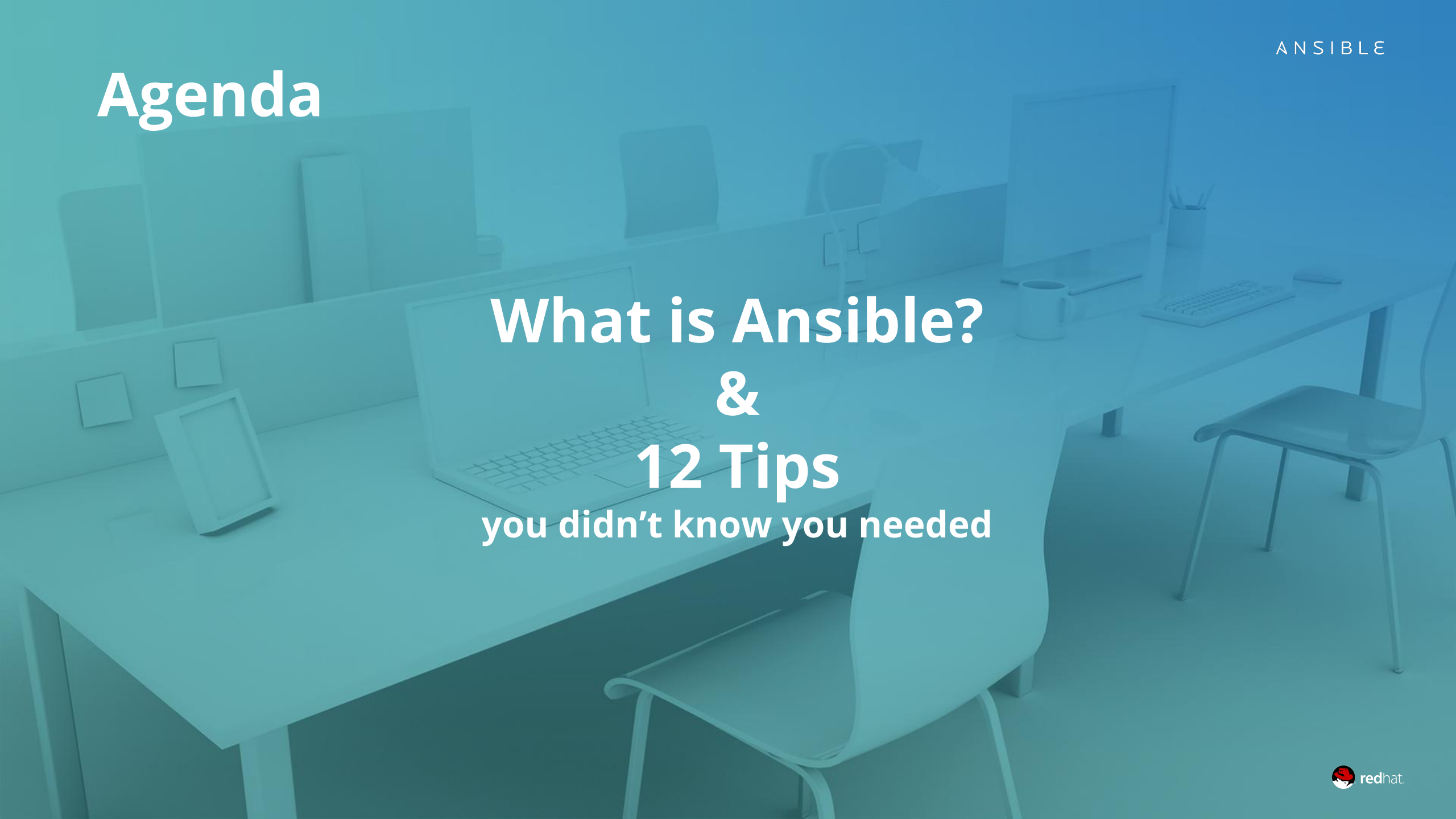
ANSIBLE

Tips and Tricks

Mike Dahlgren - miked@redhat.com - Chief Architect

redhat.

Agenda



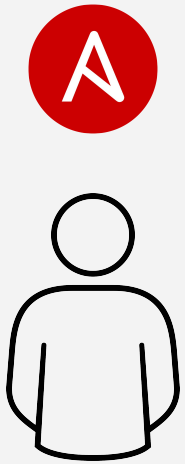
What is Ansible?
&
12 Tips
you didn't know you needed

Red Hat
Ansible
Automation

What is Ansible?

Ansible technical introduction and overview

Red Hat



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

Why Ansible?

Simple

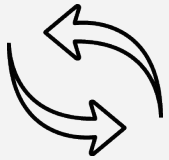
Human readable automation

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

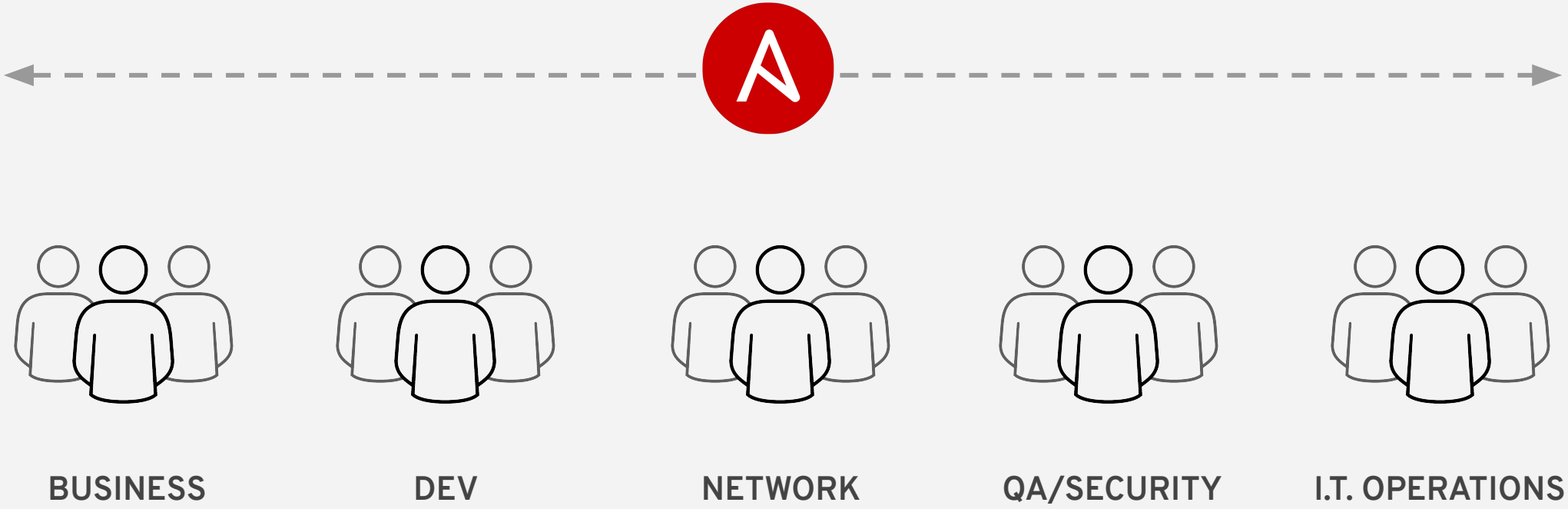
No agents to exploit or update

Get started immediately

More efficient & more secure

Red Hat

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...

Red Hat

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

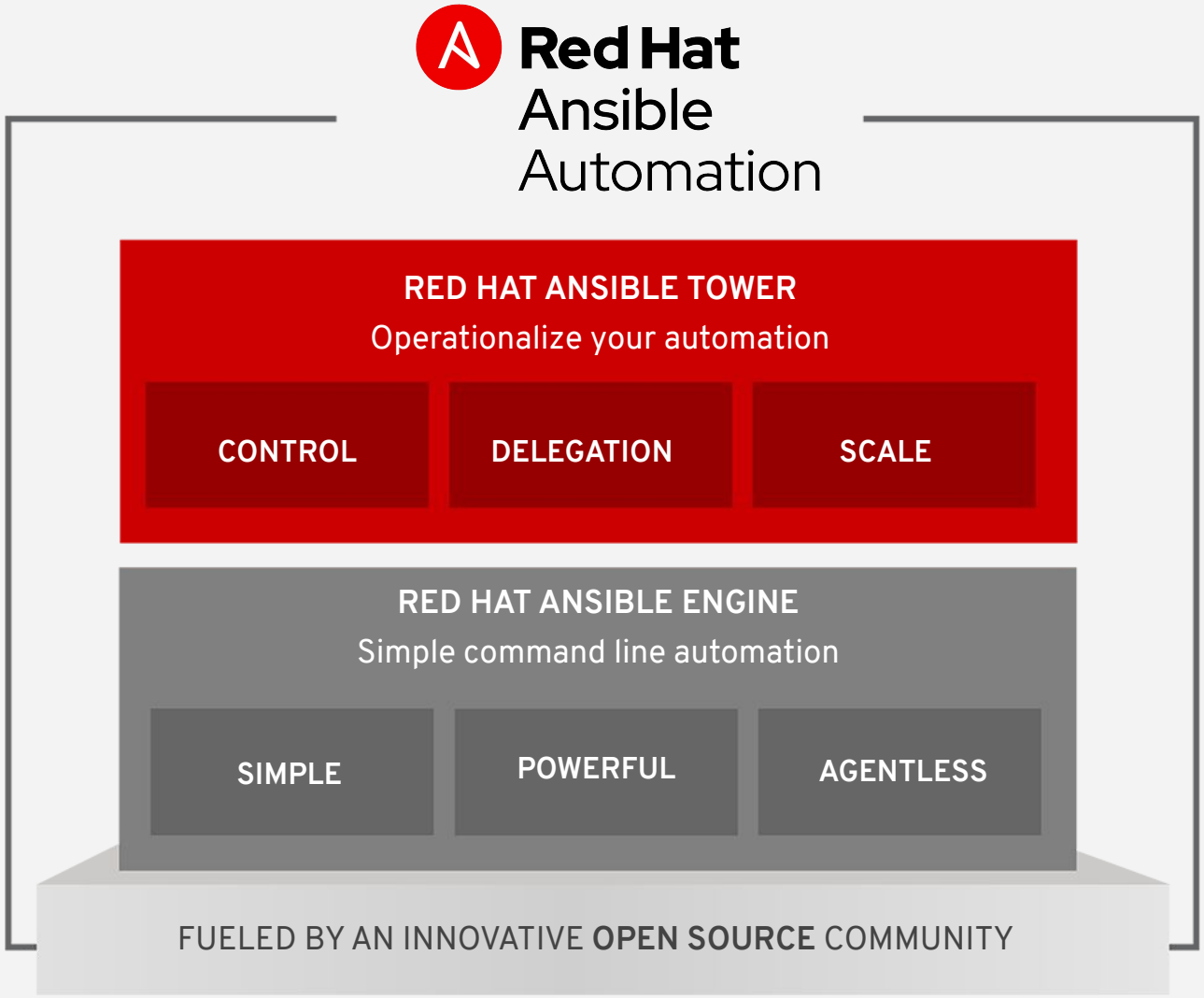
```
---
- name: install and start apache
  hosts: web
  become: yes
  vars:
    http_port: 80

  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
```

What is Ansible Automation?

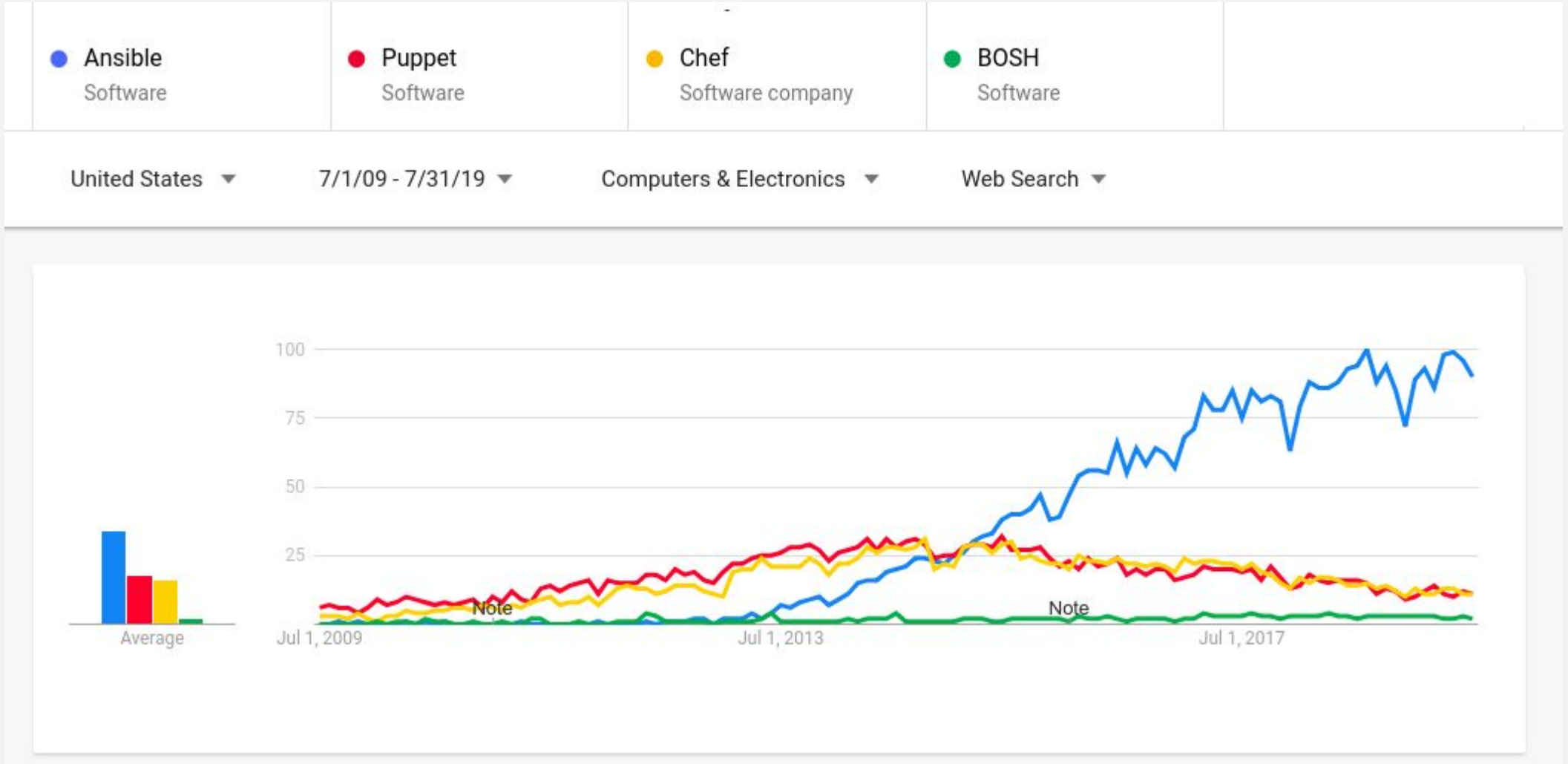
Ansible Automation is the enterprise **framework** for automating across IT operations.

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.



What are the results?



* Google Trends

Now for the Tips

Using Ansible Interactively
Using Ansible in Playbooks

Red Hat

Using Ansible interactively

Ad-hoc commands solve simple tasks at cloud scale

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

(1)

Removing Files & Directories

REMOVING A FILE FROM A SERVER

Easy for one File:

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

REMOVING ALL THE FILES AND DIRECTORIES

```
- 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
```

(2)

Forking background processes from the command line

TIME LIMITING BACKGROUND OPERATIONS

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"

(3)

Running Commands in Parallel

PARALLELISM AND STRATEGIES COMMANDS

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:
  ...
```

(4)

Overloading the Ansible config

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
```

Tips for Playbooks

Better faster easier

(5)

Give everything a name, and put them in first!

Red Hat

GIVE EVERYTHING A NAME!

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

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

GIVE EVERYTHING A NAME!

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

```
PLAY [Setup localhost]
**********************************************************
TASK [Create User]
**********************************************************
[...]
```

(6)

Always use the full YAML syntax

Red Hat

BOTH WORK, ONE IS BETTER!

Use full YAML SYNTAX -

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

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

YAML/ANSIBLE

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

(7)

Don’t store data set facts!

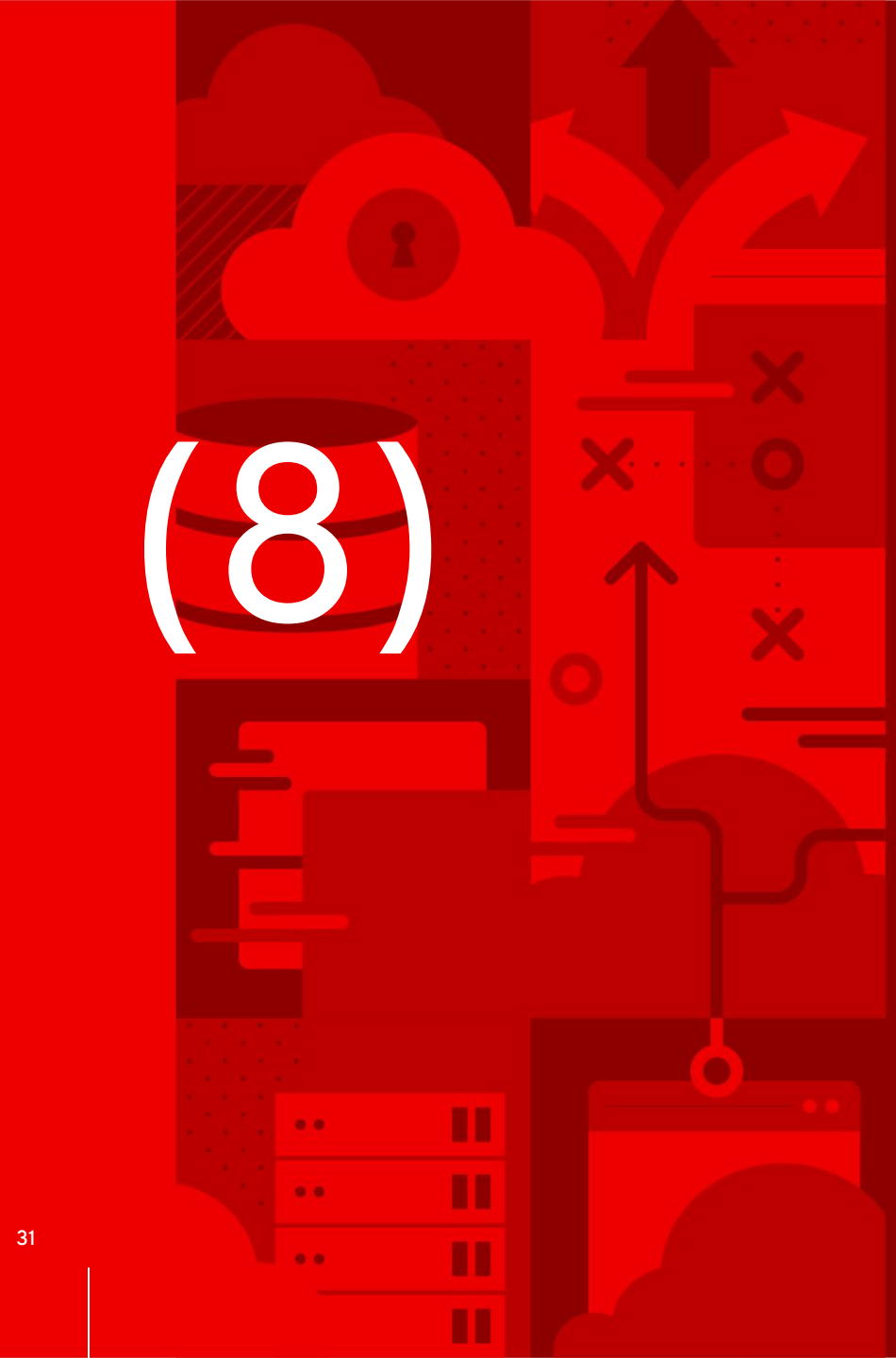
Red Hat

STORE FACTS ON SERVERS

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"



(8)

Clean up debugging tasks (Negative verbosity?)

Red Hat

CLEAN UP YOUR DEBUGGING TASKS

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

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

(9)

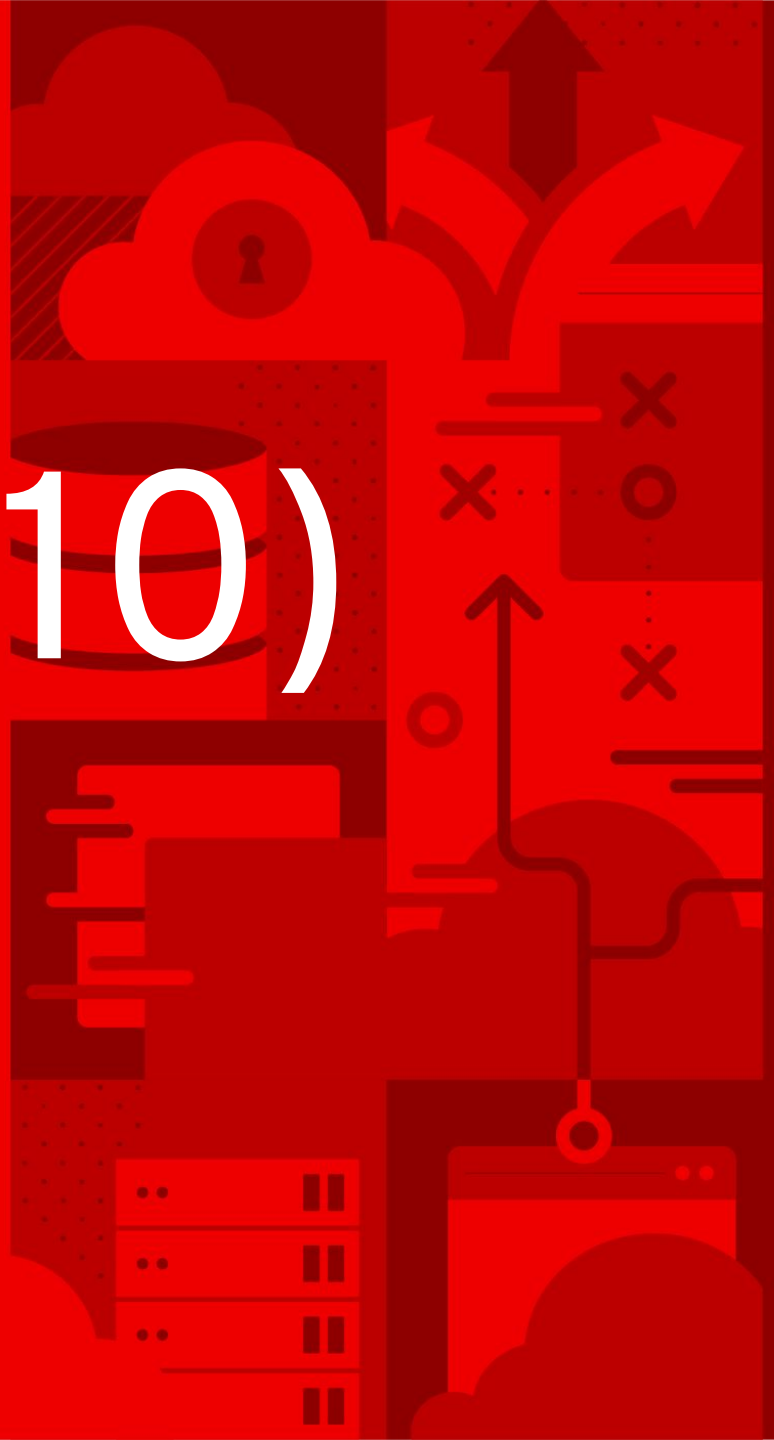
Use smoke tests (always check end services)

Red Hat

DO NOT JUST START SERVICES -- USE SMOKE TESTS

```
- 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
```

(10)



Make life easier with patterns

Red Hat

USE PATTERNS QUICKLY AND EFFECTIVELY

Wildcards work

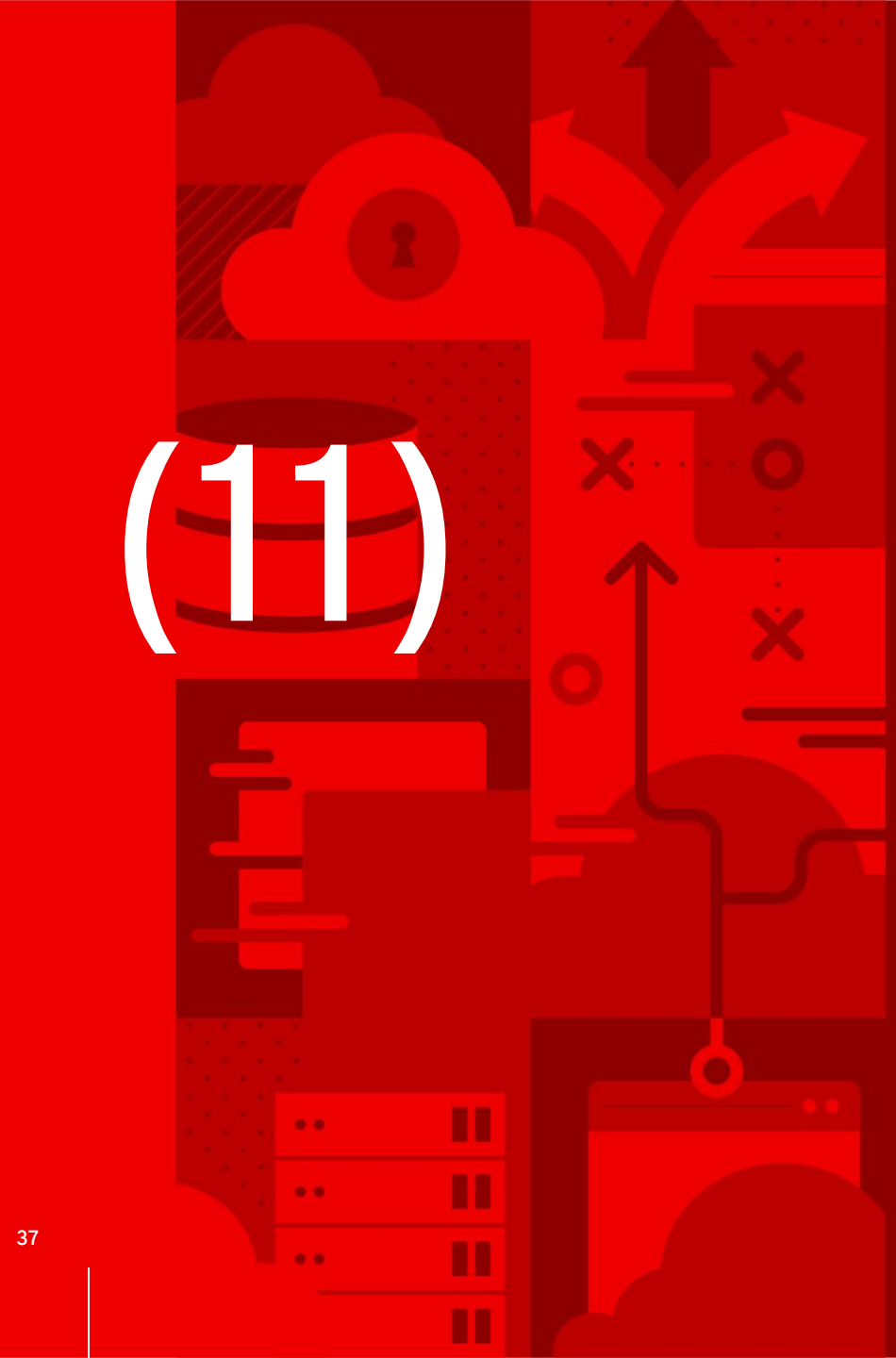
one*.com:dbservers

So can Regex

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

But would this work?

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



(11)

Abuse Regex

Red Hat

CHANGE THE UNCHANGEABLE WITH REGEX

```
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
```

(12)

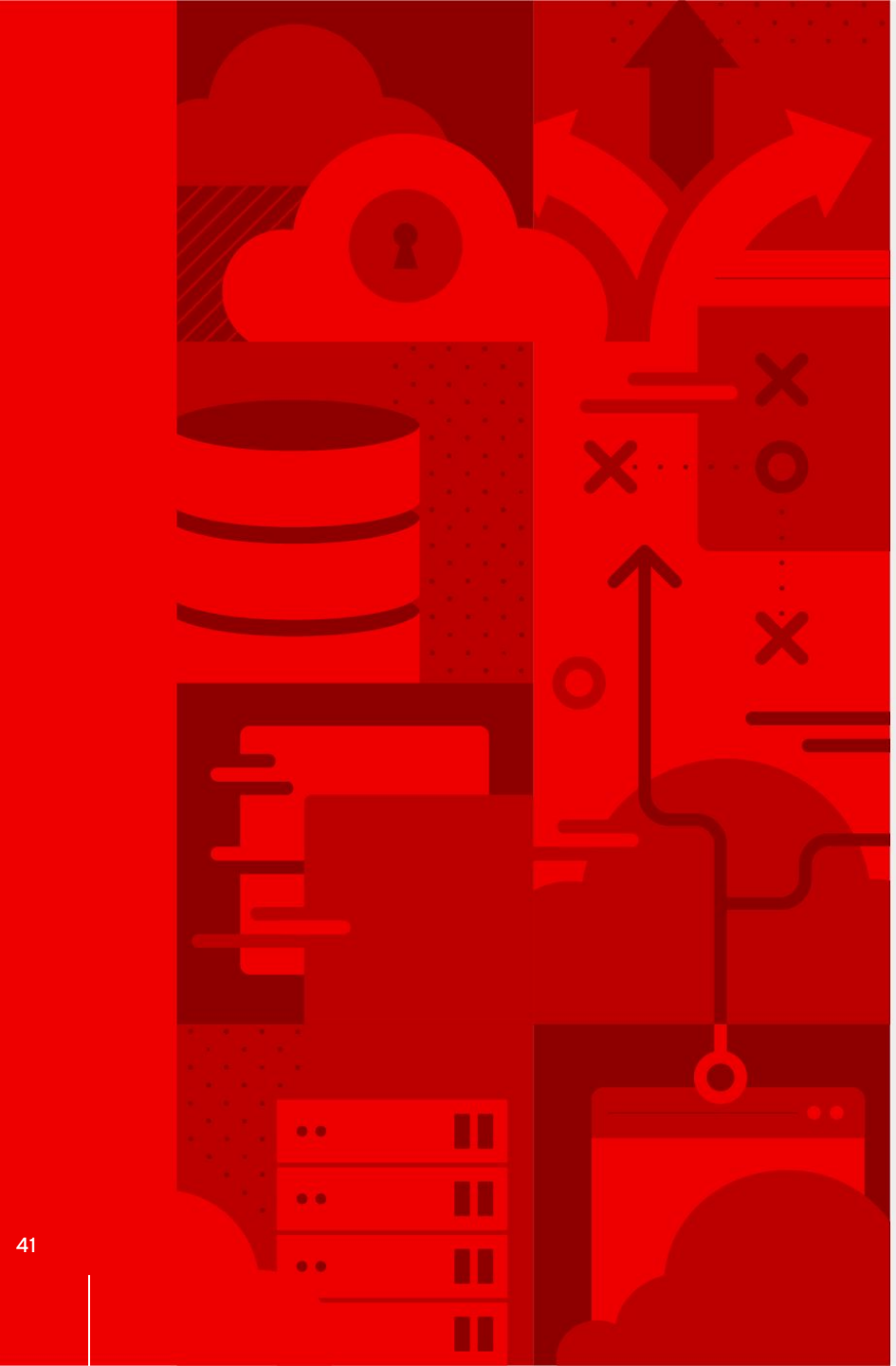
Disable warnings

Red Hat

QUIET COMMANDS AND DISABLE WARNINGS

```
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
```



What is your favorite Trick?

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat