RED HAT SUMMIT

MANAGE WINDOWS WITH ANSIBLE

The what, the why and the how?

Roland Wolters
Senior Product Marketing Manager
2019-05-09

WHO AM I?

- Roland Wolters
- @liquidat
- > 10 years experience
- Consultant
- Project Manager
- Solution Architect
- Product Marketing Manager

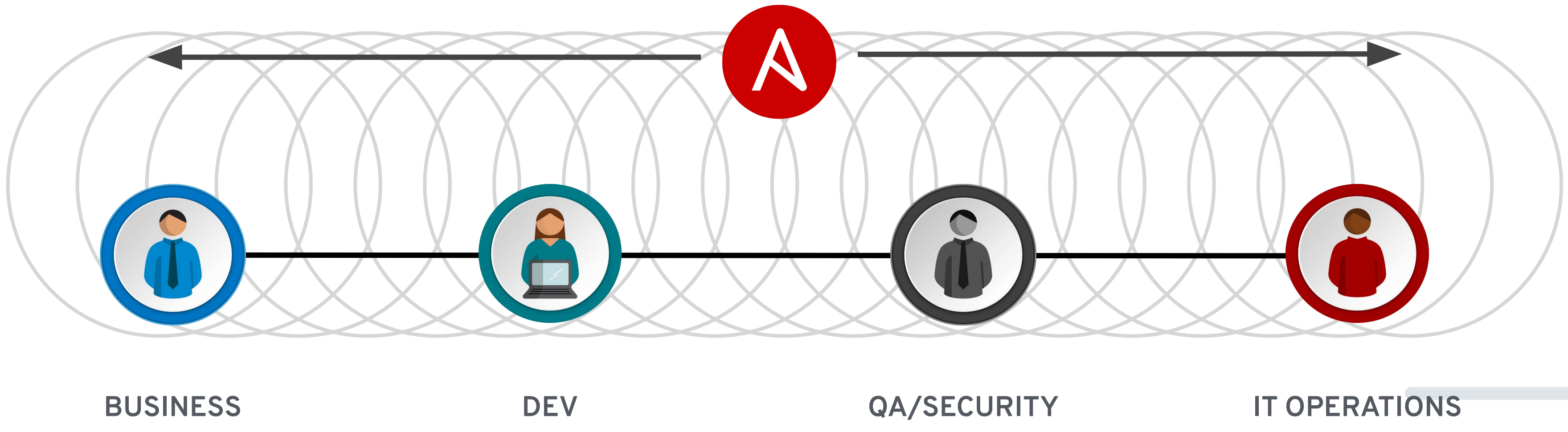
WHO AM I?

Ansible Ninja

WHO AM I?

```
cat baby.yml
---
- name: baby
  hosts: parental_units
  roles:
    - eat
    - sleep
    - poop
    - love
```

ANSIBLE IS THE UNIVERSAL LANGUAGE



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.

ANSIBLE AUTOMATES TECHNOLOGIES YOU USE

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

ANSIBLE TOWER

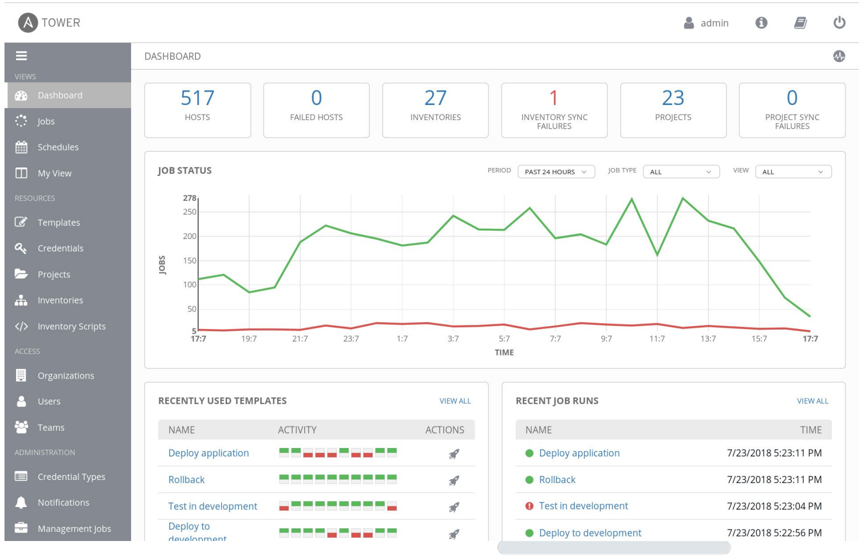
Ansible Tower is an **enterprise framework** for controlling, securing and managing your Ansible
automation – with a **UI and RESTful API.**

- **Role-based access control**

- **Deploy** entire applications with **push-button deployment** access

- All automations are **centrally logged**

- Works with Linux nodes, networking devices - and of course Windows nodes



517	0	27	1	23	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

RECENTLY USED TEMPLATES

NAME	ACTIVITY	ACTIONS
Deploy application		
Rollback		
Test in development		
Deploy to development		

RECENT JOB RUNS

NAME	TIME
● Deploy application	7/23/2018 5:23:11 PM
● Rollback	7/23/2018 5:23:11 PM
● Test in development	7/23/2018 5:23:04 PM
● Deploy to development	7/23/2018 5:22:56 PM

ANSIBLE WINDOWS AUTOMATION

Use Ansible to deploy and manage Windows systems and applications.

70+

Windows Modules

PLAYBOOK EXAMPLE: WINDOWS

```
- hosts: new_servers

  tasks:
- name: ensure IIS is running
  win_service:
    name: W3Svc
    state: running

  - name: add a domain user
    win_domain_user:
      name: somebody
      upn: somebody@mydomain.local
      groups:
        - Domain Admins
```

SO HOW DOES IT WORK?

NOT SSH

- WinRM (HTTP-based remote shell protocol)
- Non-interactive logon
- Different connection plugin
- Requires `pywinrm`

STILL NOT SSH

- PSRP support since Ansible 2.7
- Faster, better
- File transfer
- Requires `pypsrp`

… Microsoft OpenSSH?

POWERSHELL

- Unlike Python, "just there" on modern Windows
- We can use .NET
- Powershell 3+, Windows 7/Server 2008 RC2+

INVENTORY

- Windows has its own connection type
- Variable in inventory must be set
- Similar to other target platforms

INVENTORY EXAMPLE: WINDOWS

```
[windows]
mssqlserver.example.com
iisserver.example.com

[windows:vars]
ansible_connection=winrm
```

OR

```
ansible_connection=psrp
```

WHAT CAN WE DO NEXT WEEK?

MONDAY:

COMMANDS & SCRIPTS

WINDOWS COMMAND

- Simply executes a command
- Not run through shell → no shell variables, no shell specific commands
- Quite secure
- No real idempotency

WINDOWS COMMAND

```
- name: run a cmd command
  win_command: cmd.exe /c mkdir C:\temp

- name: run a vbs script
  win_command: cscript.exe script.vbs

- name: run from specific folder, skip when condition already met
  win_command: wbadmin -backupTarget:C:\backup\
  args:
    chdir: C:\somedir\
    creates: C:\backup\
```

WINDOWS SHELL

- Executes within a PowerShell
- Use PowerShell commands, variables, etc.
- Even multi-line scripts possible
- Less secure!
- No real idempotency

WINDOWS SHELL

```
- name: run command through the shell
  win_shell: Write-Host Hello world

- name: run multi-lined shell commands
  win_shell: |
    $value = Test-Path -Path C:\temp
    if ($value) {
        Remove-Item -Path C:\temp -Force
    }
    New-Item -Path C:\temp -ItemType Directory
```

SCRIPT

- Works on Linux and Windows
- Transfers and executes a script
- Local copy can still be templated!
- Only use in cases where the other modules don't work
- No real idempotency

SCRIPT

```
- name: run a script
  script: /tmp/myscript.bat
```

TUESDAY:

SOFTWARE MANAGEMENT

APPLICATION INSTALLATION

WAYS TO INSTALL SOFTWARE	
win_package	The default module to install MSI or EXE
win_chocolatey	If possible, use Chocolatey! A package management framework for Windows - like the app stores on mobile phones, homebrew or the repositories on Linux distributions. Community driven.
win_feature	Installs or uninstalls Windows Roles or Features on Windows Server using the Add/Remove-WindowsFeature Cmdlets on Windows 2008 R2 and Install/Uninstall-WindowsFeature Cmdlets on Windows 2012.
win_update	Manage updates: install KBs, install all updates from a certain category and blacklist what does not fit your current setup.
win_hotfix	Install or remove windows hotfixes.

APPLICATION INSTALLATION WITH WIN_PACKAGE

```
- name: Install Visual C thingy
  win_package:
    path: http://download.microsoft.com/.../vcredist_x64.exe
    product_id: '{CF2BEA3C-26EA-32F8-AA9B-331F7E34BA97}'
    arguments:
    - /install
    - /passive
    - /norestart
```

APPLICATION INSTALLATION WITH WIN_CHOCOLATEY

```
- name: Install multiple packages
  win_chocolatey:
    name:
    - procexp
    - putty
    - windirstat
    state: present
```

WINDOWS FEATURE

```
- name: Install IIS
  win_feature:
    name: Web-Server
    state: present

- name: Install IIS with sub features and management tools
  win_feature:
    name: Web-Server
    state: present
    include_sub_features: yes
    include_management_tools: yes
```

WINDOWS UPDATES

- Basic, synchronous updates - `win_updates`
- Uses configured source (Windows Update/WSUS)
- (New in 2.5): transparent SYSTEM + auto reboot

WINDOWS UPDATES

```
- name: install critical updates except blacklisted
  win_updates:
   category_names: CriticalUpdates
   reboot: yes # <--- new in 2.5!
   blacklist: # <--- new in 2.5!
   - KB4056892
```

REBOOTS

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

REBOOTS

```
# Apply updates and reboot if necessary
- win_updates:
  register: update_result
- win_reboot:
  when: update_result.reboot_required

# Reboot a slow machine that might have lots of updates to apply
- win_reboot:
    shutdown_timeout: 3600
    reboot_timeout: 3600
```

WEDNESDAY:

CONFIGURATION MANAGEMENT & SERVICES

REGISTRY

- Manage individual key/value (win_regedit)
- Manage idempotent bulk import (win_regmerge)

REGISTRY

```
- name: ensure registry value
  win_regedit:
    path: HKLM\Software\Microsoft\Windows
    name: SomeValueName
    value: 0x12345

- name: merge registry data
  win_regmerge:
    path: ComplexRegData.reg
```

ACLS

- More granular than Linux permissions
- SDDL?!
- More like SELinux ACLs

ACLS

```
- name: ensure owner recursively
  win_owner:
    path: C:\Program Files\SomeApp
    user: Administrator
    recurse: true


- name: ensure complex ACLs
  win_acl:
    path: C:\Temp
    user: Users
    rights: ReadAndExecute,Write,Delete
    inherit: ContainerInherit,ObjectInherit
```

SERVICES

- `win_service` looks/acts like Linux service module
- Provides fine control over complex service behavior config in Windows SCM (who/what/when/how)

SERVICES

```
- name: ensure IIS is running
  win_service:
    name: W3Svc
    state: running

- name: ensure firewall service is stopped/disabled
  win_service:
    name: MpsSvc
    state: stopped
    start_mode: disabled
```

THURSDAY:

DOMAINS
&
CREDENTIALS

DOMAINS

- Enterprise identity management
- Makes auth complex
- Ansible can do "throwaway" domains easily
- Promote/depromote Domain Controllers
- Joining/leaving domain is simple
- Manage basic domain objects

DOMAINS

```
- name: create a domain
  win_domain:
    dns_domain_name: mydomain.local
    safe_mode_password: ItsASecret

- name: add a domain user
  win_domain_user:
    name: somebody
    upn: somebody@mydomain.local
    groups:
      - Domain Admins
```

BECOME

- Run with full privileges that are available to remote user
- Uses `runas` user
- Ansible >= 2.5, else UAC and `SeTcbPrivilege`
- `become_user`: local or domain user account, local service accounts like System or NetworkService

BECOME

```
- win_whoami:


- win_whoami:
  become: yes


- win_whoami:
  become: yes
  become_user: System
```

FRIDAY:

DSC, ANYONE?

WHAT ABOUT DSC?

CONFIGURATIONS

- Declarative PowerShell scripts
- Define and configure instances of resources
- DSC will simply “make it so”
- Idempotent

RESOURCES

- "Make it so" part of DSC
- Contain the code
- Files, Windows processes, VM running in Azure, etc.

WHY USE ANSIBLE & DSC TOGETHER?

- Embed DSC in a broader automation approach
- Cover more than just Windows
- Tasks which are not idempotent by design

WHY USE ANSIBLE & DSC TOGETHER?

- Use Tower features via Ansible to govern DSC execution
- Manage DSC resources
- Free form module

ANSIBLE & DSC - WE HAVE MODULES FOR THAT!

```
- name: install xDNSServer DSC module on target
  win_psmodule:
    name: xDnsServer

- name: create DNS zone
  win_dsc:
    resource_name: xDnsServerPrimaryZone
    name: createdbyansible.com
```

WRAP-UP

WRAP-UP

Windows is a first class citizen within the Ansible ecosystem!

DO NEXT

GET STARTED

- ✓ Try Tower for free
  ansible.com/tower-trial
- ✓ Three steps to start right off
  ansible.com/get-started
- ✓ Want to learn more?
  ansible.com/resources

FOCUS ON WINDOWS

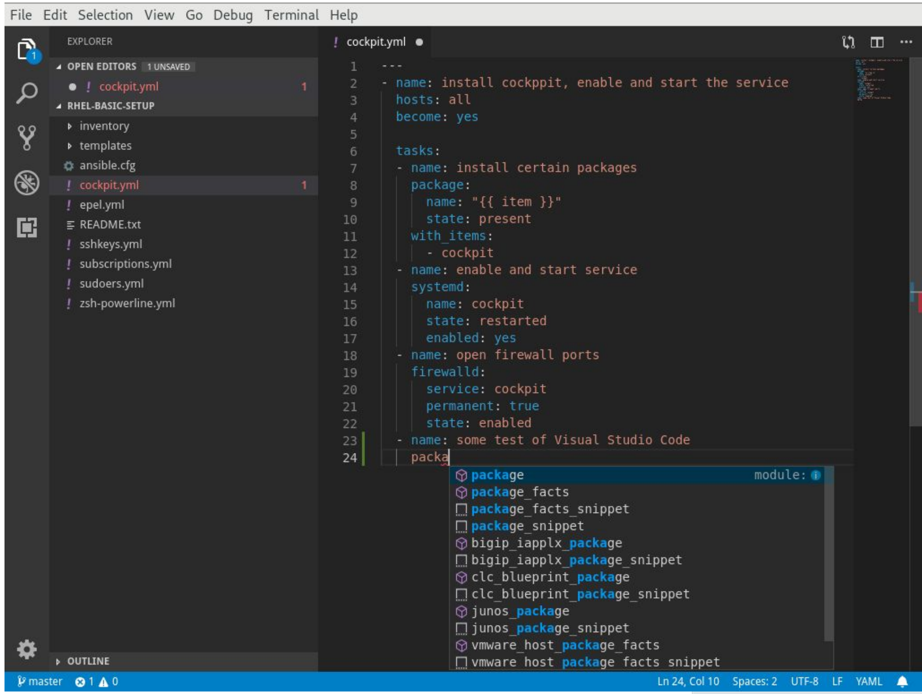
- ✓ Connect to your hosts
  ansible.com/blog/connecting-to-a-windows-host
- ✓ Check out roles for Windows platform
  on galaxy.ansible.com
- ✓ Start with Ansible and Azure
  docs.microsoft.com/en-us/azure/ansible/

HOW TO CODE?

USE VISUAL STUDIO CODE

Lightweight but **powerful** open source editor. And a rich Ansible extension is available - provided by Microsoft.

- Code completion
- Syntax highlighting
- Run playbooks



```
---
- name: install cockppit, enable and start the service
  hosts: all
  become: yes

  tasks:
  - name: install certain packages
    package:
      name: "{{ item }}"
      state: present
    with_items:
      - cockpit
  - name: enable and start service
    systemd:
      name: cockpit
      state: restarted
      enabled: yes
  - name: open firewall ports
    firewalld:
      service: cockpit
      permanent: true
      state: enabled
  - name: some test of Visual Studio Code
    packa
```

```
package                                   module:
package_facts
package_facts_snippet
package_snippet
bigip_iapplx_package
bigip_iapplx_package_snippet
clc_blueprint_package
clc_blueprint_package_snippet
junos_package
junos_package_snippet
vmware_host_package_facts
vmware_host_package_facts_snippet
```

RED HAT
SUMMIT

THANK YOU

linkedin.com/company/Red-Hat

facebook.com/RedHatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat