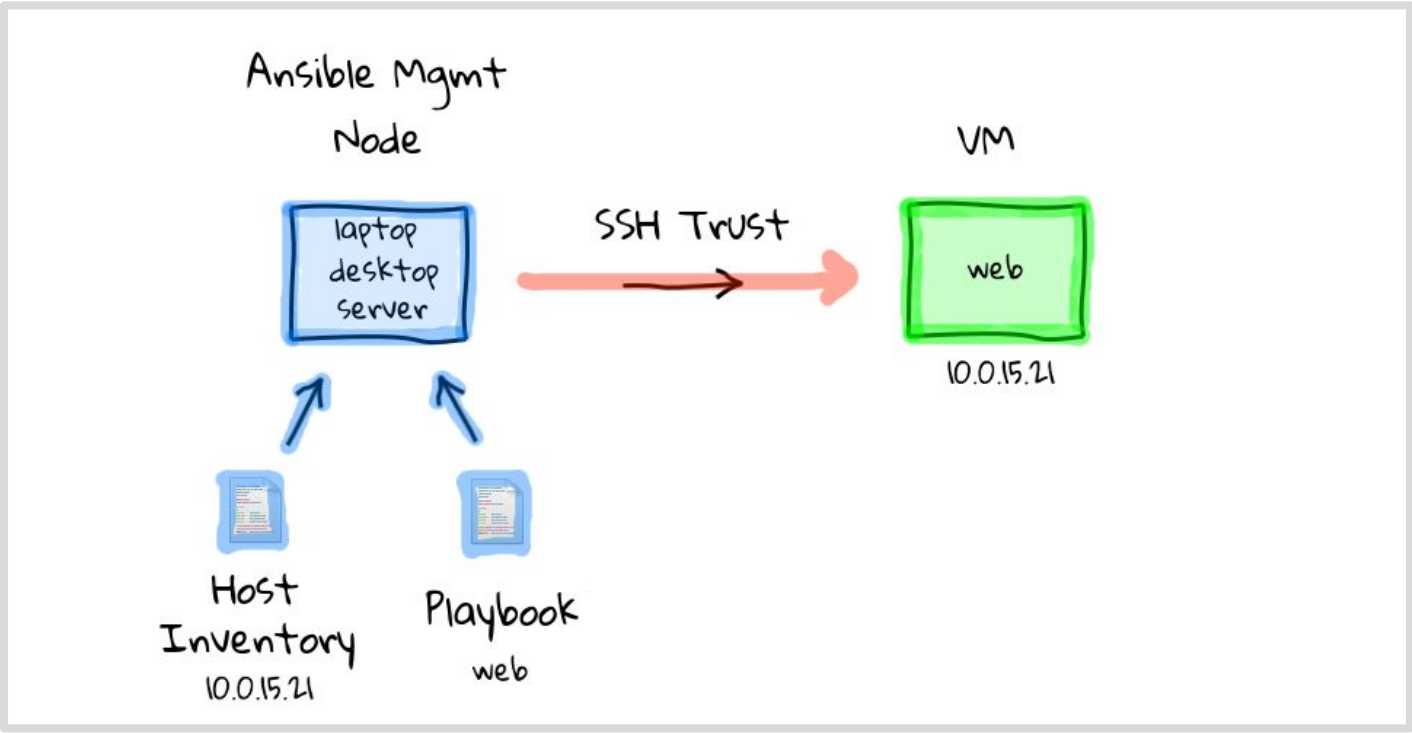
Dynatrace
Airbrake
BigPanda
Datadog
LogicMonitor
Nagios
New Relic
PagerDuty
Sensu
StackDriver
Zabbix
+more

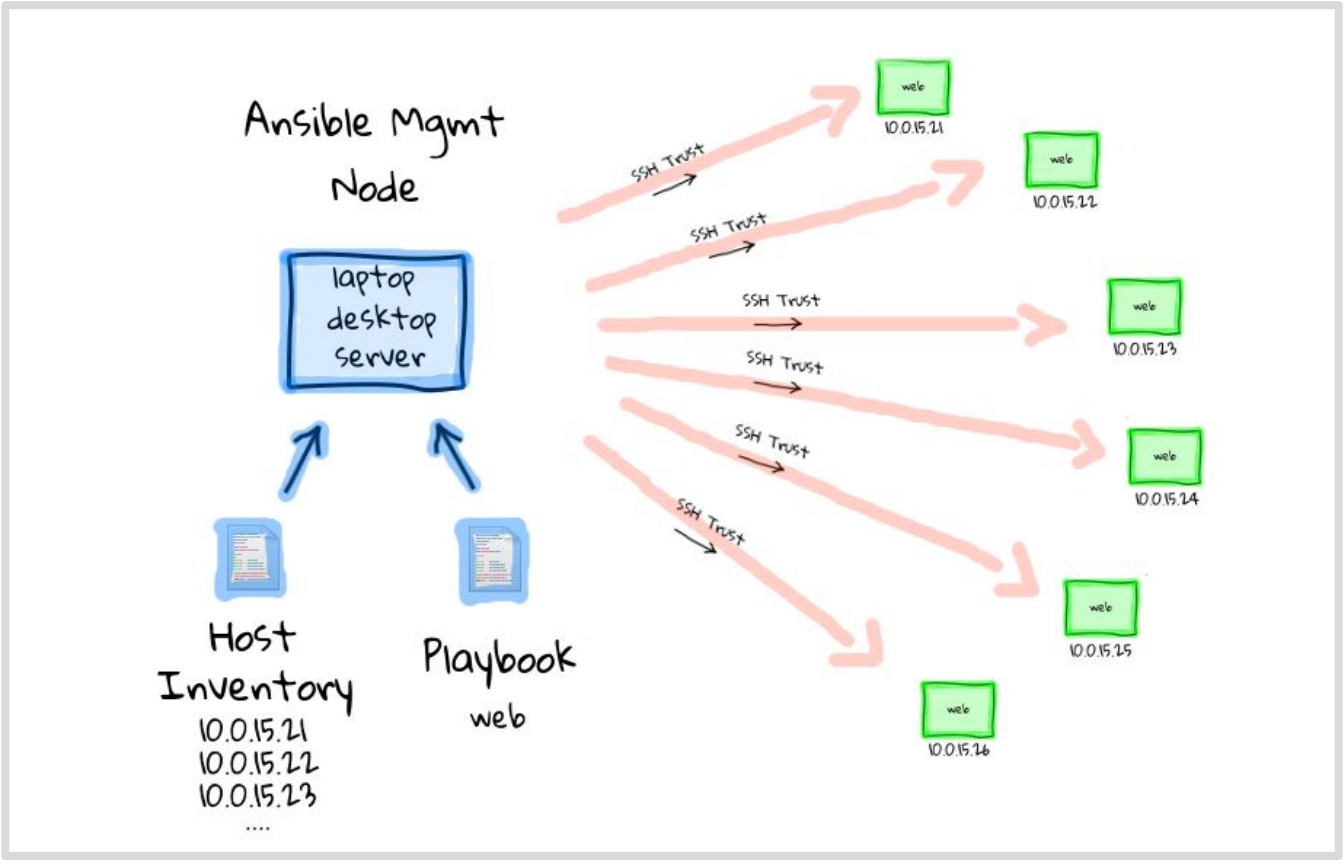
SHOW ME THE CODE!


```
---
- name: install and start apache
  hosts: web
  become: yes
  vars:
     http_port: 80

  tasks:
    - name: httpd package is present
      yum:
        name: httpd
        state: latest

    - name: latest index.html file is present
      copy:
        src: files/index.html
        dest: /var/www/html/

    - name: httpd is started
      service:
        name: httpd
       state: started
```

HOW DOES IT WORK?




Ansible Mgmt Node
laptop
desktop
server
SSH Trust
VM
web
10.0.15.21
Host Inventory
10.0.15.21
Playbook
web

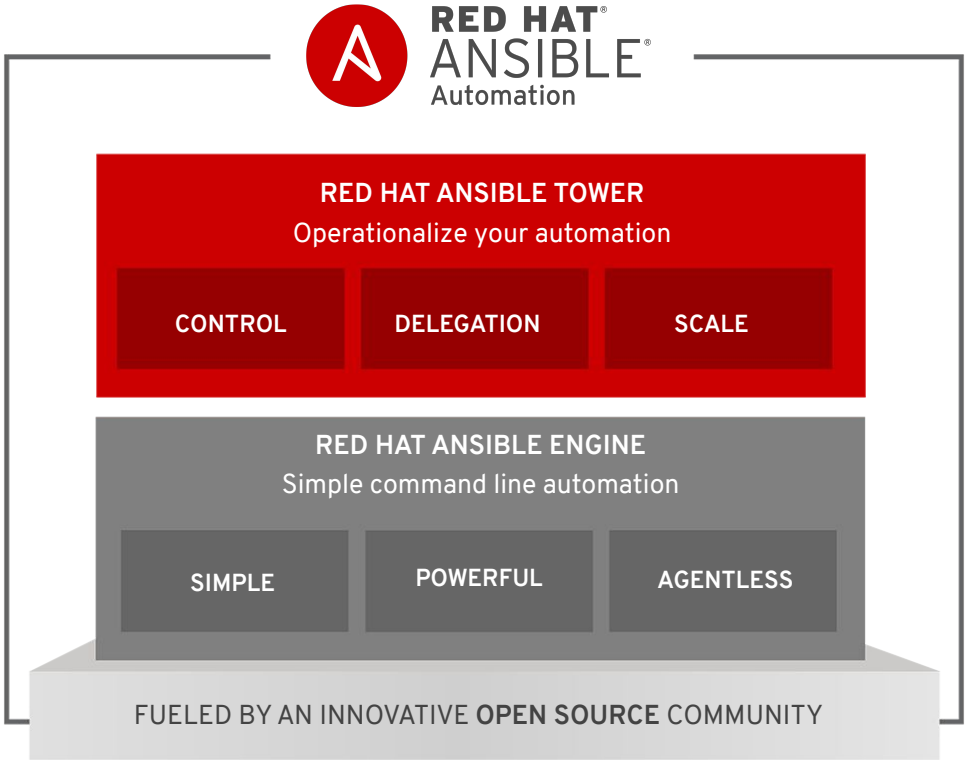

Ansible Mgmt Node
laptop
desktop
server
Host Inventory
10.0.15.21
10.0.15.22
10.0.15.23
....
Playbook
web
SSH Trust
SSH Trust
SSH Trust
SSH Trust
SSH Trust
SSH Trust
web
10.0.15.21
web
10.0.15.22
web
10.0.15.23
web
10.0.15.24
web
10.0.15.25
web
10.0.15.26

WHAT IS ANSIBLE AUTOMATION?

Ansible Automation is the enterprise **framework** for automating across IT operations.

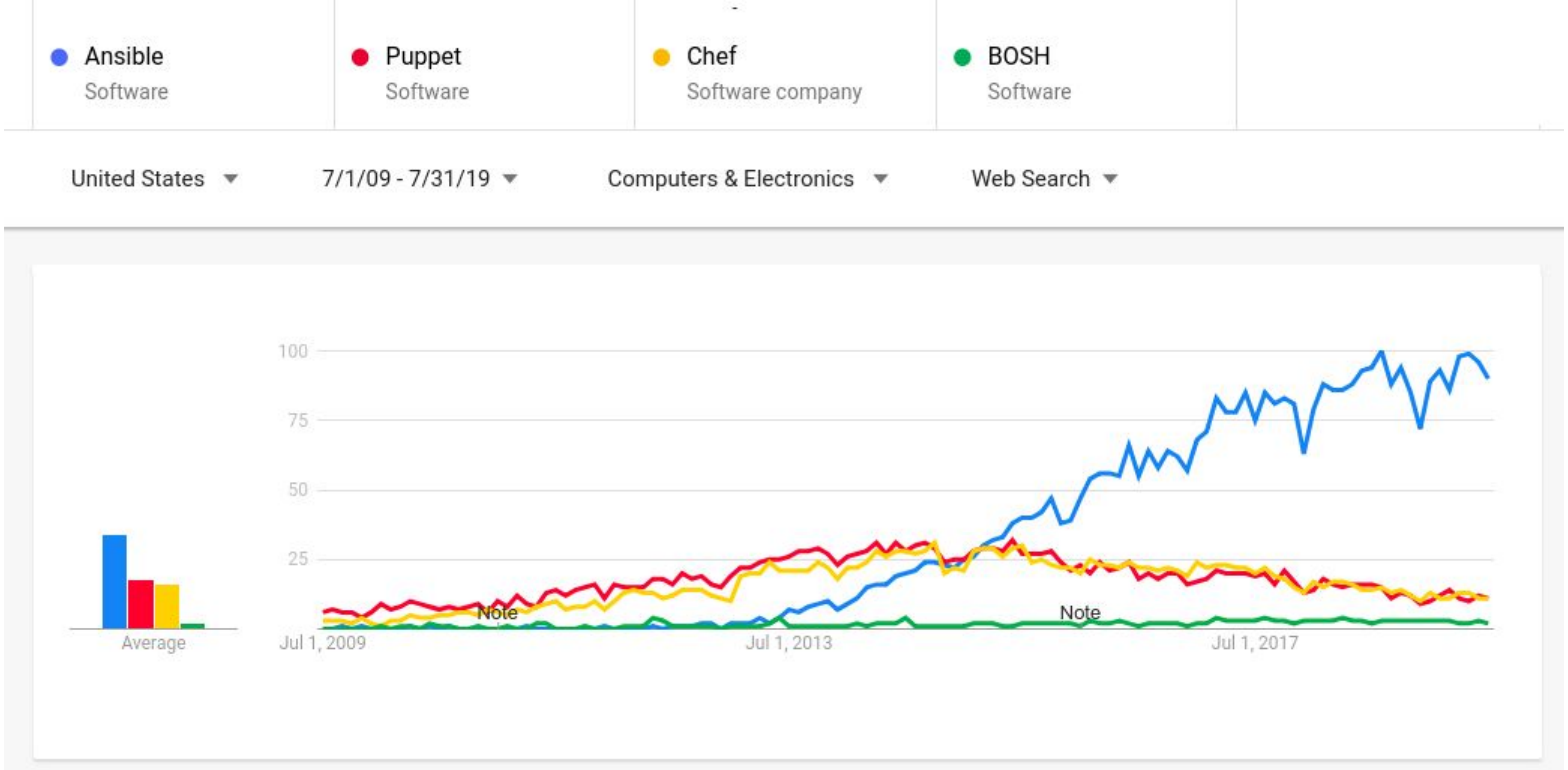
Ansible Engine runs Ansible Playbooks, the automation **language** that can perfectly describe an IT application infrastructure.

Ansible Tower allows you **scale** IT automation, manage complex deployments and speed productivity.



Red Hat

WHERE IS THE MARKET GOING?



* Google Trends

Red Hat

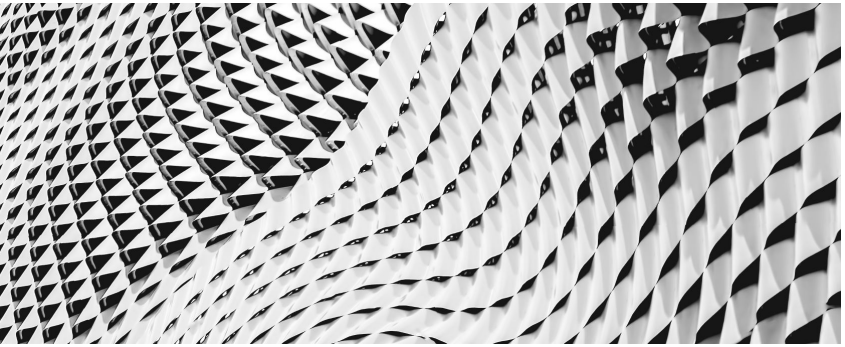
The Forrester WaveTM:

Infrastructure Automation

Platforms, Q3 2019

Red Hat

Stupid Ansible Tricks



#1

Using Ansible interactively

Ad-hoc commands solve simple tasks at cloud scale

$ ansible (targets) -m (module) -a "(arguments)"

#2

Removing a file from a server

Easy for one file:

```
$ ansible webservers -m file -a "dest=/path/to/file
state=absent"
```

Red Hat

Really remove everything! #2

```
- name: remove files and directories
  file:
        state: "{{ item }}"
        path: "/srv/deleteme/"
        owner: 1000  # set owner, group, and mode
        group: 1000
        mode: '0777'
  with_items:
      - absent
      - directory
```

Red Hat

Forking background processes from the command line

#3

Run script in background (30 Min timeout)

```
$ ansible webserver -B 3600 -a "/bin/long_cmd --do-stuff"
```

Checking on the status of a previous job

```
$ ansible web1.example.com -m async_status -a "jid=488359678239.2844"
```

We can set how often to poll the status (60 seconds)

```
$ ansible webserver -B 1800 -P 60 -a "/bin/long_cmd --do-stuff"
```

Running Commands in Parallel

#4

Number of forks can easily be defined with -f (default is 5)

$ ansible webservers -a "/sbin/reboot" -f 10

Strategies can be used to control play execution an can be changed

- Lineary strategy = in order execution (Default)
- Free strategy = finish as fast as you can

```
- hosts: all
  strategy: free
  tasks:
  ...
```

Red Hat

Use Patterns matching

#5

Wildcards work

one*.com:dbservers

So can Regex

~(web|db).*\.example\.com

But would this work?

www[01:50].example.com, db-[a:f].example.com

Red Hat

Overloading the Ansible config

#6

Set Defaults in Custom Ansible Configuration Files

- No need to type -i myhosts from the CLI
- Remove the useless .retry files
- Can be used anywhere you run Ansible

Precedence model:

```
* ANSIBLE_CONFIG (an environment variable)
* ansible.cfg (in the current directory)
* .ansible.cfg (in the home directory)
* /etc/ansible/ansible.cfg
```

Red Hat

Tips for Playbooks

Better faster easier

Give Everything a name

#7

```
---
- hosts: local
      tasks:
      -   User:
          name: user1
          State: present
```

```
PLAY ***************************************************
TASK [user] ********************************************[...]
```

Red Hat

Give Everything a name

#7

```
- name: Setup localhost
  hosts: local
       tasks:
       - name: Create User John
          user:
               name: user1
               state: present
```

```
PLAY [Setup localhost]
**********************************************

TASK [Create User John]
**********************************************[...]
```

Red Hat

Both work, second is better #8

Use full YAML SYNTAX -

- Easier to read
- Supports complex parameter values
- Better syntax in editors / version control

```
- name: add user1
  user:
    name: user1
    state: present
    group: wheel
```

YAML/ANSIBLE

```
- name: add user1
  user: name=user1 state=present groups=wheel
```

Red Hat

Set facts on servers

#9

Think Idempotently, store information on hosts

```
- hosts: webserver1
  tasks:
  - name:  “Has DNS been configured yet?”
    set_fact:
      dns_configured_yet: “no”
```

After DNS has been setup and tested change fact to “yes” or “true”

Negative Verbosity?

#10

```
- debug:
    msg: "This always displays"

- debug:
    msg: "This only displays with ansible-playbook -vv+"
    verbosity: 2
```

Disable Warnings

#11

```
PLAY [command] ********************************
    [WARNING]: Consider using yum module than running yum...
Changed: [localhost]
```

```
- hosts: all
  tasks:
    - command: yum -y install telnet
...
    - command: yum -y install telnet
      args:
        warn: False
```

Red Hat

Always verify results

#12

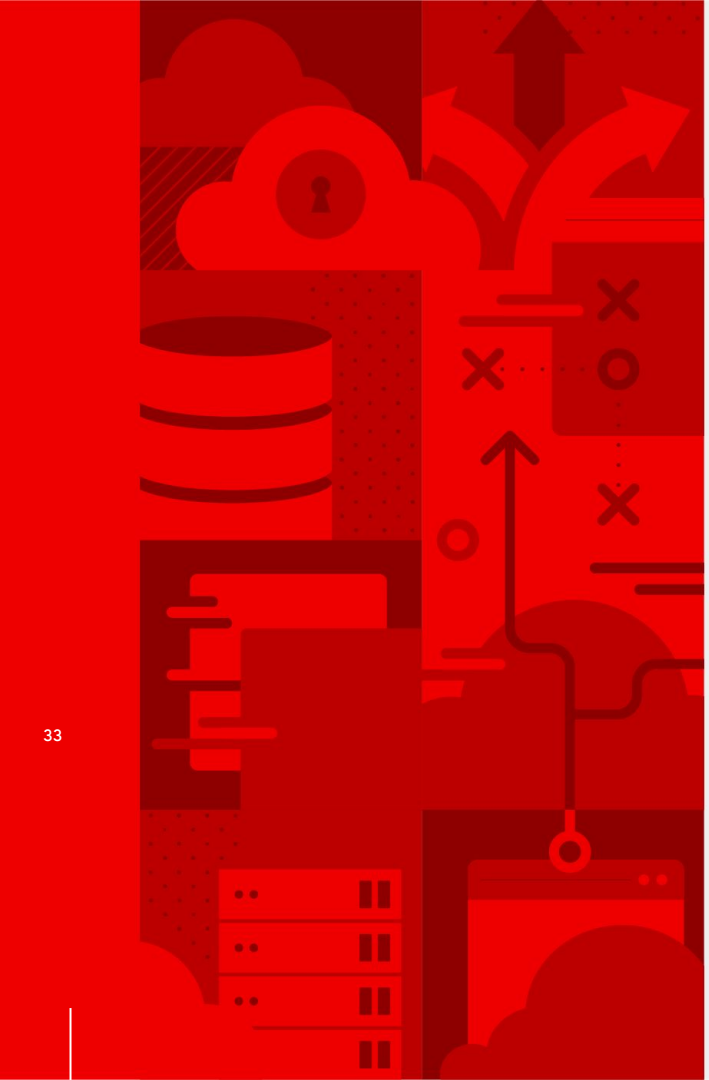
```
- name: check for proper response
  uri:
    url: http://localhost/myapp
    return_content: yes
  register: result
  until: '"Hello World" in result.content'
  retries: 10
  delay: 1
```

Abuse Regex

#13

```
vars:
  alphabet: "abcdefghijklmnopqrstuvwxyz"
tasks:
  - block:

        - name: change disk names
          replace:
             path: /etc/puppet/example/{{ hostname }}.yaml
             regexp: 'sd{{ alphabet[item | int + 1] }}'
             replace: 'sd{{ alphabet[item | int] }}'
           with_sequence: start=0 end=11
```

Red Hat



What is your favorite Trick?

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat