Ansible Playbooks

Improving Performance

Jacob Hunt
Principal Technical Account Manager (Ansible)

Agenda

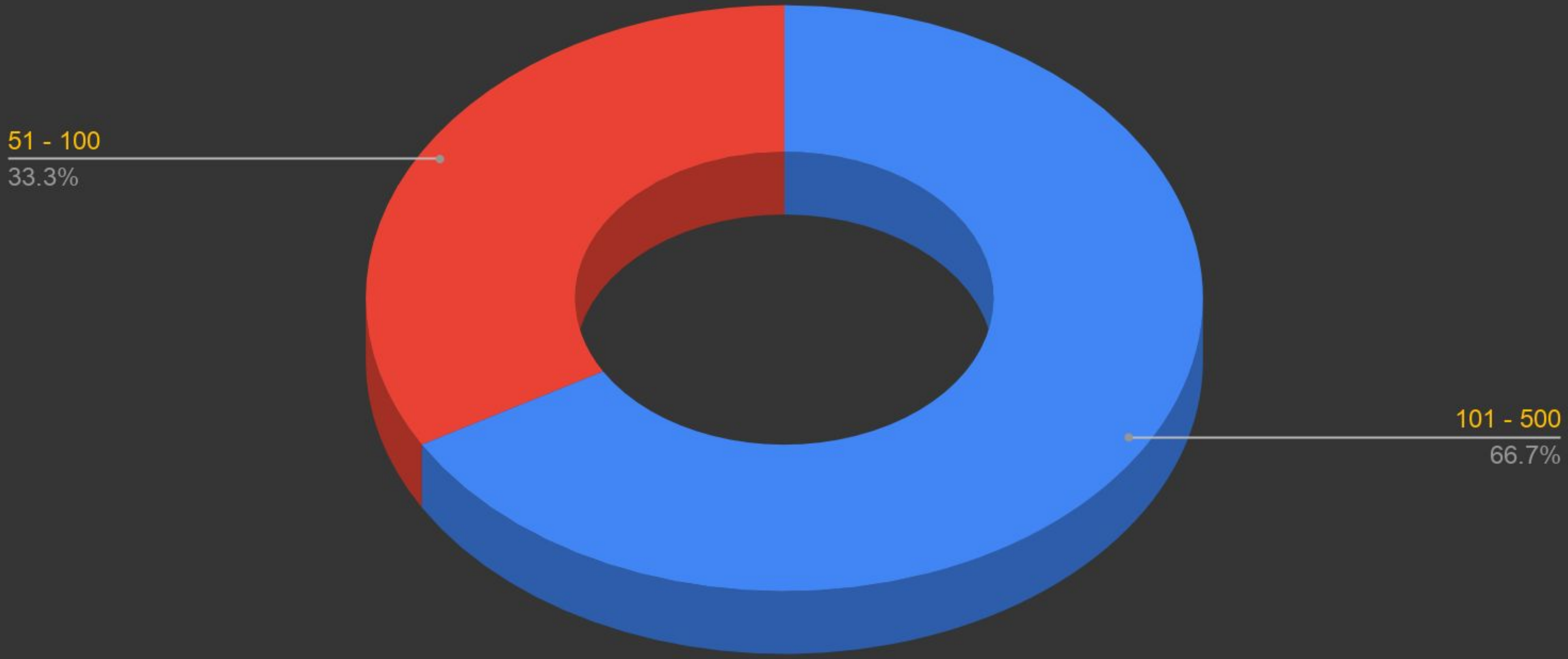
Poll: How large are your inventories?

https://bit.ly/31ck8VF

- Quick performance gains
- Using Forks in Ansible
- Profiling tasks and roles
- Tuning SSH
- Strategy what it does and when you use it
- Are we any faster?
- Instance groups
- Slicing Jobs in Ansible Tower
- Network Tuning
- Faster powershell with Windows

Red Hat

What is the average size of your inventories?



Optimize Playbooks

- Yum calls are incredibly expensive
- Don’t open a shell, unless absolutely necessary
- Don’t gather facts if they are not needed
- Consider replacing shell calls with custom modules

Quick performance boosters!

Test environment

- 3 servers - San Francisco, US
- 3 servers - New York City, US
- 3 servers - Berlin, Germany

```
- name: Install packages
  dnf:
    name: "{{ item }}"
    state: latest
  loop:
    - vim-enhanced
    - zsh
    - tmux
    - sos
    - firefox
```

❌

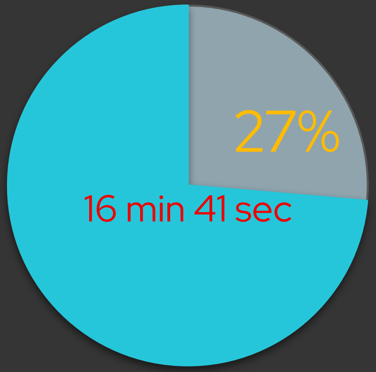
22 min 54 sec

Quick performance boosters!

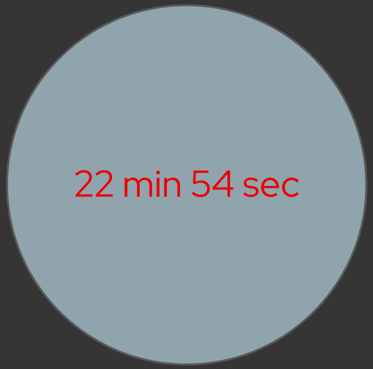
Test environment

- 3 servers - San Francisco, US
- 3 servers - New York City, US
- 3 servers - Berlin, Germany

```
- name: Install packages
  dnf:
    name: "{{ packages }}"
    state: present
```



```
- name: Install packages
  dnf:
    name: "{{ item }}"
    state: latest
  loop:
    - vim-enhanced
    - zsh
    - tmux
    - sos
    - firefox
```

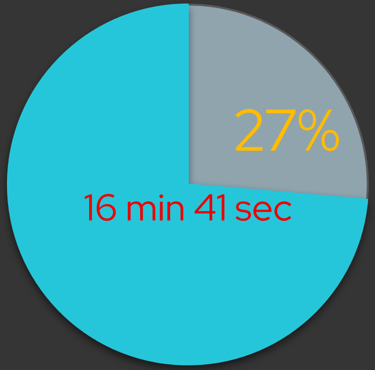


Quick performance boosters!

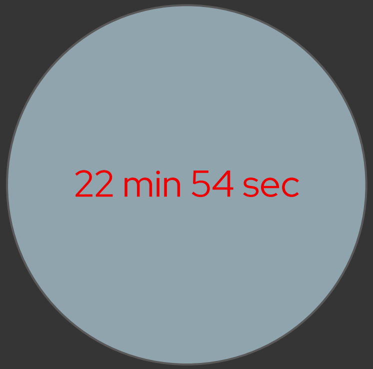
Test environment

- 3 servers - San Francisco, US
- 3 servers - New York City, US
- 3 servers - Berlin, Germany

```yaml
- name: Install packages
  dnf:
    name: "{{ packages }}"
    state: present
```

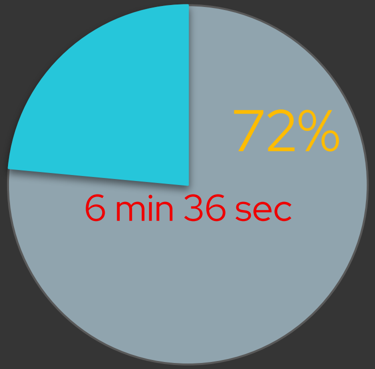


```yaml
- name: Install packages
  dnf:
    name: "{{ item }}"
    state: latest
  loop:
    - vim-enhanced
    - zsh
    - tmux
    - sos
    - firefox
```



```yaml
- name: Install packages
  dnf:
    name: "{{ packages }}"
    state: present
```

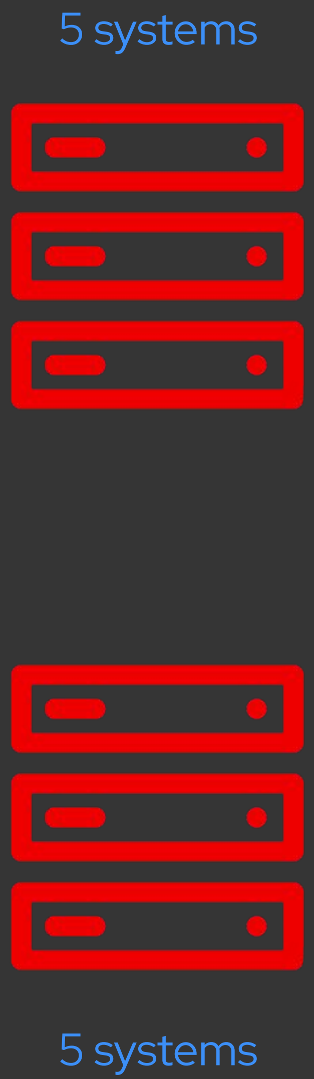
Plus config changes...



How do forks work in Ansible?

Running two tasks, with 5 forks and 10 systems

```
- name: Install httpd
  yum:
    name: httpd
    state: present

- name: Enable and start httpd
  systemd:
    name: httpd
    state: started
    enabled: yes
```

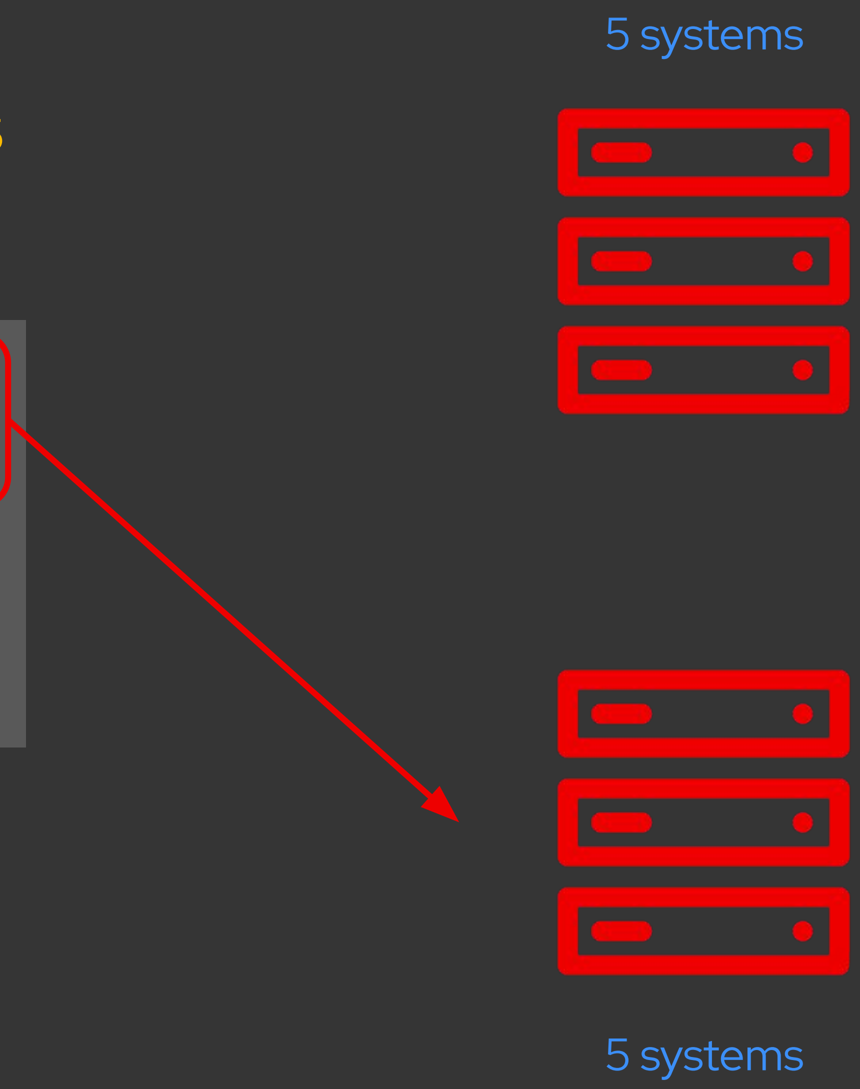


The default amount of forks is set to 5 !

How do forks work in Ansible?

Running two tasks, with 5 forks and 10 systems

```
- name: Install httpd
  yum:
    name: httpd
    state: present

- name: Enable and start httpd
  systemd:
    name: httpd
    state: started
    enabled: yes
```

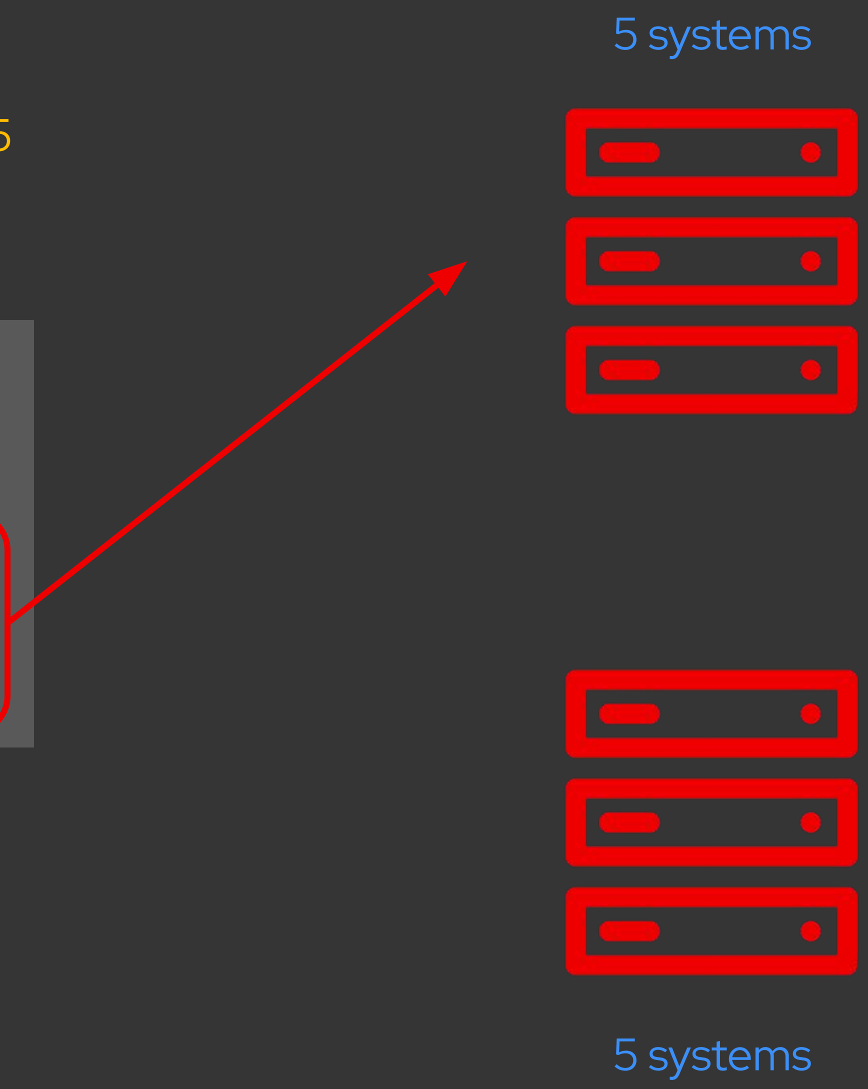


The default amount of forks is set to 5 !

How do forks work in Ansible?

Running two tasks, with 5 forks and 10 systems

```
- name: Install httpd
  yum:
    name: httpd
    state: present

- name: Enable and start httpd
  systemd:
    name: httpd
    state: started
    enabled: yes
```



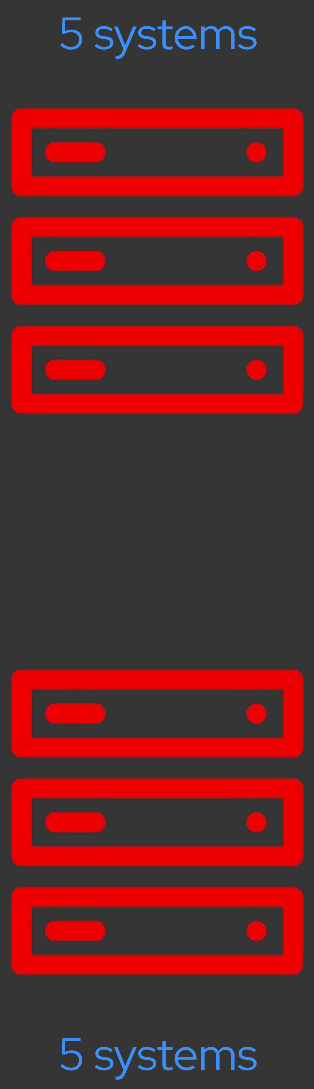
The default amount of forks is set to 5 !

Red Hat

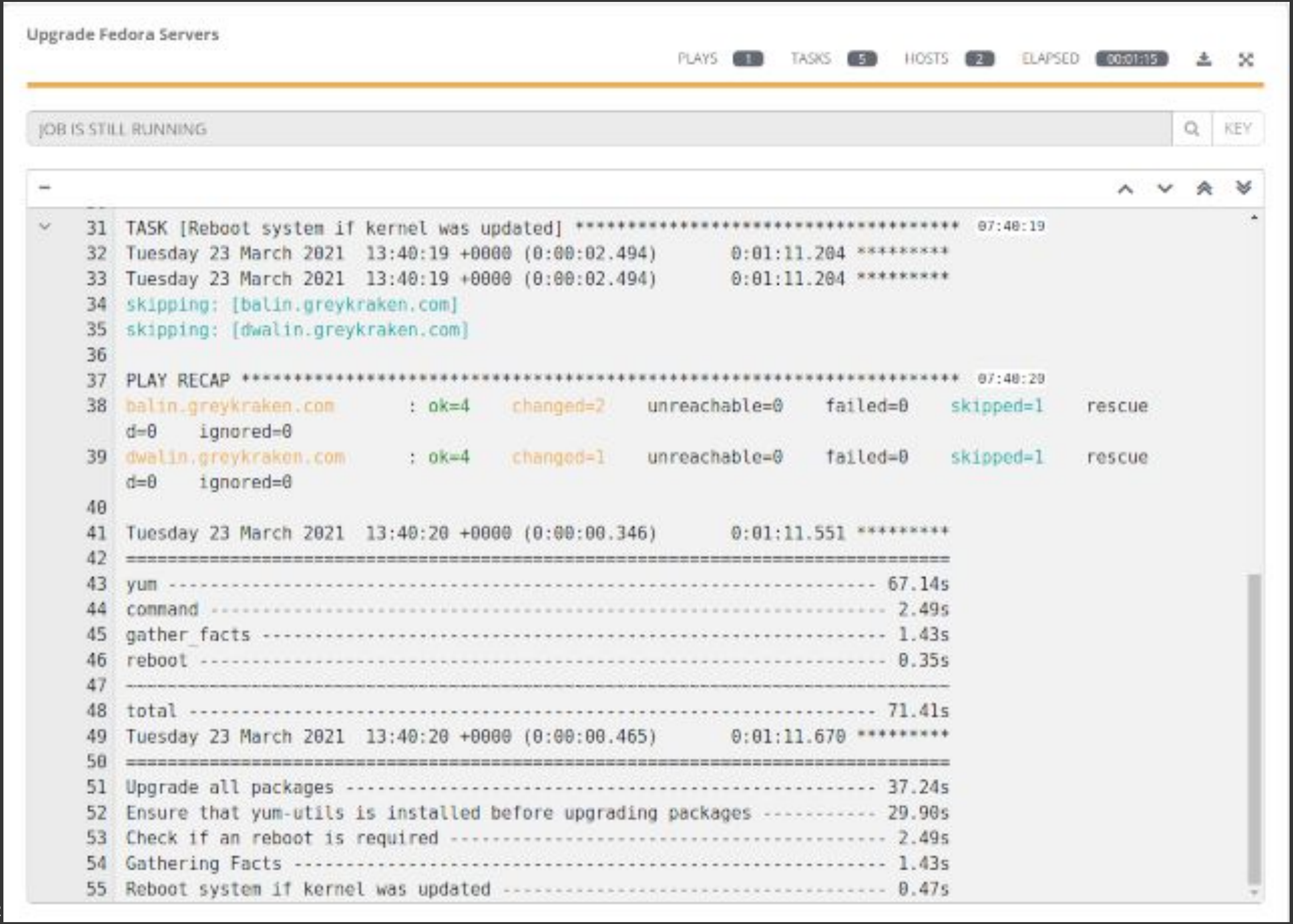
How do forks work in Ansible?

Running two tasks, with 5 forks and 10 systems

```
- name: Install httpd
  yum:
    name: httpd
    state: present

- name: Enable and start httpd
  systemd:
    name: httpd
    state: started
    enabled: yes
```



The default amount of forks is set to 5 !



```
TASK [Reboot system if kernel was updated] ************************************** 07:40:19
Tuesday 23 March 2021  13:40:19 +0000 (0:00:02.494)       0:01:11.204 *********
Tuesday 23 March 2021  13:40:19 +0000 (0:00:02.494)       0:01:11.204 *********
skipping: [balin.greykraken.com]
skipping: [dwalin.greykraken.com]

PLAY RECAP ********************************************************************** 07:40:20
balin.greykraken.com       : ok=4    changed=2    unreachable=0    failed=0    skipped=1    rescue
d=0    ignored=0
dwalin.greykraken.com      : ok=4    changed=1    unreachable=0    failed=0    skipped=1    rescue
d=0    ignored=0

Tuesday 23 March 2021  13:40:20 +0000 (0:00:00.346)       0:01:11.551 *********
===============================================================================
yum -------------------------------------------------------------------- 67.14s
command ----------------------------------------------------------------- 2.49s
gather_facts ------------------------------------------------------------ 1.43s
reboot ------------------------------------------------------------------ 0.35s
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
total ------------------------------------------------------------------ 71.41s
Tuesday 23 March 2021  13:40:20 +0000 (0:00:00.465)       0:01:11.670 *********
===============================================================================
Upgrade all packages --------------------------------------------------- 37.24s
Ensure that yum-utils is installed before upgrading packages ----------- 29.90s
Check if an reboot is required ------------------------------------------ 2.49s
Gathering Facts --------------------------------------------------------- 1.43s
Reboot system if kernel was updated ------------------------------------- 0.47s
```

How to determine what is taking so long.

profile_tasks – Ansible callback plugin for timing individual tasks and overall execution time.

profile_roles – Ansible callback plugin for timing roles.

timer – provides time statics to run roles.

```
[defaults]
callback_whitelist = profile_tasks,
timer
```

Tuning SSH

ansible.cfg

Multiplexing (ControlPersist) and Pipelining

- Multiplexing is enabled by default for 60 seconds
- Pipelining isn’t enabled by default, potentially one of the fastest gains
  - Rather than copy and run the python code, ssh “pipes” it to the node thus saving a connection.

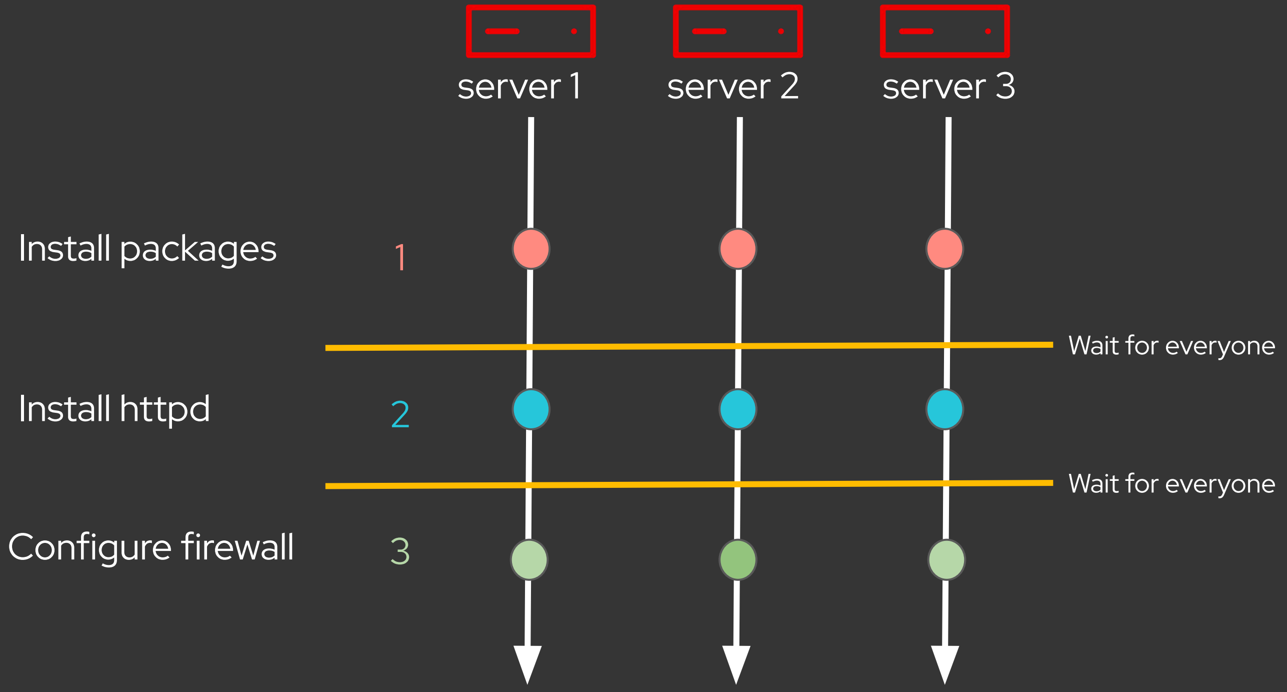
```
[defaults]
callback_whitelist = profile_tasks, profile_roles
forks = 20

[ssh_connection]
pipelining = True
ssh_args = “-o ControlMaster=auto -o ControlPersist=1800s \

-o PreferredAuthentications=publickey”
```

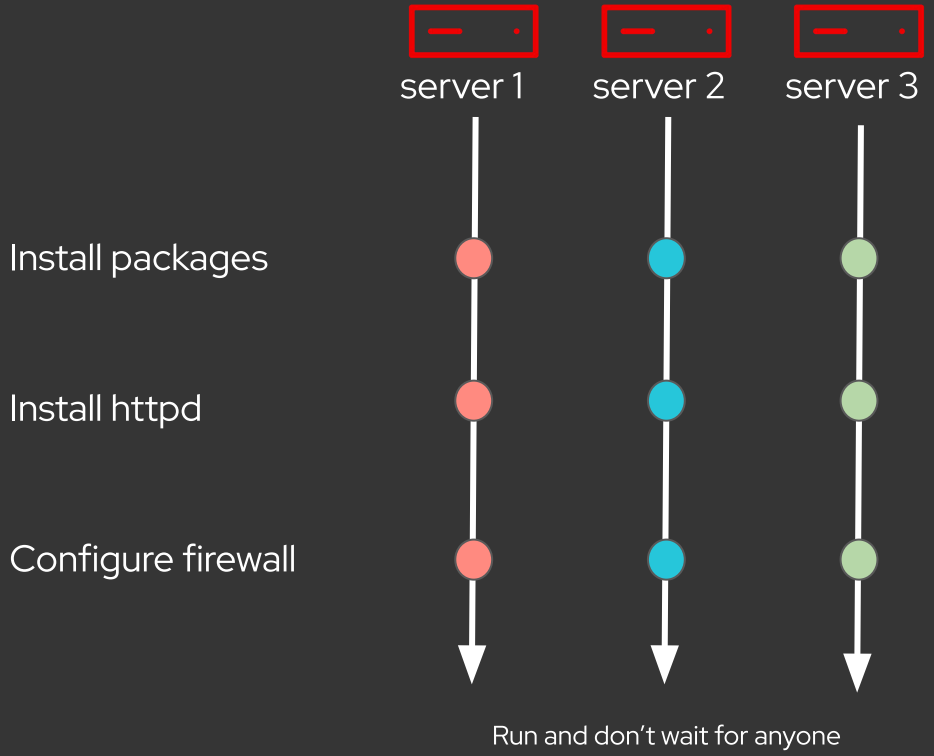
How to verify your config

```
ANSIBLE_FORCE_COLOR(default) = False
ANSIBLE_NOCOLOR(default) = False
ANSIBLE_NOCOWS(default) = False
ANSIBLE_PIPELINING(/var/home/jhunt/vagrant/performance/lamp_simple/ansible.cfg) =
True
ANSIBLE_SSH_ARGS(/var/home/jhunt/vagrant/performance/lamp_simple/ansible.cfg) = -o
ControlPersist=300s -o PreferredAuthentications=publickey
ANSIBLE_SSH_CONTROL_PATH(default) = None
ANSIBLE_SSH_CONTROL_PATH_DIR(default) = ~/.ansible/cp
ANSIBLE_SSH_EXECUTABLE(default) = ssh
ANSIBLE_SSH_RETRIES(default) = 0
ANY_ERRORS_FATAL(default) = False
BECOME_ALLOW_SAME_USER(default) = False
BECOME_PLUGIN_PATH(default) = ['/var/home/jhunt/.ansible/plugins/become',
'/usr/share/ansible/plugins/become']
CACHE_PLUGIN(/var/home/jhunt/vagrant/performance/lamp_simple/ansible.cfg) = jsonfile
CACHE_PLUGIN_CONNECTION(/var/home/jhunt/vagrant/performance/lamp_simple/ansib
le.cfg) = /tmp/ansible-facts
CACHE_PLUGIN_PREFIX(default) = ansible_facts
CACHE_PLUGIN_TIMEOUT(default) = 86400
```

ansible-config dump

Use your strategy...wisely.





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

Let’s check our results

```
[defaults]
callback_whitelist = profile_tasks, profile_roles
forks = 20

[ssh_connection]
pipelining = True
ssh_args = -o ControlPersist=300s -o PreferredAuthentications=publickey
```

```
===============================================================================
db --------------------------------------------------------------------- 70.41s
common ----------------------------------------------------------------- 33.39s
web -------------------------------------------------------------------- 23.14s
gather_facts ------------------------------------------------------------ 1.00s
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
total ----------------------------------------------------------------- 127.94s
===============================================================================
db : Install Mysql package --------------------------------------------- 40.66s
common : Install ntp --------------------------------------------------- 15.27s
web : Install http and php etc ----------------------------------------- 15.08s
db : Start MySQL Service ----------------------------------------------- 12.15s
db : Enable SCL repos --------------------------------------------------- 6.12s
<snip>
Playbook run took 0 days, 0 hours, 2 minutes, 7 seconds
```

```
---
hosts: all
become: true
strategy: free
tasks:
  - name: Install packages
    dnf:
      name: "{{ item }}"
      state: present
    with_items:
      - vim-enhanced
      - zsh
      - tmux
      - sos
      - firefox
```

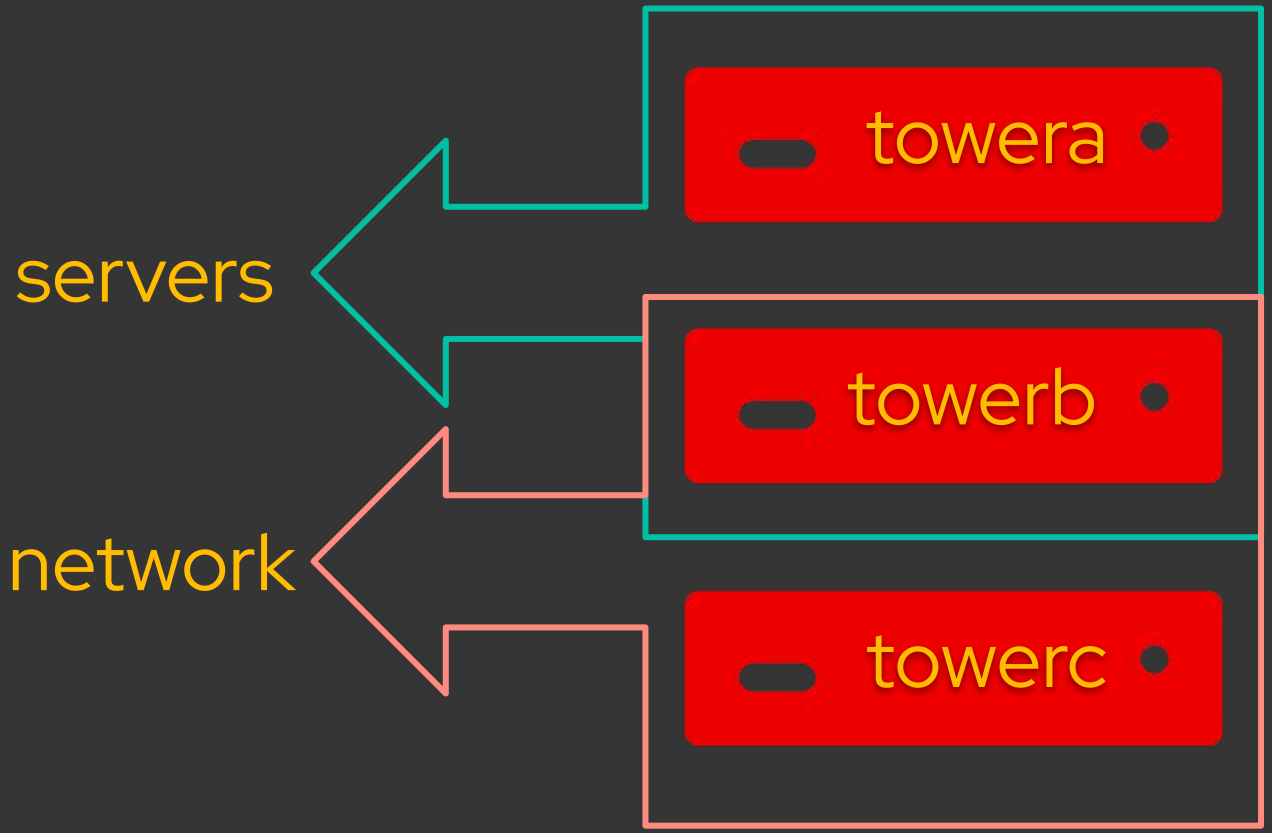
The db role is 40% faster!

The MySQL task is 50% faster!

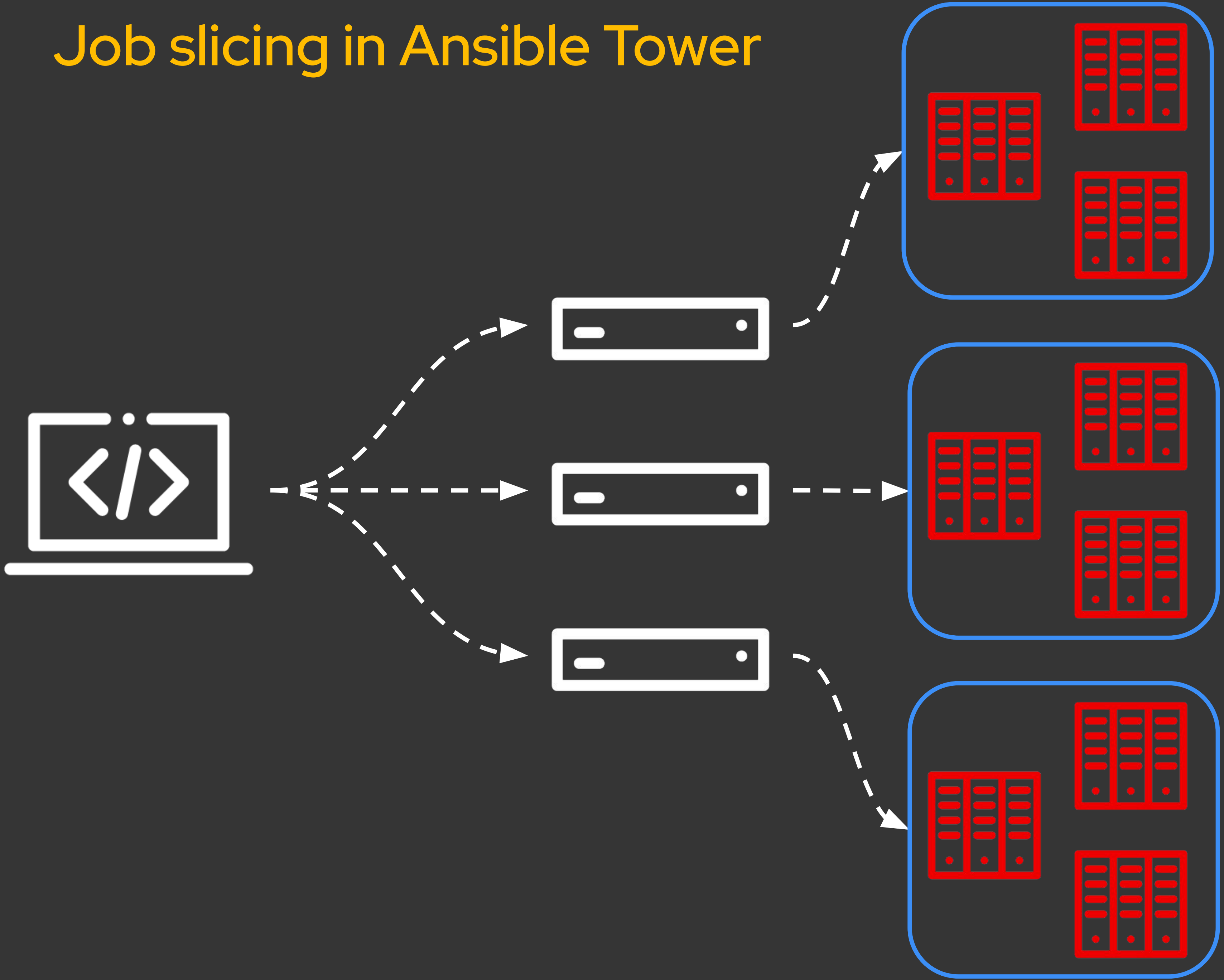
Ansible Tower Instance Groups

Each instance group has its own job queue.

- Any node in the group can take jobs off of that queue.
- Jobs can be assigned to an instance group in three ways.
  - Organization
  - Inventory
  - individual job template



Job slicing in Ansible Tower



The inventory is divided by the number of slices.

Performance Improvements in Automation Controller 4.1 vs. Ansible Tower 3.8

- Average job duration decreased by ~22%
- Job events processing time decreased by ~23%
- Cleanup job runtime decreased by ~98%
- Gather analytics runtime decreased by ~60%

Red Hat

Ansible Network Tuning Tips

strategy: free

With free strategy, available forks are used to execute tasks on each host as quickly as possible. Even if an earlier task is still running on one host, Ansible executes later tasks on other hosts. The free strategy uses available forks more efficiently.

show running

Show running command is the most resource intensive command to execute on a network device, because of the way queries are handled by the network OS. Using the command will significantly slow performance especially on large devices.

ProxyCommand

Ansible must open a new SSH connection for every task. To maximize the performance benefits of the persistent connection types avoid using jump hosts whenever possible.

--forks

The more forks you allow, the more memory and processing power Ansible will use. Since most network tasks are run on the control host, this means your system can quickly become cpu or memory bound.

Red Hat

Ansible also does Windows

To speed up the startup of PowerShell by around 10x, run the following PowerShell snippet in an Administrator session.

```
function Optimize-PowershellAssemblies {
  # NGEN powershell assembly, improves startup time of powershell by 10x
  $old_path = $env:path
  try {
    $env:path =
[Runtime.InteropServices.RuntimeEnvironment]::GetRuntimeDirectory()
    [AppDomain]::CurrentDomain.GetAssemblies() | % {
      if (! $_.location) {continue}
      $Name = Split-Path $_.location -leaf
      if ($Name.startswith("Microsoft.PowerShell.")) {
        Write-Progress -Activity "Native Image Installation" -Status "$name"
        ngen install $_.location | % {"`t$_"}
      }
    }
  } finally {
    $env:path = $old_path
  }
}
Optimize-PowershellAssemblies
```

https://bit.ly/3olD5zq

Red Hat

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat