Ansible for Windows

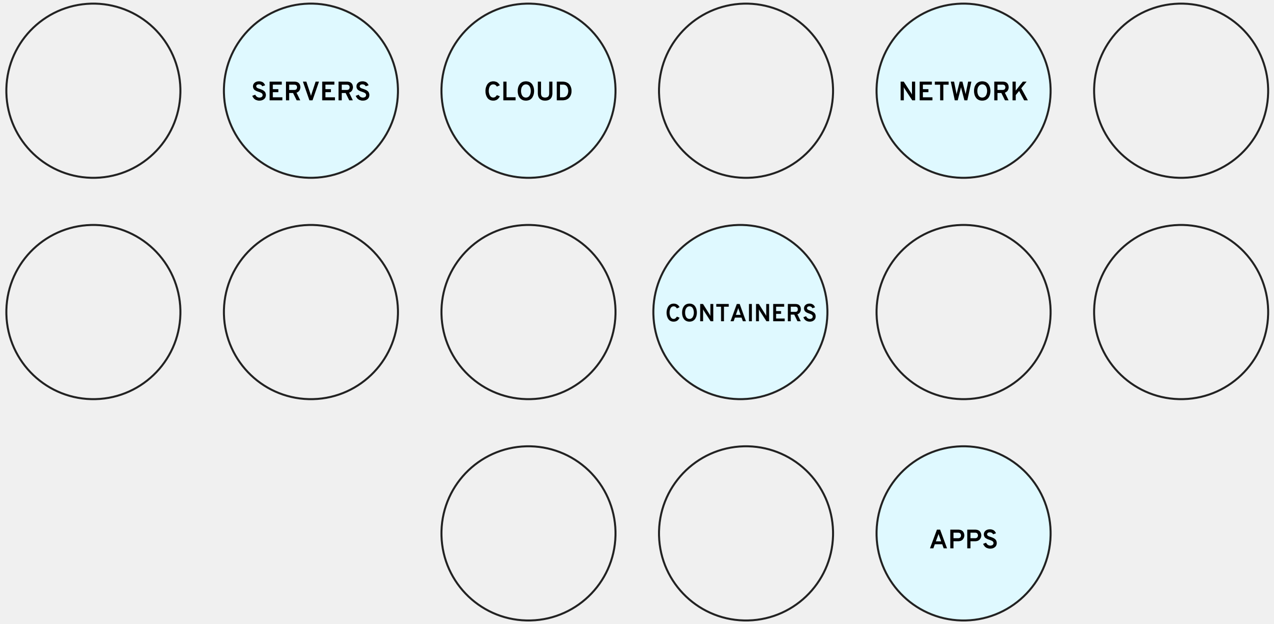
Getting Started

Yvo Wiskerke
Sr. Business Development Manager - Red Hat

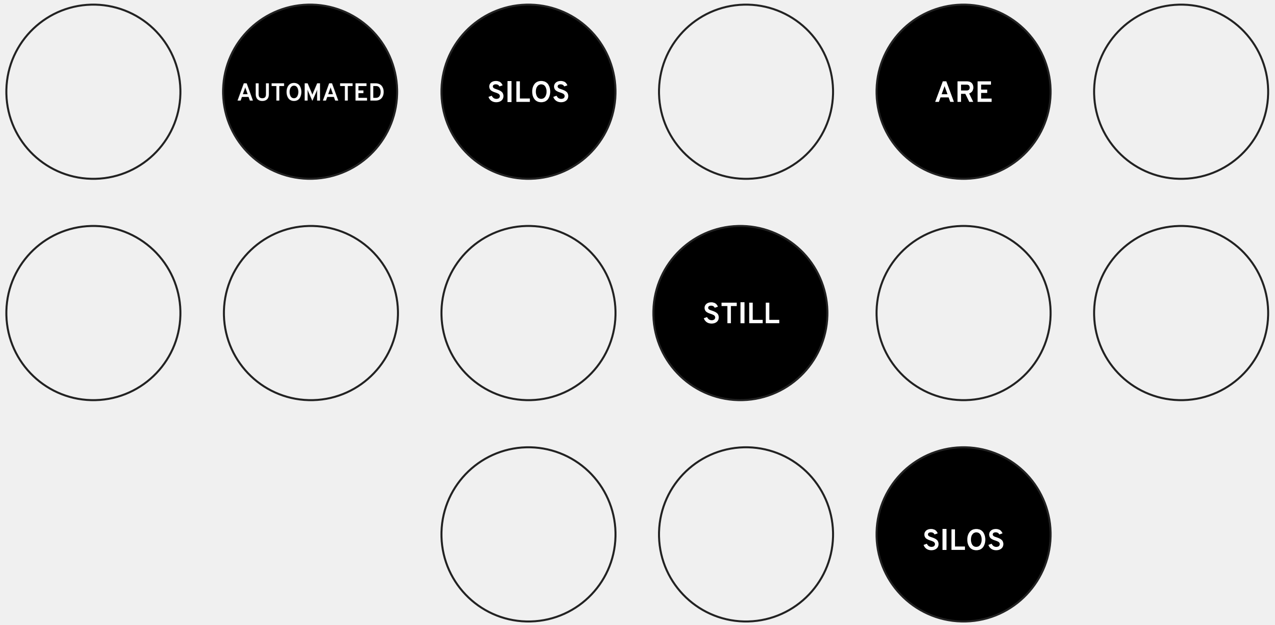
THE WORLD IS AUTOMATING

Those who succeed in automation will win

AUTOMATION IN ENTERPRISE IT TODAY



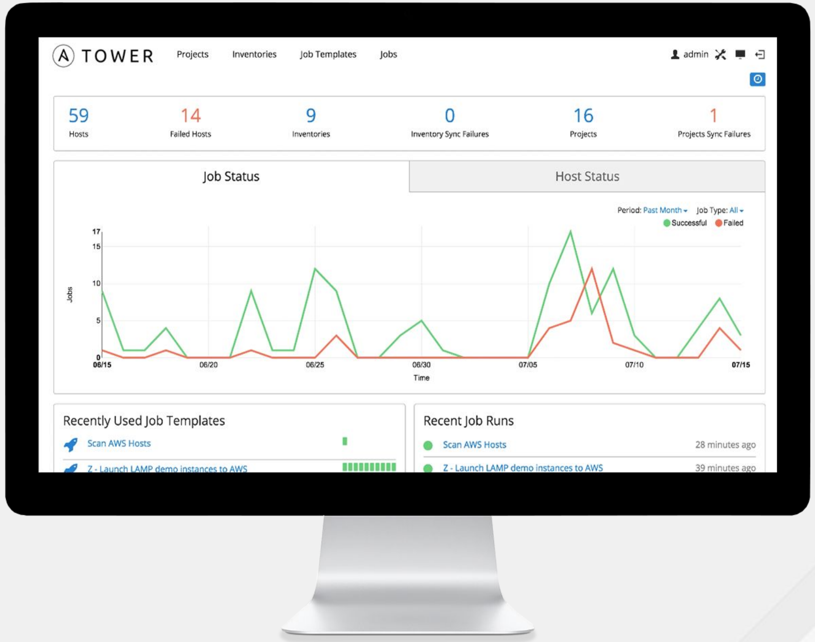
AUTOMATION IN ENTERPRISE IT TODAY



WHAT IS ANSIBLE AUTOMATION?

Ansible is an open source community project sponsored by Red Hat. It’s a **simple automation language** that can perfectly describe IT application environments in **Ansible Playbooks**.

Ansible Tower is an **enterprise framework** for controlling, securing and managing your Ansible automation with a **UI and RESTful API.**



59	14	9	0	16	1
Hosts	Failed Hosts	Inventories	Inventory Sync Failures	Projects	Projects Sync Failures

Recently Used Job Templates

- Scan AWS Hosts
- Z - Launch LAMP demo instances to AWS

Recent Job Runs

Scan AWS Hosts	28 minutes ago
Z - Launch LAMP demo instances to AWS	39 minutes ago

v1 - Set config file to use on boot
1. Write multiple configuration files
• For each environment/region
2. Inspect metadata on boot and use the matching config file

31,000+
Stars on GitHub

1900+
Ansible modules

500,000+
Downloads a month

THE ANSIBLE WAY

CROSS PLATFORM

Agentless support for all major OS variants, physical, virtual, cloud and network devices.

HUMAN READABLE

Perfectly describe and document every aspect of your application environment.

PERFECT DESCRIPTION OF APPLICATION

Every change can be made by Playbooks, ensuring everyone is on the same page.

VERSION CONTROLLED

Playbooks are plain-text. Treat them like code in your existing version control.

DYNAMIC INVENTORIES

Capture all the servers 100% of the time, regardless of infrastructure, location, etc.

ORCHESTRATION PLAYS WELL WITH OTHERS

Every change can be made by Playbooks, ensuring everyone is on the same page.

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

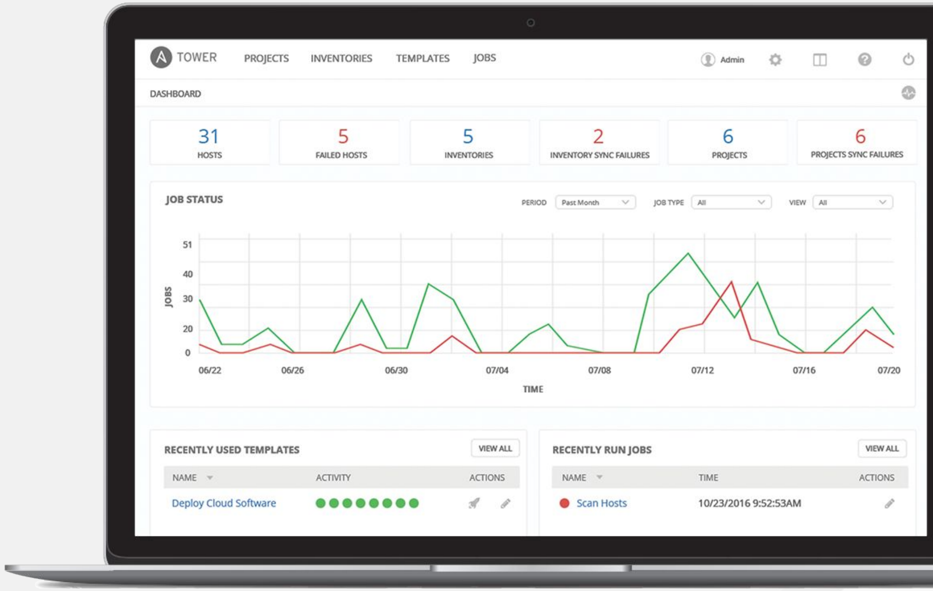
No agents to exploit or update

Get started immediately

More efficient & more secure

ANSIBLE TOWER by Red Hat®

- Build & manage dynamic inventory
- Roles-Based Access Control
- Workflows
- Ongoing Compliance
- Running Playbooks at Scale
- RESTful API



31	5	5	2	6	6
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECTS SYNC FAILURES

RECENTLY USED TEMPLATES

NAME	ACTIVITY	ACTIONS
Deploy Cloud Software	●●●●●●●●	

RECENTLY RUN JOBS

NAME	TIME	ACTIONS
● Scan Hosts	10/23/2016 9:52:53AM	

WHAT CAN I DO WITH 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...

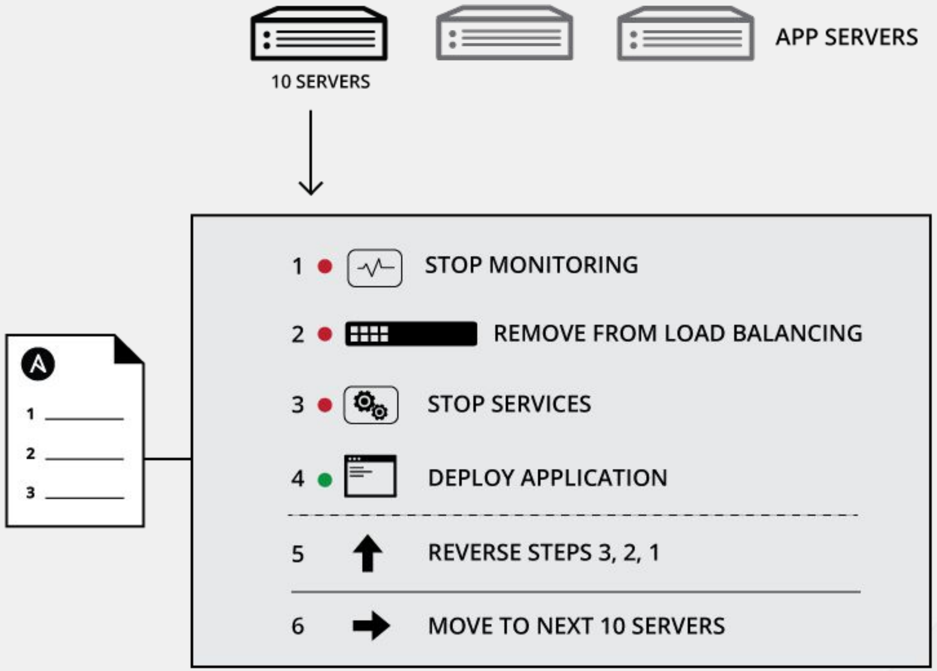
Firewalls	Load Balancers	Applications	Containers	Clouds
Servers	Infrastructure	Storage	Network Devices	**And more...**

WHY IS AUTOMATION IMPORTANT?

Your applications and systems **are more than just collections of configurations**. They're a finely tuned and **ordered list** of tasks and processes that result in **your working application**.

Ansible can do it all:

- Provisioning
- App Deployment
- Configuration Management
- Multi-tier Orchestration

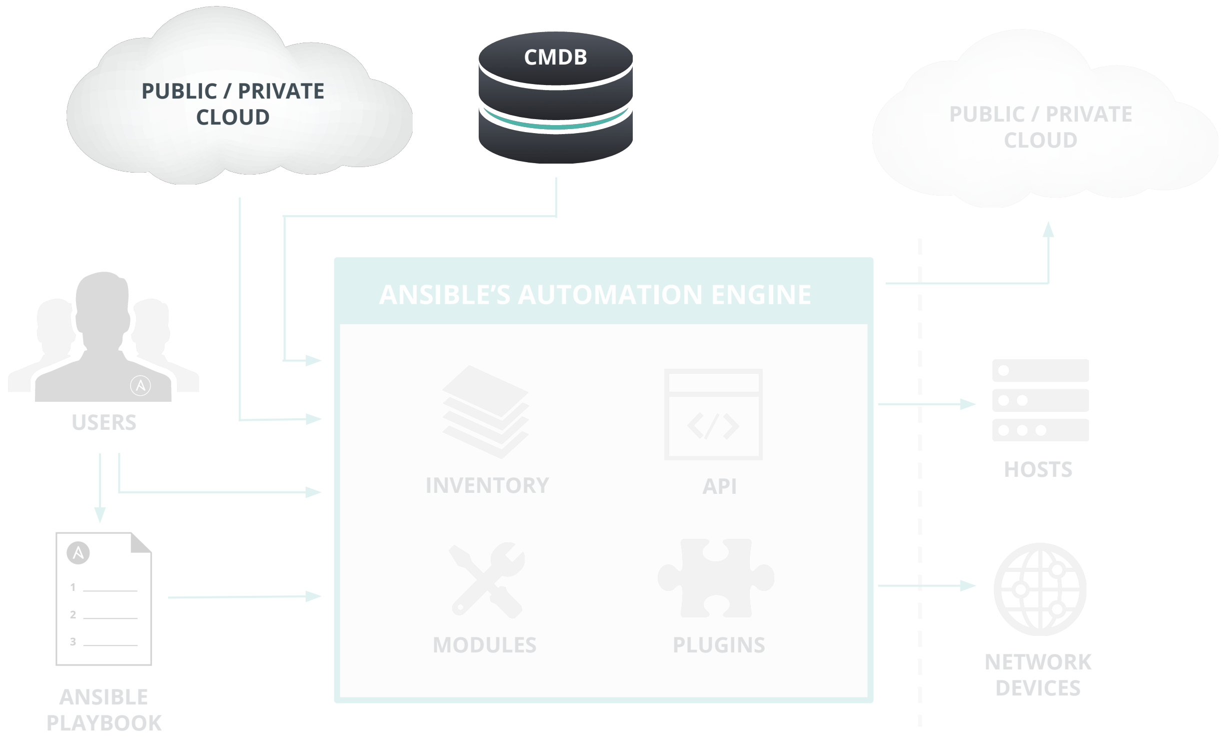


WHY AUTOMATE?

- We all have to do more with less
- Consistently deliver predictable results faster
- Increase number of deployments, reduce time between deployments
- Innovate faster

ANSIBLE

HOW ANSIBLE WORKS



CLOUD

OpenStack, VMware, EC2, Rackspace, GCE, Azure, Spacewalk, Hanlon, Cobbler

CUSTOM CMDB

redhat.

ANATOMY OF A PLAYBOOK

```
- hosts: windows
```

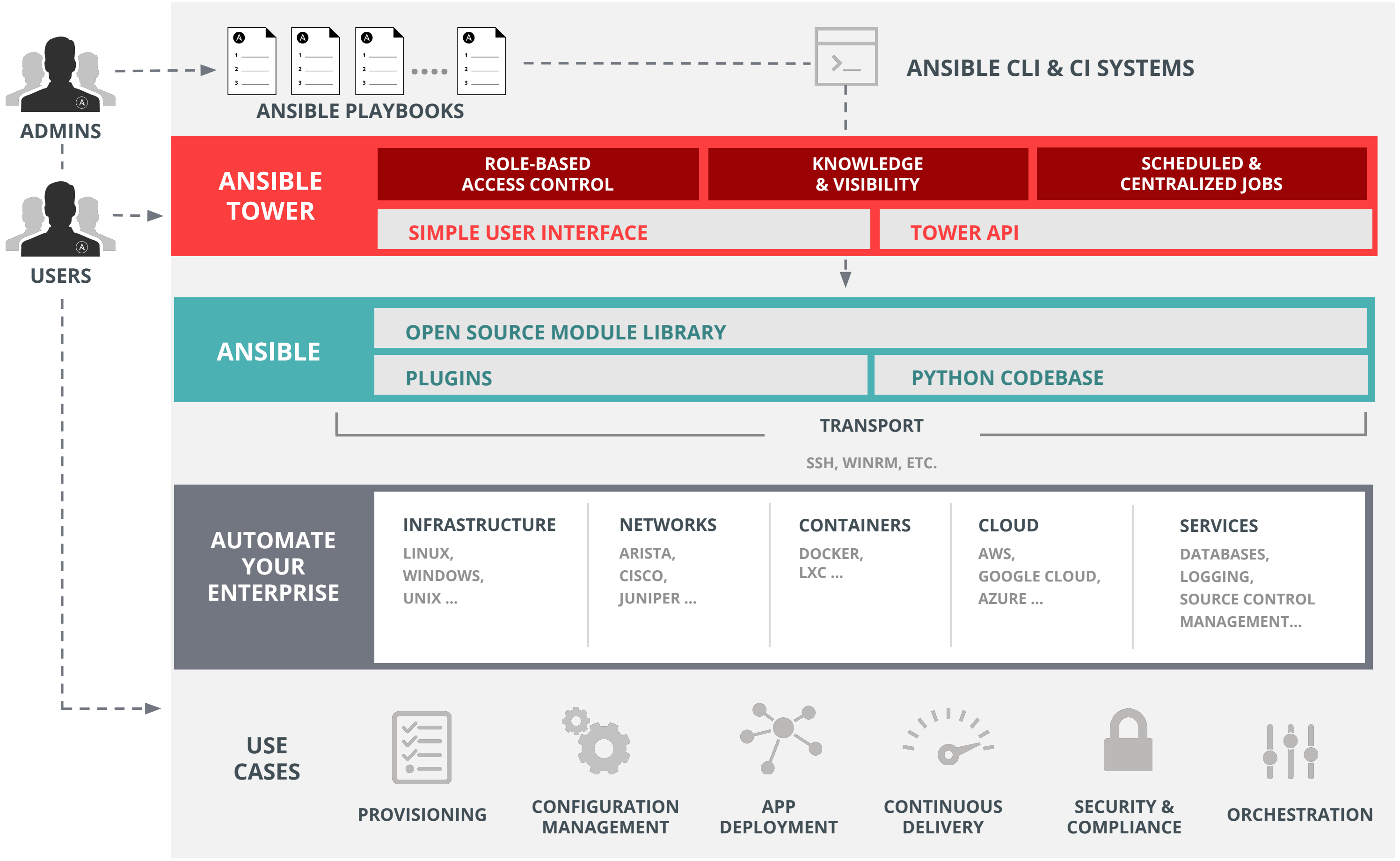
Inventory

```
  vars:
    network_name_servers:
      - 8.8.8.8
      - 8.8.4.4
```

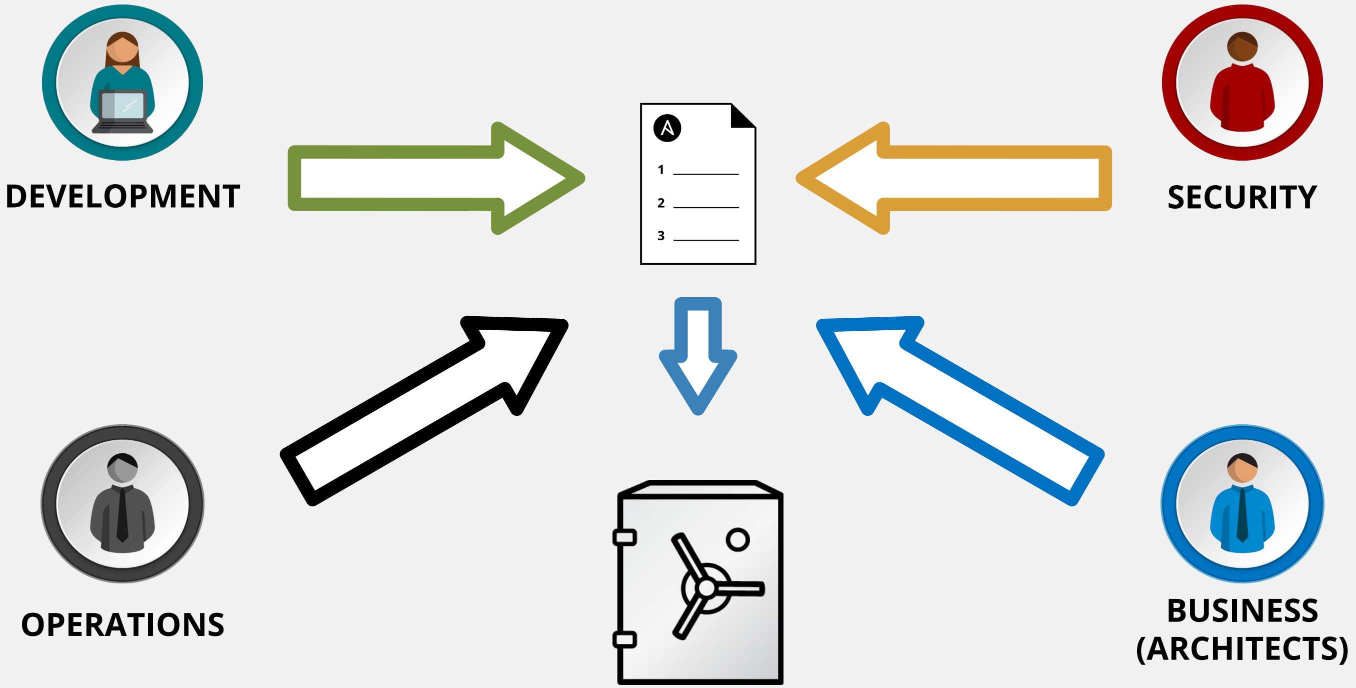
Variables

```
  tasks:
    - name: Configure the dns for all interfaces
      win_dns_client:
        adapter_names: "*"
        Ipv4_addresses: "{{ network_name_servers }}"
```

The task to perform

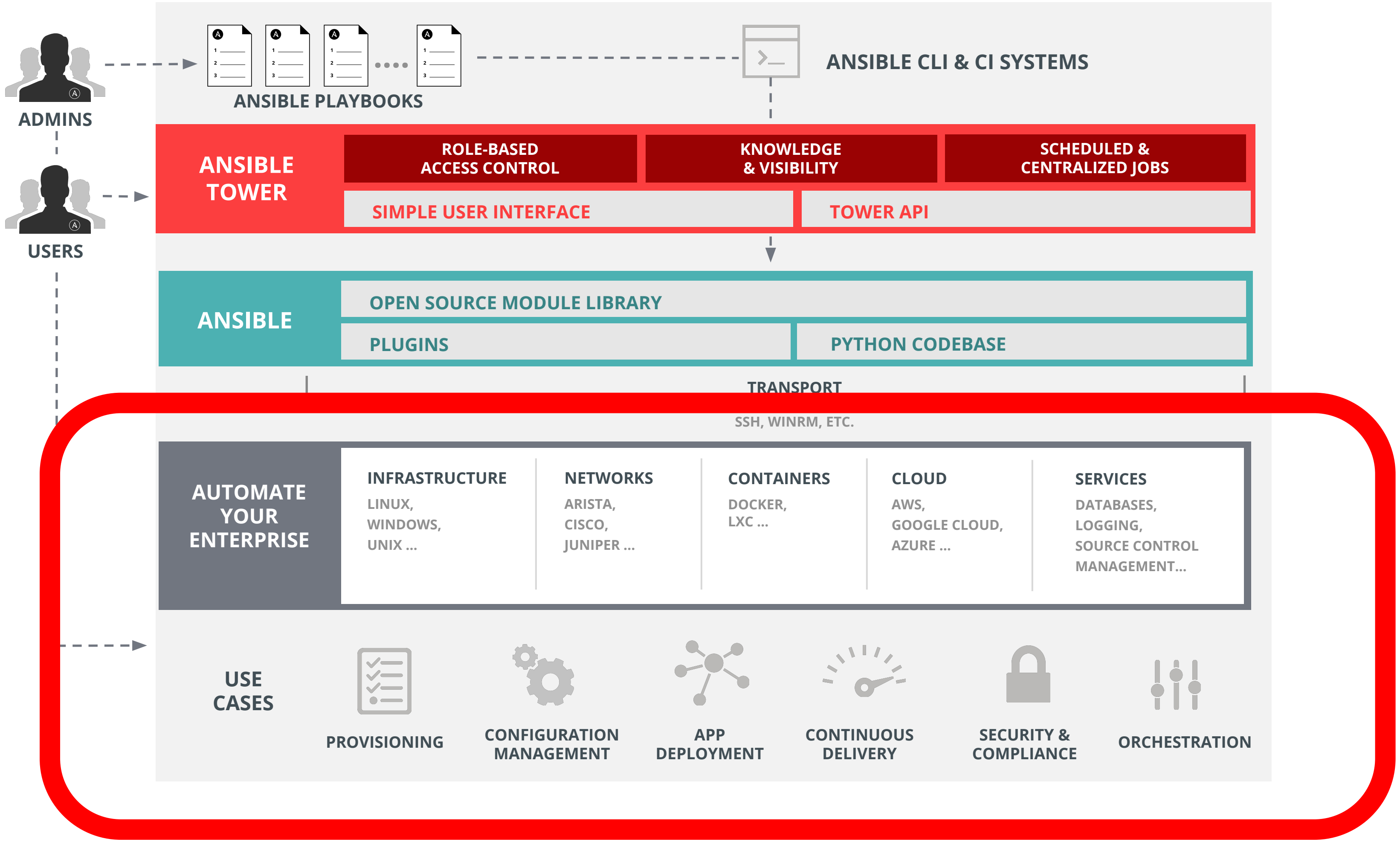

ADMINS
USERS
ANSIBLE PLAYBOOKS
ANSIBLE CLI & CI SYSTEMS
ANSIBLE TOWER
ROLE-BASED ACCESS CONTROL
KNOWLEDGE & VISIBILITY
SCHEDULED & CENTRALIZED JOBS
SIMPLE USER INTERFACE
TOWER API
ANSIBLE
OPEN SOURCE MODULE LIBRARY
PLUGINS
PYTHON CODEBASE
TRANSPORT
SSH, WINRM, ETC.
AUTOMATE YOUR ENTERPRISE
INFRASTRUCTURE
LINUX, WINDOWS, UNIX ...
NETWORKS
ARISTA, CISCO, JUNIPER ...
CONTAINERS
DOCKER, LXC ...
CLOUD
AWS, GOOGLE CLOUD, AZURE ...
SERVICES
DATABASES, LOGGING, SOURCE CONTROL MANAGEMENT...
USE CASES
PROVISIONING
CONFIGURATION MANAGEMENT
APP DEPLOYMENT
CONTINUOUS DELIVERY
SECURITY & COMPLIANCE
ORCHESTRATION

Communicate with Playbooks



WHY ANSIBLE FOR WINDOWS?

- Common objections
  a. I already use Powershell
  b. I already have System Center
  c. I use other tools to automate my windows estate


ADMINS
USERS
ANSIBLE PLAYBOOKS
ANSIBLE CLI & CI SYSTEMS
ANSIBLE TOWER
ROLE-BASED ACCESS CONTROL
KNOWLEDGE & VISIBILITY
SCHEDULED & CENTRALIZED JOBS
SIMPLE USER INTERFACE
TOWER API
ANSIBLE
OPEN SOURCE MODULE LIBRARY
PLUGINS
PYTHON CODEBASE
TRANSPORT
SSH, WINRM, ETC.
AUTOMATE YOUR ENTERPRISE
INFRASTRUCTURE
LINUX, WINDOWS, UNIX ...
NETWORKS
ARISTA, CISCO, JUNIPER ...
CONTAINERS
DOCKER, LXC ...
CLOUD
AWS, GOOGLE CLOUD, AZURE ...
SERVICES
DATABASES, LOGGING, SOURCE CONTROL MANAGEMENT...
USE CASES
PROVISIONING
CONFIGURATION MANAGEMENT
APP DEPLOYMENT
CONTINUOUS DELIVERY
SECURITY & COMPLIANCE
ORCHESTRATION

INFRASTRUCTURE AGNOSTIC USE CASES

- Security & OS Hardening
- Updates and Patches
- User management
- Configuration management
- Software deployment

NOT SSH

- WinRM (HTTP-based remote shell protocol)
- Non-interactive logon
- Different connection plugin
- Microsoft OpenSSH?

POWERSHELL

- All Windows modules in Ansible written in Powershell
- Unlike Python, “just there” on modern Windows
- We can use .NET
- Powershell 3+, Windows 7/Server 2008+
- Access to the DSC universe via win_dsc

WINDOWS HOST REQUIREMENTS

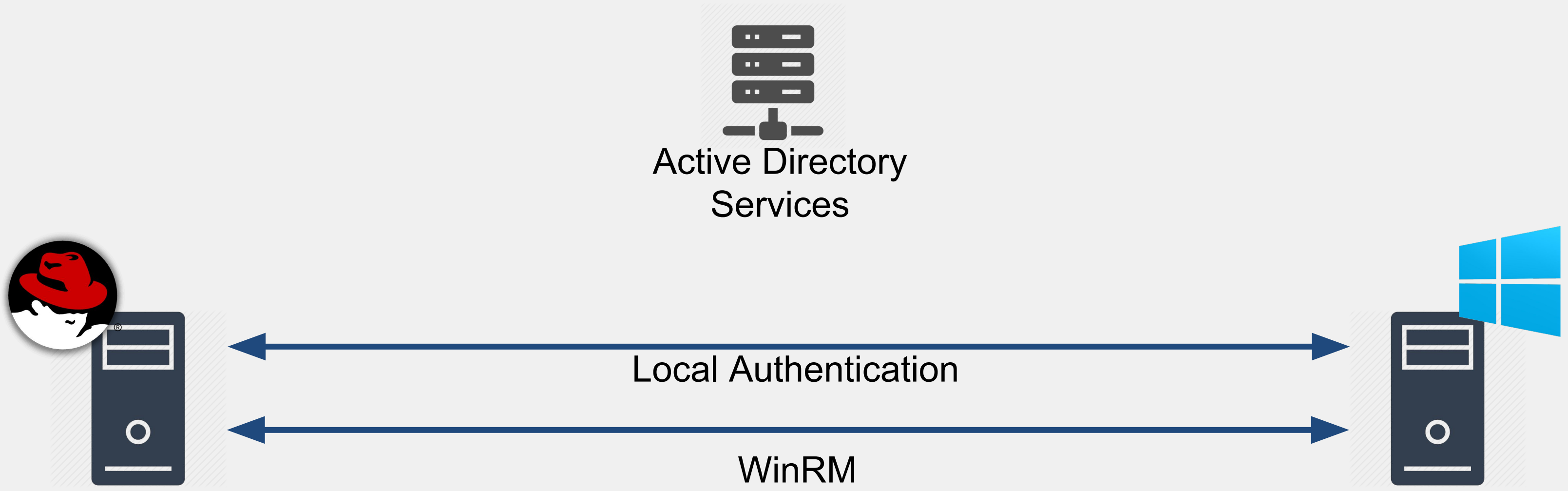
- Supported desktop OSs include Windows 7, 8.1, and 10
- Supported server OSs are Windows Server 2008, 2008 R2, 2012, 2012 R2, and 2016.
- Ansible requires PowerShell 3.0 or newer and at least .NET 4.0 to be installed on the Windows host.
- A WinRM listener should be created and activated.

For more details: https://docs.ansible.com/ansible/latest/user_guide/windows_setup.html

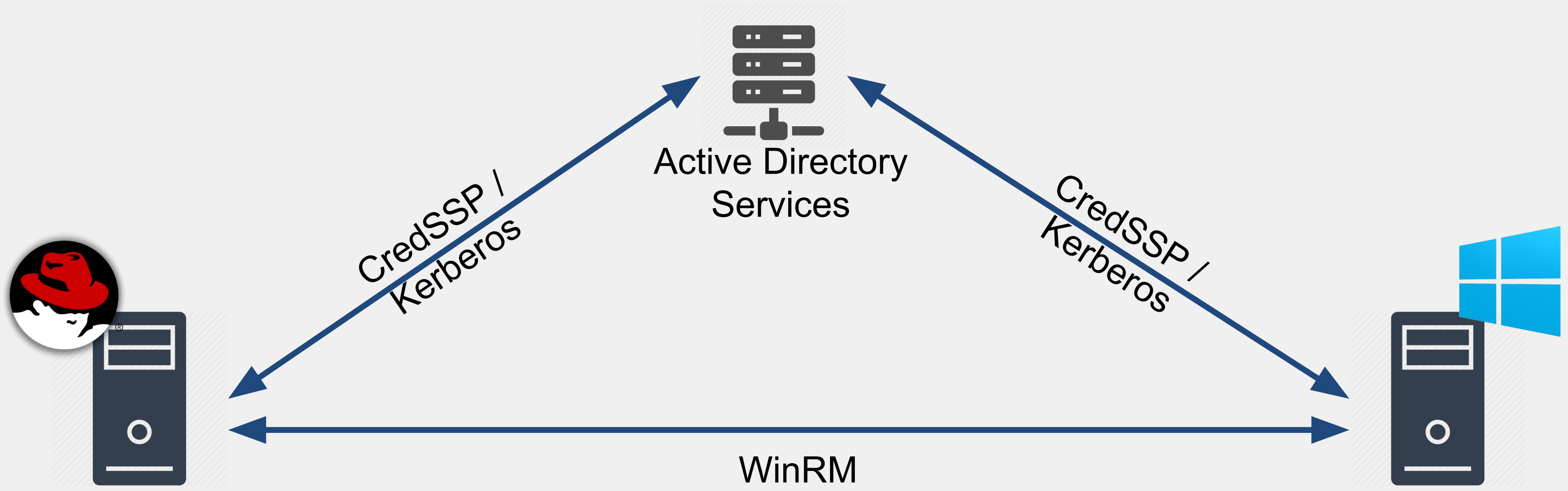
GETTING ANSIBLE READY FOR WINDOWS

- Install ansible and kerberos as per documentation
- Configure the Kerberos by setting the default realm and adding your domain controller information in the linux kerberos configuration file.
- Configure the inventory file with client machine info
- Ready to run playbooks

GETTING READY FOR WINDOWS



GETTING READY FOR WINDOWS



AUTHENTICATION OPTIONS

Option	Local Accounts	Active Directory Accounts	Credential Delegation	HTTP Encryption
Basic	Yes	No	No	No
Certificate	Yes	No	No	No
Kerberos	No	Yes	Yes	Yes
NTLM	Yes	Yes	No	Yes
CredSSP	Yes	Yes	Yes	Yes

USE CASE - INSTALLING SOFTWARE

1. Using the *win_chocolatey* module. This sources the program data from the default public *Chocolatey* repository. Internal repositories can be used instead by setting the *source* option.
2. Using the *win_package* module. This installs software using an MSI or .exe installer from a local/network path or URL.
3. Using the *win_command* or *win_shell* module to run an installer manually.

USE CASE - INSTALLING SOFTWARE

```
# install/uninstall with chocolatey
- name: ensure 7-Zip is installed via Chocolatey
  win_chocolatey:
    name: 7zip
    state: present

- name: ensure 7-Zip is not installed via
Chocolatey
  win_chocolatey:
    name: 7zip
    state: absent
```

```
# install/uninstall with win_package
- name: download the 7-Zip package
  win_get_url:
    url: http://www.7-zip.org/a/7z1701-x64.msi
    dest: C:\temp\7z.msi

- name: ensure 7-Zip is installed via win_package
  win_package:
    path: C:\temp\7z.msi
    state: present

- name: ensure 7-Zip is not installed via
win_package
  win_package:
    path: C:\temp\7z.msi
    state: absent
```

USE CASE - SETTING UP USERS/GROUPS

1. The modules *win_user*, *win_group* and *win_group_membership* manage Windows users, groups and group memberships locally.
2. The modules *win_domain_user* and *win_domain_group* manages users and groups in a domain.

USE CASE - SETTING UP USERS/GROUPS

```
- name: create local group to contain new users
  win_group:
    name: LocalGroup
    description: Allow access to C:\Development folder

- name: create local user
  win_user:
    name: '{{item.name}}'
    password: '{{item.password}}'
    groups: LocalGroup
    update_password: no
    password_never_expired: yes
  with_items:
  - name: User1
    password: Password1
  - name: User2
    password: Password2
```

```
- name: ensure each account is created
  win_domain_user:
    name: '{{item.name}}'
    upn: '{{item.name}}@MY.DOMAIN.COM'
    password: '{{item.password}}'
    password_never_expires: no
    groups:
    - Test User
    - Application
    company: Ansible
    update_password: on_create
  with_items:
  - name: Test User
    password: Password
  - name: Admin User
    password: SuperSecretPass01
  - name: Dev User
    password: '@fvr3IbFBujSRh!3hBg%wgFucD8^x8W5'
```

USE CASE - WINDOWS UPDATES

win_updates is used to install multiple updates by category
Basic, synchronous updates that uses configured source (Windows update/WSUS)

win_hotfix can be used to install a single update or hotfix file that has been downloaded locally.

The *win_hotfix* module has a requirement that the DISM PowerShell cmdlets are present (W2K12+)

USE CASE - WINDOWS UPDATES

```
- name: Run Updates then wait 7 mins before reboot
    win_updates:
      category_names:
        - Application
        - CriticalUpdates
        - SecurityUpdates
      whitelist:
        - KB4093120
    reboot: yes
    reboot_timeout: 420

- name: Run Updates, except KB4056892(endless loop)
    win_updates:
      category_names:
        - CriticalUpdates
        - SecurityUpdates
      blacklist:
        - KB4056892
```

```
- name: download KB3172729 for Server 2012 R2
  win_get_url:
    url:
http://download.windowsupdate.com/d/msdownload/update/s
oftware/secu/2016/07/windows8.1-kb3172729-x64_e8003822a
7ef4705cbb65623b72fd3cec73fe222.msu
    dest: C:\temp\KB3172729.msu

- name: install hotfix
  win_hotfix:
    hotfix_kb: KB3172729
    source: C:\temp\KB3172729.msu
    state: present
  register: hotfix_result

- name: reboot host if required
  win_reboot:
  when: hotfix_result.reboot_required
```

Reboots, oh the reboots

- win_reboot action makes managed reboots trivial
- wait_for_connection is just the second half

USE CASE - DESIRED STATE CONFIGURATION

Desired State Configuration, or DSC, is a tool built into PowerShell that can be used to define a Windows host setup through code.

Since Ansible 2.4, the *win_dsc* module has been added and can be used to leverage existing DSC resources when interacting with a Windows host.

For DSC windows host **must** have PowerShell v5.0 or newer installed.

All supported hosts, except for Windows Server 2008 (non R2) can be upgraded to PowerShell v5.

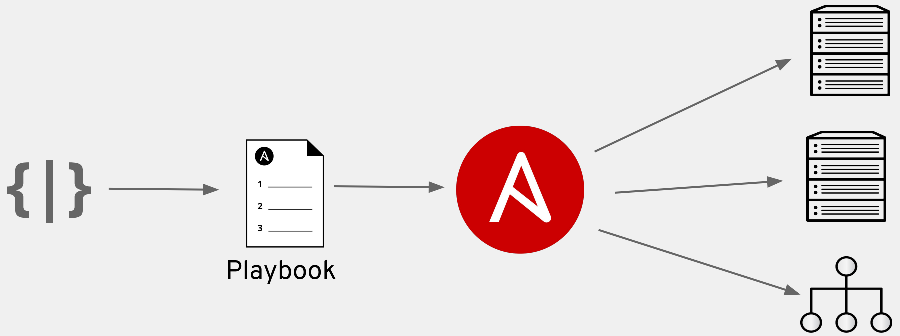
USE CASE - DESIRED STATE CONFIGURATION

```
- name: use win_dsc module with the Registry
DSC resource
  win_dsc:
    resource_name: Registry
    Ensure: Present
    Key:
HKEY_LOCAL_MACHINE\SOFTWARE\ExampleKey
    ValueName: TestValue
    ValueData: TestData
```

```
- name: use win_dsc with
PsDscRunAsCredential to run as a different
user
  win_dsc:
    resource_name: Registry
    Ensure: Present
    Key: HKEY_CURRENT_USER\ExampleKey
    ValueName: TestValue
    ValueData: TestData
    PsDscRunAsCredential_username:
'{{ansible_user}}'
    PsDscRunAsCredential_password:
'{{ansible_password}}'
  no_log: true
```

USE CASE - SECURITY

- Define firewall rules in one variable file
- Apply to many different systems



Policy Abstraction

```
fw_rules:
    - { rule: "public", src_ip: 0.0.0.0/0, dst_ip: 192.133.160.23/32, dst_port: 32400, proto: tcp, action: allow, comment: app1 }
    - { rule: "public", src_ip: 0.0.0.0/0, dst_ip: 192.133.160.23/32, dst_port: 1900, proto: udp, action: allow, comment: app2  }
    - { rule: "public", src_ip: 0.0.0.0/0, dst_ip: 192.133.160.23/32, dst_port: 3005, proto: tcp, action: allow, comment: app3  }
    - { rule: "public", src_ip: 0.0.0.0/0, dst_ip: 192.133.160.23/32, dst_port: 5353, proto: udp, action: allow, comment: app4  }
```

```
  - name: Insert ASA ACL
        asa_config:
          lines:
            - "access-list {{ item.rule
| ipaddr('network') }}{{ item.dst_ip
        with_items: "{{ fw_rules }}"
```

```
- name: Create security rules
  win_firewall_rule:
    name: "{{ item.comment }}"
    remoteport: "{{ item.dst_port }}"
    remoteip: "{{ item.dst_ip }}"
    action: "{{ item.action }}"
    direction: out
    protocol: "{{ item.proto }}"
    state: present
    enabled: yes
  with_items: "{{ fw_rules }}"
```

```
- name: Create security rules
    panos_security_rule:
      operation: "{{ item.action | default (omit) }}"
      rule_name: "{{ item.comment | default (omit) }}"
      service: "{{ item.dst_port | default (omit) }}"
      description: "{{ item.description | default (omit)
      source_zone: "{{ item.rule | default (omit) }}"
      destination_zone: "{{ item.destination_zone | defa
      action: "{{ item.action | default ('allow') }}"
      commit: "{{ item.comment | default (omit) }}"
```

LOTS MORE WINDOWS MODULES

win_acl
win_acl_inheritance
win_audit_policy_system
win_audit_rule
win_certificate_store
win_chocolatey
win_command
win_copy
win_defrag
win_disk_facts
win_disk_image
win_dns_client
win_domain
win_domain_computer
win_domain_controller
win_domain_group
win_domain_membership
win_domain_user
win_dotnet_ngen

win_dsc
win_environment
win_eventlog
win_eventlog_entry
win_feature
win_file
win_file_version
win_find
win_firewall
win_firewall_rule
win_get_url
win_group
win_group_membership
win_hostname
win_hotfix
win_iis_virtualdirectory
win_iis_webapplication
win_iis_webapppool
win_iis_webbinding

win_iis_website
win_lineinfile
win_mapped_drive
win_msg
win_msi (D)
win_nssm
win_owner
win_package
win_pagefile
win_path
win_pester
win_ping
win_power_plan
win_product_facts
win_psexec
win_psmodule
win_rabbitmq_plugin
win_reboot
win_reg_stat

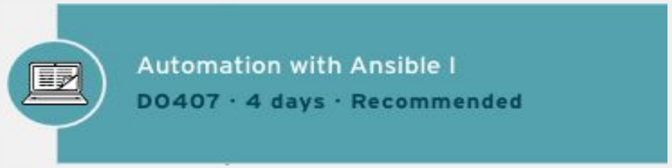
win_regedit
win_region
win_regmerge
win_robocopy
win_route
win_say
win_scheduled_task
win_scheduled_task_stat
win_security_policy
win_service
win_share
win_shell
win_shortcut
win_stat
win_tempfile
win_template
win_timezone
win_toast
win_unzip

win_updates
win_uri
win_user
win_user_right
win_wait_for
win_wakeonlan
win_webpicmd
win_whoami

https://docs.ansible.com/ansible/latest/modules/list_of_windows_modules.html

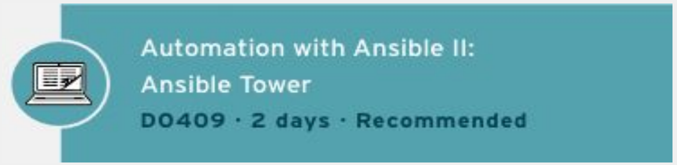
ANSIBLE TRAINING & HANDS ON LABS

- Recommended training



Automation with Ansible I

DO407 · 4 days · Recommended



Automation with Ansible II: Ansible Tower

DO409 · 2 days · Recommended

- 29 November 2018 - Hands On Lab Ansible & Ansible Tower

THANK YOU

redhat.

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos