Red Hat
Enterprise Linux

Ansible Automation Platform 2 Knowledge Sharing

2023/04/27

Hunter Feng
Senior Technical Account Manager

Red Hat

1. 程式換版: 備份原檔案、更換新檔案、更換後檢 查檔案屬性&checksum
2. 新機安裝
3. 主機備份還原
4. 存取權限審 查: 抓Windows, RHEL, SUSE主機使用者及權限
5. 網路備援切換 : 異地備援切換時, 切換 router線路、切換Palo Alto防火牆設定
6. 網路設定備份

簡報大綱

- Ansible Automation Platform 2 架構說明
- Ansible Automation Platform 2架構範例
- Ansible Automation Platform 2 Demo
- Ansible Automation Platform 2 Use Case
- 補充 : Ansible for Windows Automation

Red Hat
Enterprise Linux

Ansible Automation Platform 2
架構說明

Hunter Feng
Senior Technical Account Manager

Red Hat

Red Hat Named a Leader in Infrastructure Automation by Industry Research Firm

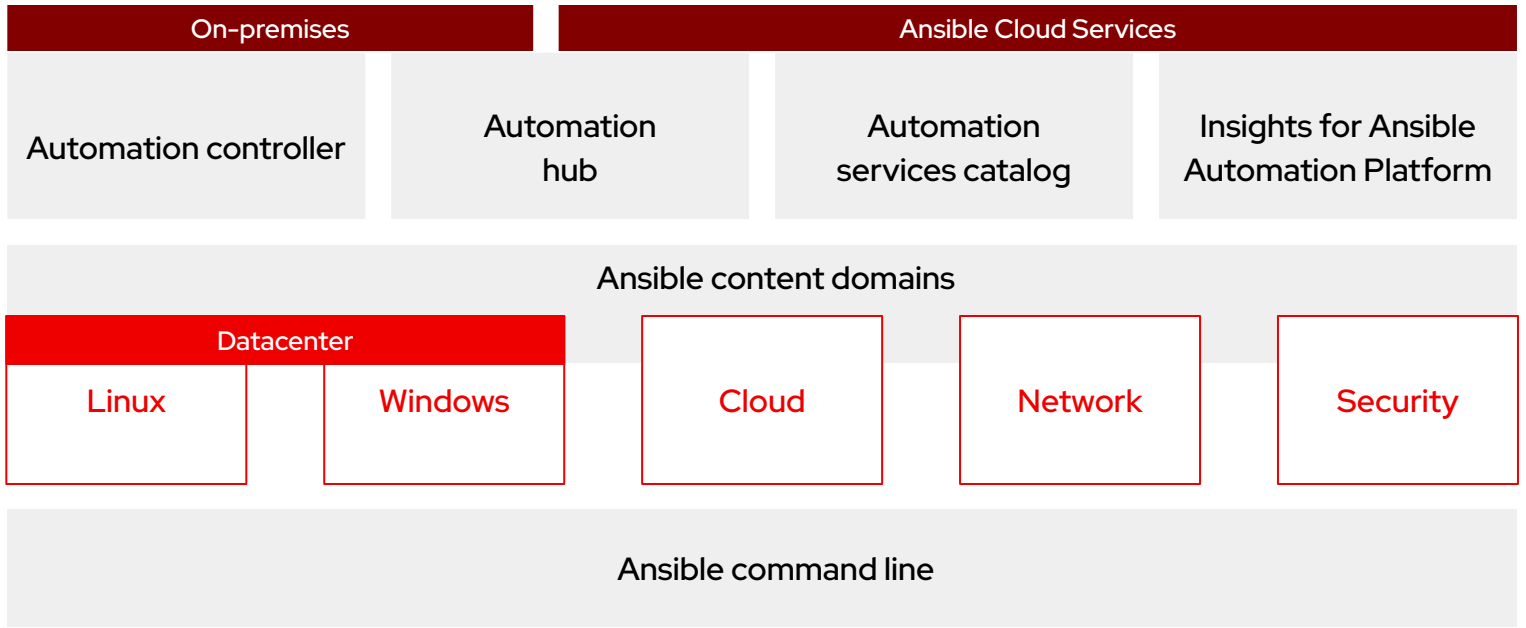
https://www.forrester.com/policies/forrester-wave-methodology/

Red Hat

What makes a platform?

Red Hat
Ansible Automation Platform

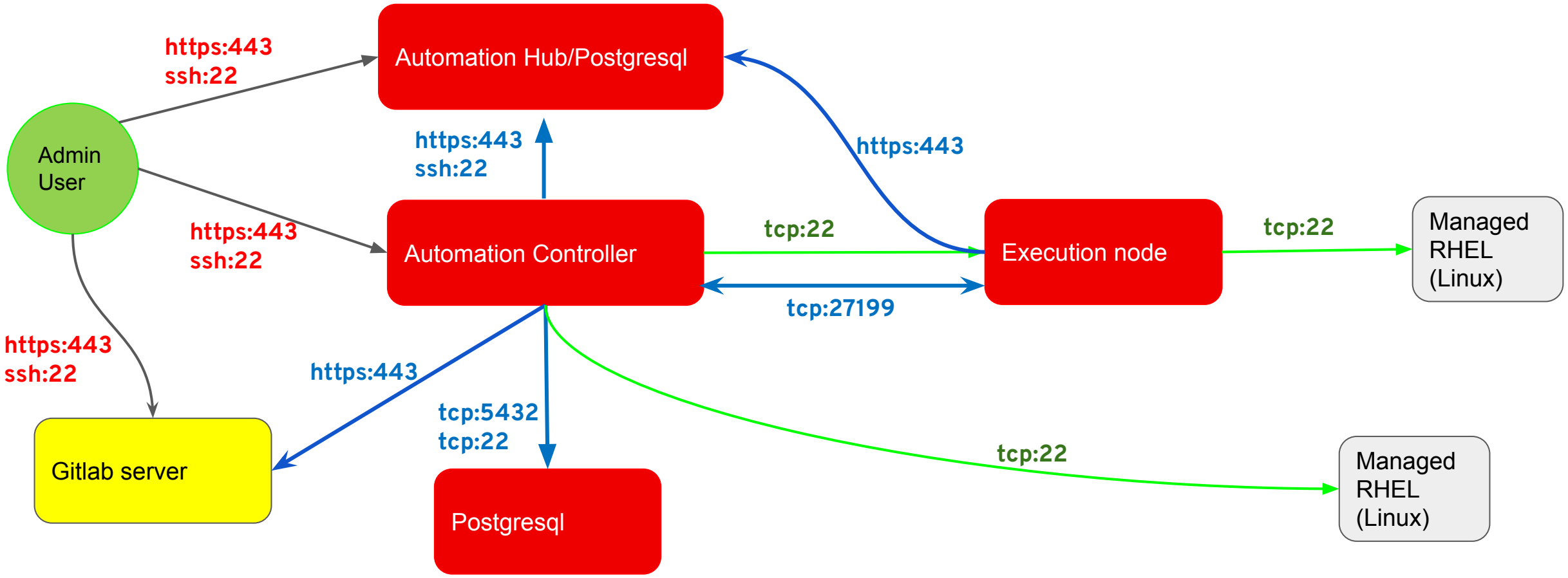
- Content creators
- Operators
- Domain experts
- Users



Fueled by an open source community

Red Hat

AAP2 Network Topology

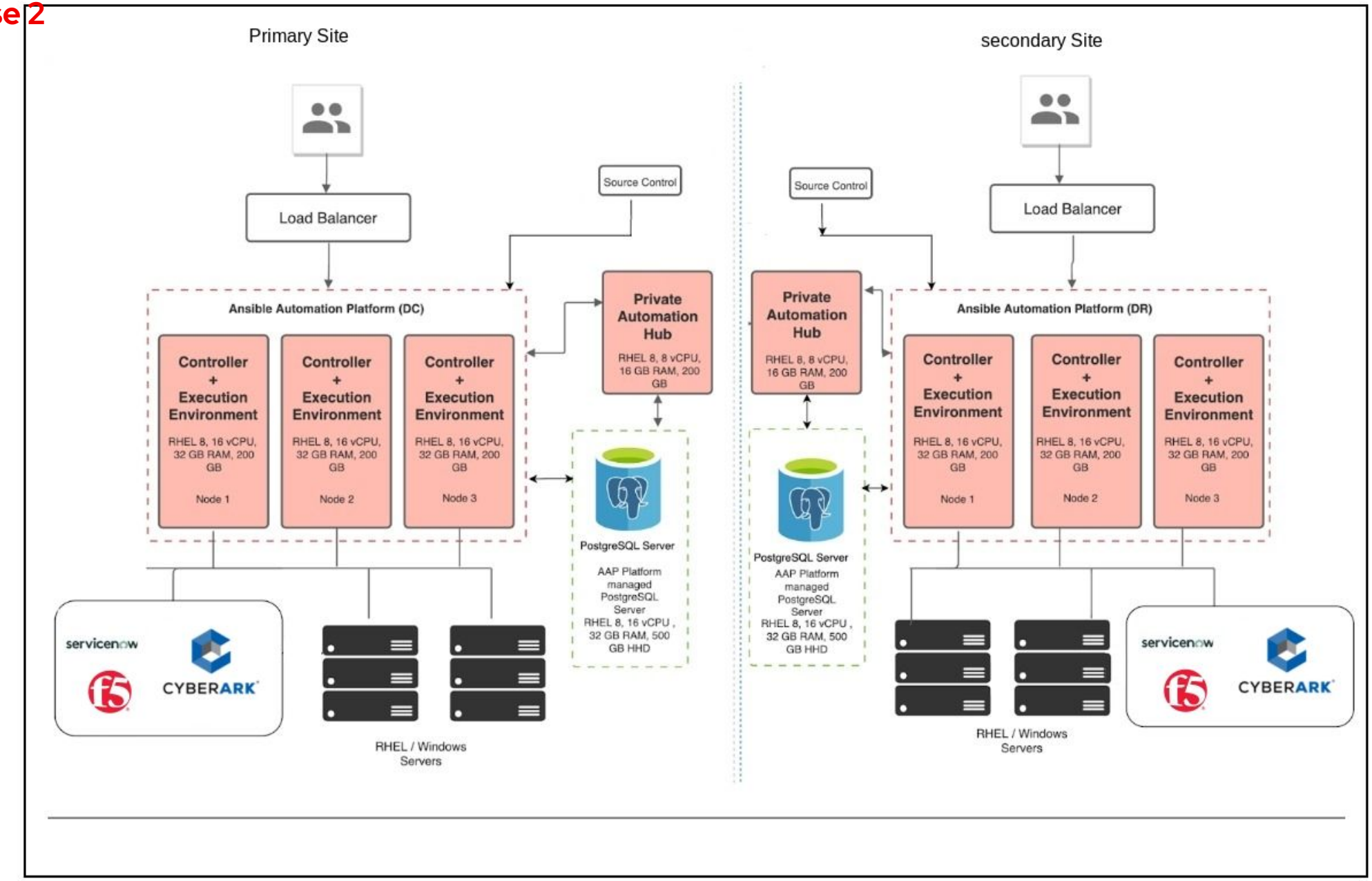


Red Hat
Enterprise Linux

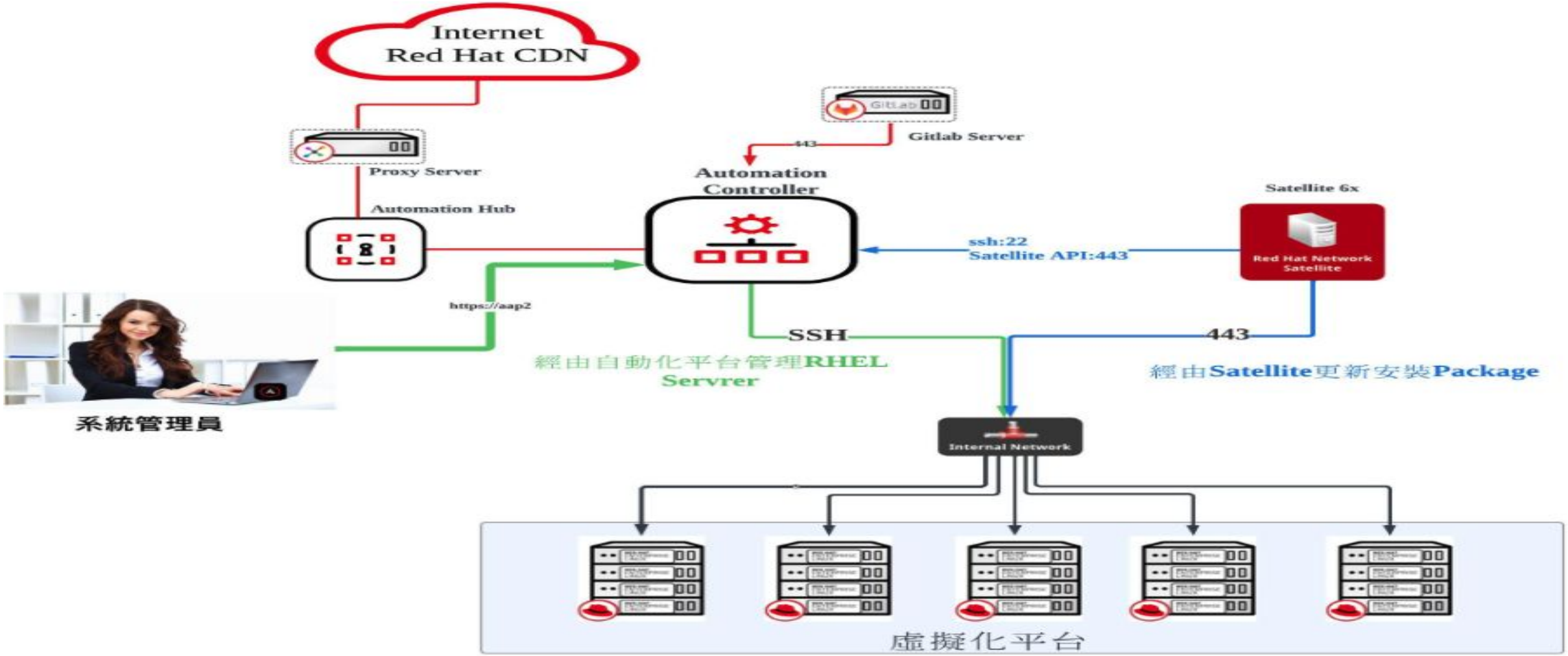
Red Hat AAP2 架構範例

Red Hat

Use case 2

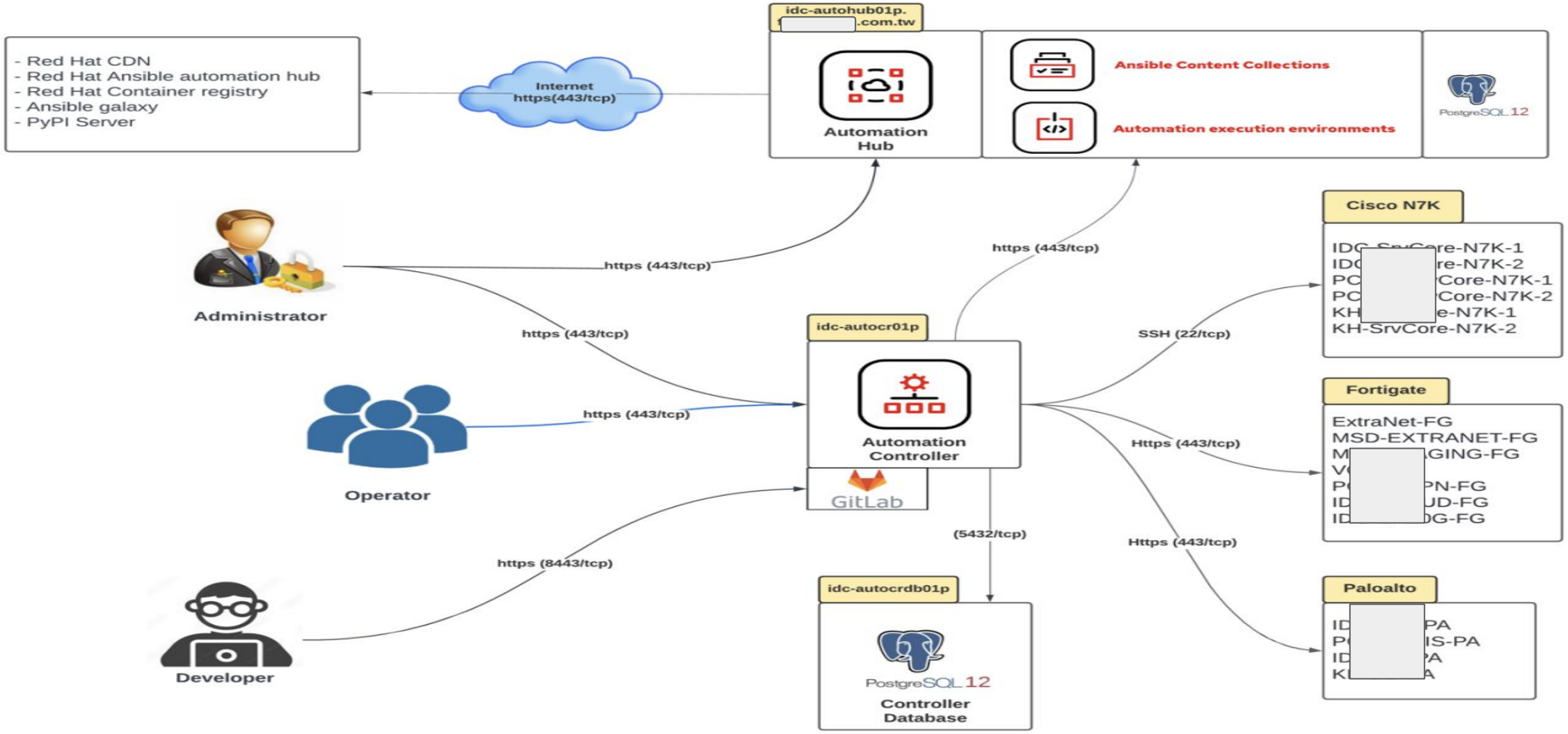

Primary Site
secondary Site
Source Control
Load Balancer
Private Automation Hub
RHEL 8, 8 vCPU, 16 GB RAM, 200 GB
Ansible Automation Platform (DC)
Ansible Automation Platform (DR)
Controller + Execution Environment
RHEL 8, 16 vCPU, 32 GB RAM, 200 GB
Node 1
Node 2
Node 3
PostgreSQL Server
AAP Platform managed PostgreSQL Server
RHEL 8, 16 vCPU , 32 GB RAM, 500 GB HHD
servicenow
CYBERARK
RHEL / Windows Servers

自動化平台實踐



AAP 架範例 2

三台主機 : automation hub(with hub DB) x 1 + controller x 1 + controller db x 1



Red Hat

Interactive labs for Red Hat Ansible Automation Platform

- https://www.redhat.com/en/interactive-labs/ansible#manage
- 申請測試版本
  https://www.redhat.com/en/technologies/management/ansible/trial

Create

Create and share automation across your organization—from development and operations to security and network teams.

Get started with ansible-navigator

Install ansible-navigator and take a closer look at the command line.

30 mins

Network automation basics: First playbook

Learn the fundamentals of Red Hat Ansible Automation Platform for network automation using ansible-navigator.

20 mins

Network automation: Backup and restore

Learn how to perform network configurations and backups using Red Hat Ansible Automation Platform.

20 mins

Network automation basics: Resource modules

Learn Red Hat Ansible Automation Platform playbook basics for network automation.

40 mins

Network automation basics: Facts

Learn about retrieving facts from a Cisco IOS-XE device.

30 mins

Get started with ansible-builder

Install ansible-builder and learn how to create custom execution environments.

50 mins

Manage

Manage network and IT practices efficiently—from rapid development and deployment, to simplified operations and analytics, to consistent end-to-end user experiences.

Use Red Hat Ansible Automation Platform on Microsoft Azure

Deploy Red Hat Ansible Automation Platform on Microsoft Azure and perform automation tasks in your Azure environment.

45 mins

Network automation: Infrastructure awareness

Learn how to use Red Hat Ansible Automation Platform to retrieve facts from network infrastructure and create dynamic documentation.

15 mins

Network automation basics: Surveys

Learn how to create an automation controller survey to configure a Cisco IOS network device.

20 mins

Sign Ansible Content Collections with private automation hub

Learn how to sign Ansible Content Collections using a private automation hub and install collections with ansible-galaxy CLI.

30 mins

Get started with automation controller

Explore the automation controller interface and complete some basic tasks.

25 mins

DevOps & CI/CD with automation controller

Integrate a CI/CD pipeline into automation controller to see how Red Hat Ansible Automation Platform supports DevOps practices.

60 mins

Red Hat

Red Hat
Enterprise Linux

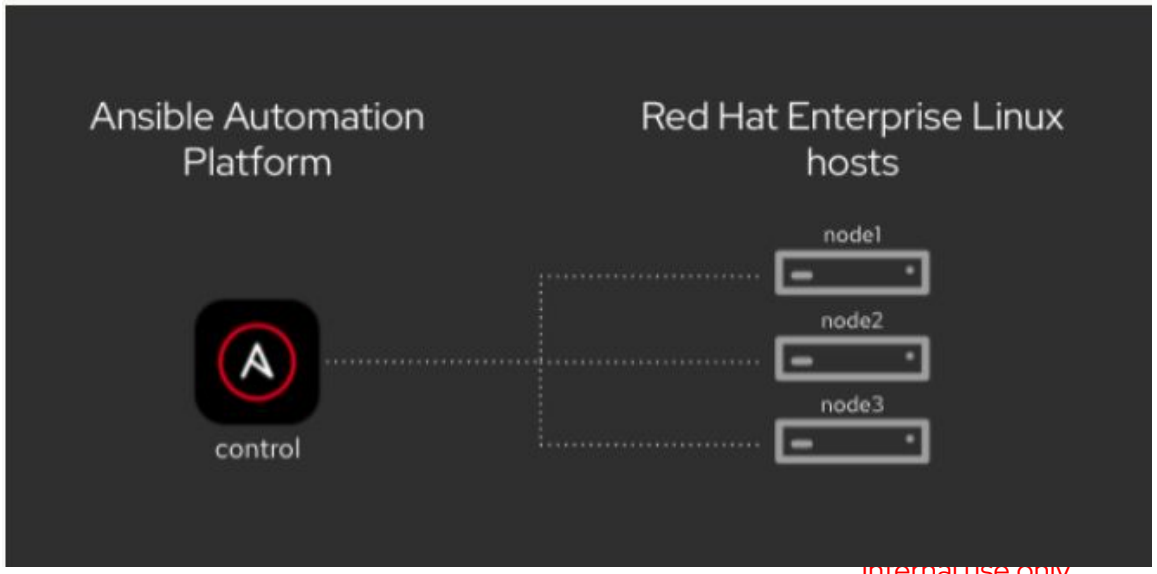
Let's Demo

Hunter Feng
Senior Technical Account Manager

Red Hat

Lab 1 : Automation Deploy Application

- Exercise 2.1 - Introduction to automation controller
- Exercise 2.2 - Inventories, credentials and ad hoc commands
- Exercise 2.3 - Projects & job templates
- Exercise 2.4 - Surveys
- Exercise 2.5 - Role based access control
- Exercise 2.6 - Workflows



Lab 2 : Automation Deploy Application

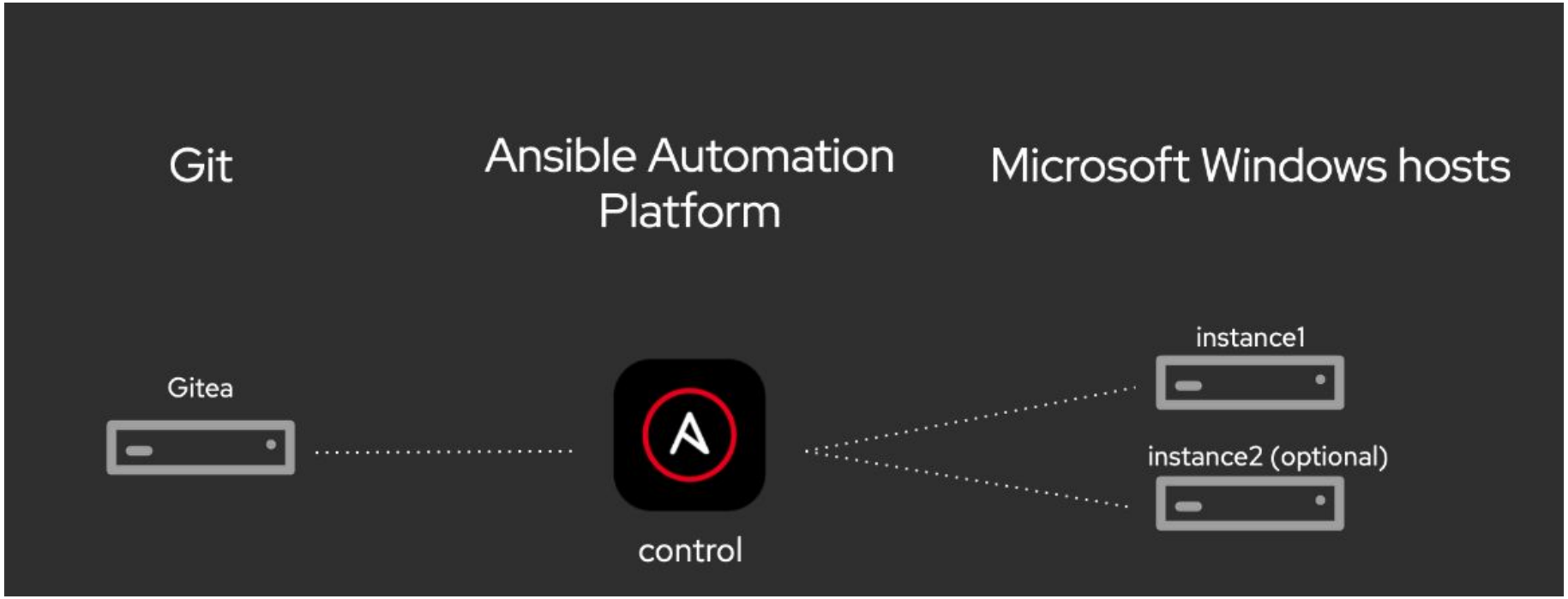
此範例使用Ansible **copy/fecth** module，另可使用**synchronize** module，同rsync功能

- 備份流程，備份/opt/myapp下的資料至Ansible Controller

- 還原流程，將備份資料還原至Node下myapp1資料夾下

Lab 3 : Ansible for Windows Automation

- Exercise 1 - Intro and configuration of Automation Controller
- Exercise 2 - Ad-hoc commands
- Exercise 3 - Intro to playbooks
- Exercise 4 - Automation Controller projects



Red Hat

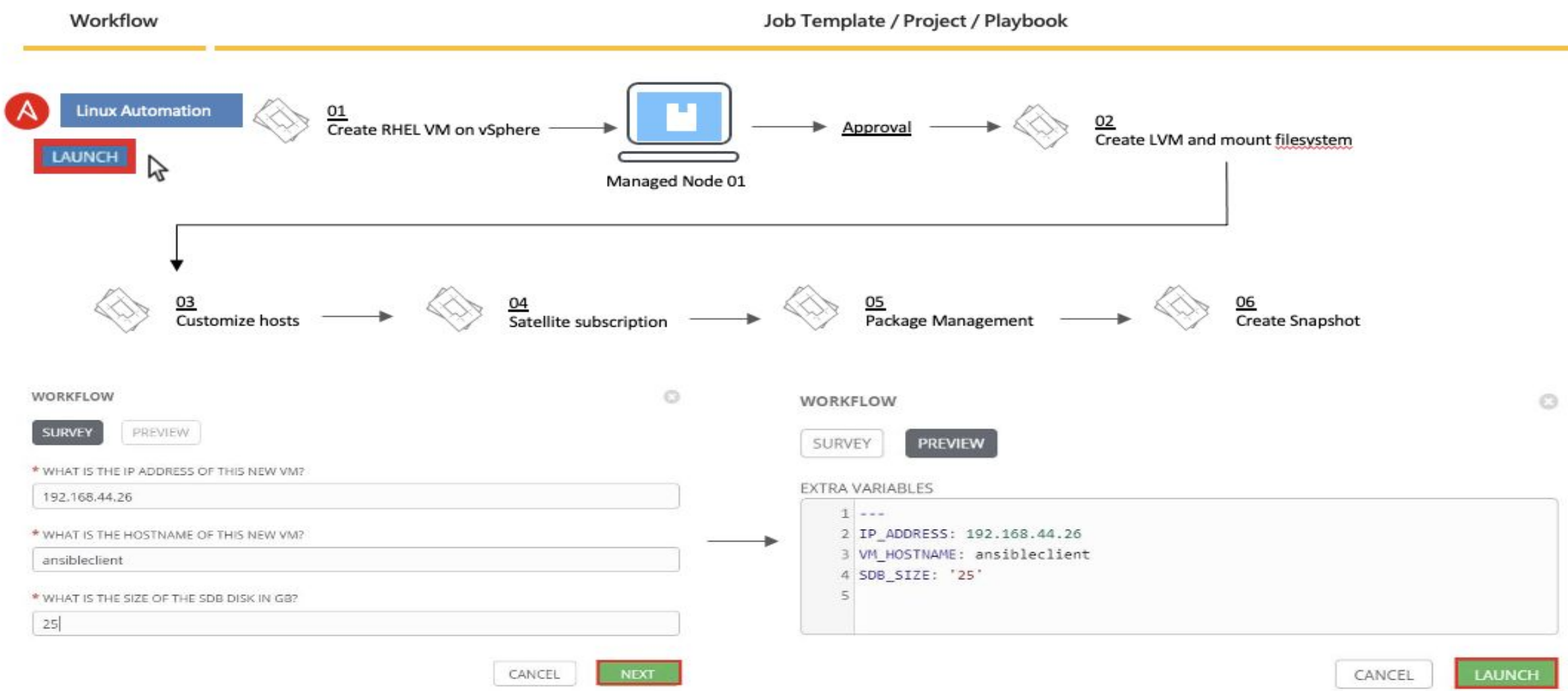
Red Hat
Enterprise Linux

Ansible Automation Platform 2 Use Case

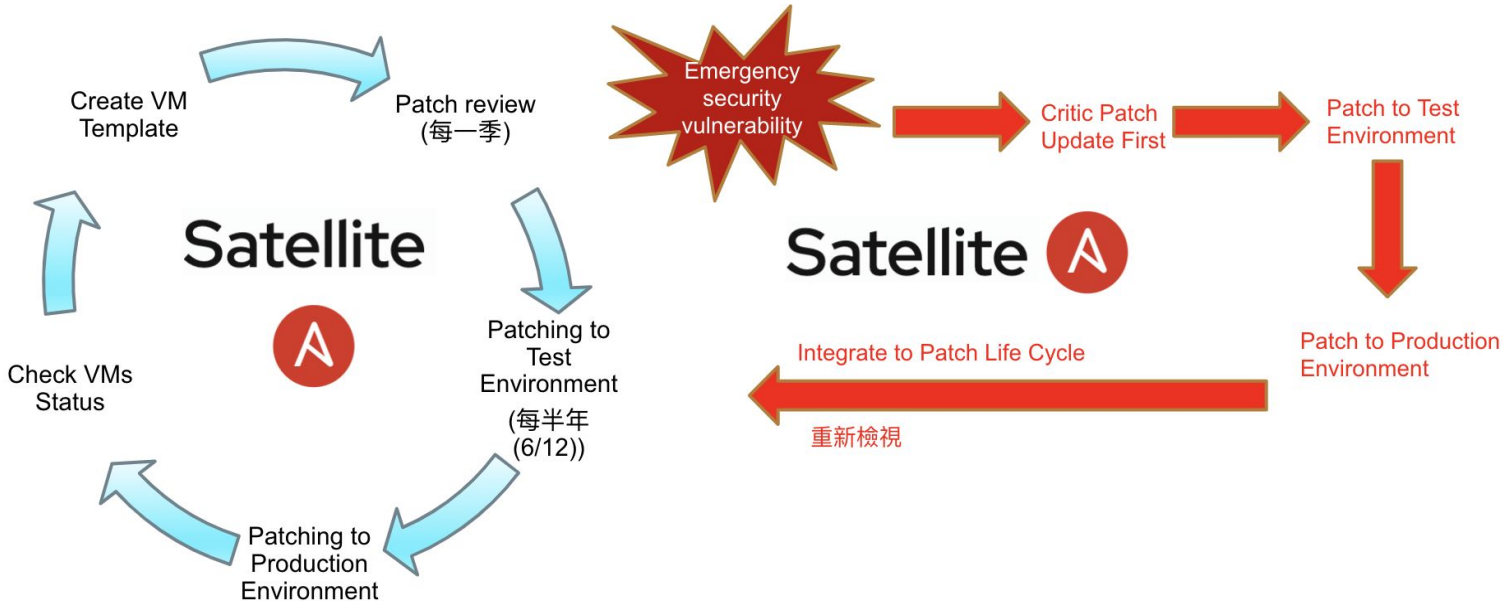
Hunter Feng
Senior Technical Account Manager

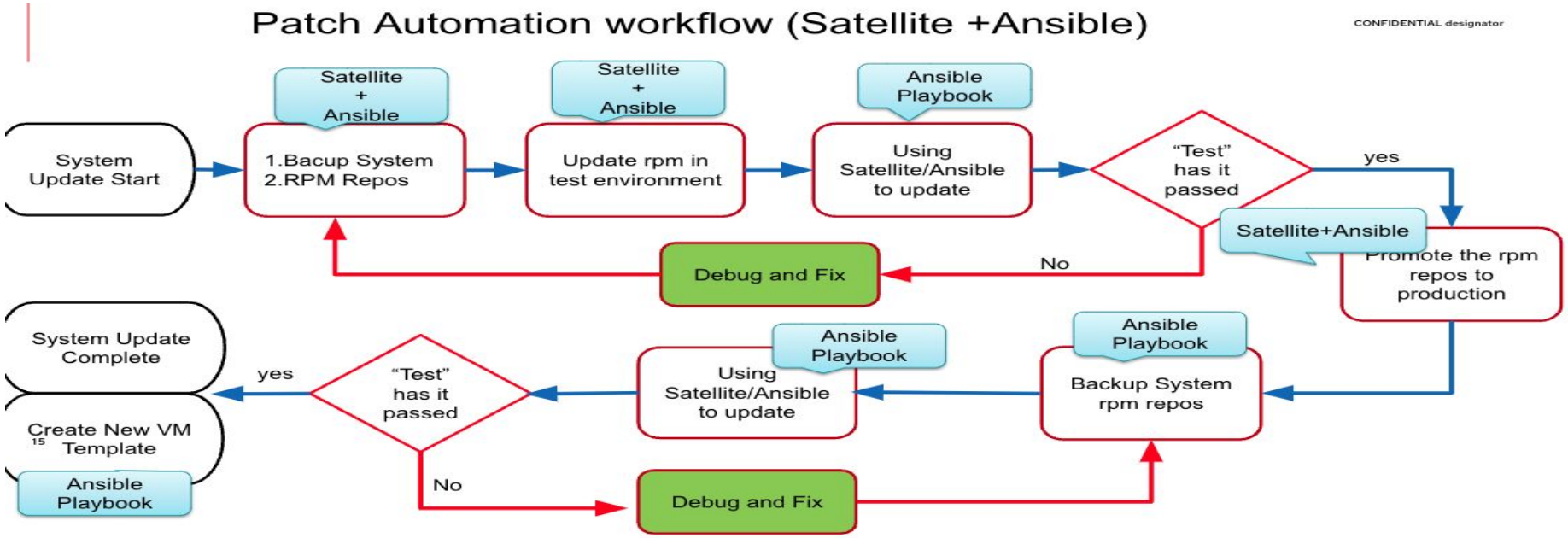
Red Hat

系統標準化，協助infa團隊管理，實現Infrastructure as Code (IaC).



Patch 流程自動化，快速修補漏洞，加強資安防護





系統檢查報表，資安檢查報表，協助IT團隊監控系統

TWGCB check report - localhost.localdomain

```
Server/System Configuration
Platform        : RedHat
Host Name       : localhost.localdomain
IP Address      : 192.168.8.101
Subnet Mask     : 255.255.254.0
OS Version      : release 8.5 ( Ootpa )
Kernel Version  : 4.18.0-348.23.1.el8_5.x86_64
```

Check started at: 2022-08-14 23:00:03 Sunday

TWGCB-ID	檢查結果	類別	原則設定名稱	說明	備註說明
TWGCB-01-008-0032	PASS +detail	系統設定與維護	GPG簽章驗證	+detail	check if gpgcheck=1
TWGCB-01-008-0033	PASS +detail	系統設定與維護	sudo套件	+detail	check if sudo RPM exists
TWGCB-01-008-0034	FAILED +detail	系統設定與維護	設定sudo指令使用pty	+detail	check use_pty in sudoers
TWGCB-01-008-0035	FAILED +detail	系統設定與維護	sudo自定義日誌檔案	+detail	check sudo logfile
TWGCB-01-008-0036	FAILED +detail	系統設定與維護	AIDE套件	+detail	check aide RPM exists
TWGCB-01-008-0037	FAILED +detail	系統設定與維護	定期檢查檔案系統完整性	+detail	check aide check daily job
TWGCB-01-008-0038	PASS +detail	系統設定與維護	開機載入程式設定檔之所有權	+detail	check boot files owner
TWGCB-01-008-0039	FAILED +detail	系統設定與維護	開機載入程式設定檔之權限	+detail	check boot files permission
TWGCB-01-008-0040	FAILED +detail	系統設定與維護	開機載入程式之密碼	+detail	check boot files password
TWGCB-01-008-0041	PASS +detail	系統設定與維護	單一使用者模式身分驗證	+detail	check single user mode
TWGCB-01-008-0042	FAILED +detail	系統設定與維護	核心傾印功能	+detail	check core dump feature
TWGCB-01-008-0043	PASS +detail	系統設定與維護	記憶體位址空間配置隨機載入	+detail	check kernel.randomize_va_space
TWGCB-01-008-0044	FAILED +detail	系統設定與維護	設定全系統加密原則	+detail	check crypto policy
TWGCB-01-008-0045	PASS +detail	系統設定與維護	/etc/passwd檔案所有權	+detail	check /etc/passwd file owner
TWGCB-01-008-0046	PASS +detail	系統設定與維護	/etc/passwd檔案權限	+detail	check /etc/passwd file permission
TWGCB-01-008-0047	PASS +detail	系統設定與維護	/etc/shadow檔案所有權	+detail	check /etc/shadow file owner
TWGCB-01-008-0048	PASS +detail	系統設定與維護	/etc/shadow檔案權限	+detail	check /etc/shadow file permission
TWGCB-01-008-0049	PASS +detail	系統設定與維護	/etc/group檔案所有權	+detail	check /etc/group file owner
TWGCB-01-008-0050	PASS +detail	系統設定與維護	/etc/group檔案權限	+detail	check /etc/group file permission
TWGCB-01-008-0051	PASS +detail	系統設定與維護	/etc/gshadow檔案所有權	+detail	check /etc/gshadow file owner
TWGCB-01-008-0052	PASS +detail	系統設定與維護	/etc/gshadow檔案權限	+detail	check /etc/gshadow file permission
TWGCB-01-008-0053	PASS +detail	系統設定與維護	/etc/passwd檔案所有權	+detail	check /etc/passwd- file owner
TWGCB-01-008-0054	FAILED +detail	系統設定與維護	/etc/passwd-檔案權限	+detail	check /etc/passwd- file permission

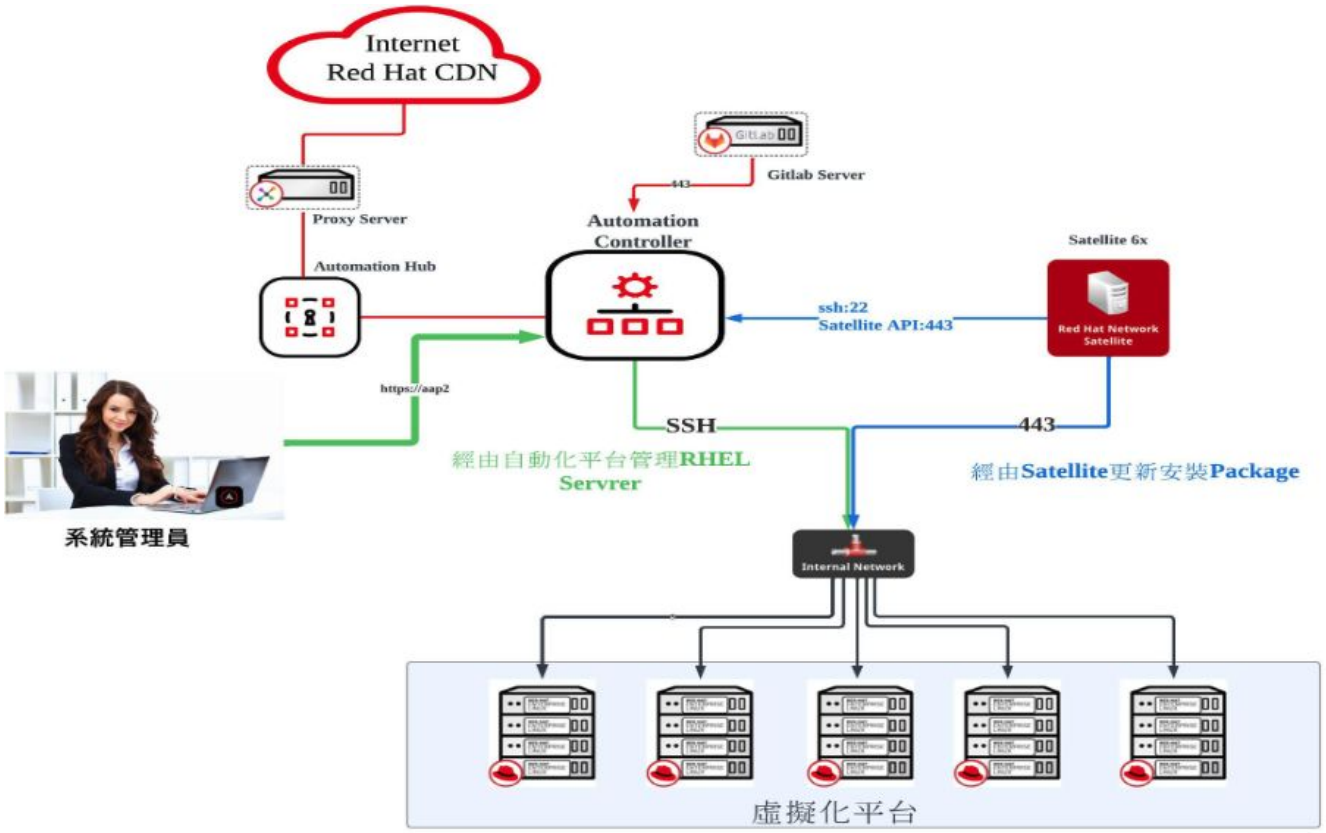
系統檢查報表，資安檢查報表，協助IT團隊監控系統

主機系統		主機網路		主機容量	
系統平台	RedHat	網路介面	ens32	記憶體大小	3731 MB
主機名稱	kyjump.cloudcube.local	網路位址	192.168.129.120	SWAP大小	4091 MB
系統版本	release 8.7 (Ootpa)	網路遮罩	255.255.255.0	硬碟大小	dm-14.00 GB dm-0195.00 GB sda200.00 GB
核心版本	4.18.0-425.3.1.el8.x86_64	MAC位址	00:50:56:85:1f:0f		

執行時間 2023-02-24 09:23:56 Friday ~ 2023-02-24 09:55:46 Friday

規則代號	規則項目	規則說明	檢查結果
規則類別 A. 磁碟與檔案系統			
TWGCB-01-008-0001	cramfs 檔案系統	▪ 這項原則設定決定是否支援 cramfs 檔案系統 ▪ cramfs (compressed ROM file system，壓縮唯讀閃存檔案系統)檔案系統是一開放式之 Linux 檔案系統，目的是更簡單更有效率 ▪ cramfs 檔案系統以 zlib 壓縮資料，不需載入到記憶體中，因此可節省許多記憶體空間，可直接使用 cramfs 映像檔案無須先解壓，使用於某些舊系統或對記憶體有限制之地方 ▪ 停止支援 cramfs 檔案系統，以降低系統被攻擊面	+more
TWGCB-01-008-0002	squashfs 檔案系統	▪ 這項原則設定決定是否支援 squashfs 檔案系統 ▪ squashfs 是一個即時解壓縮之檔案系統，專門為唯讀壓縮檔案系統之使用而設計，常見於各 Linux 發行版之 LiveCD ▪ 停止支援 squashfs 檔案系統，以降低系統被攻擊面	+more

自動化平台實踐



Challenge

- 升請主機過程過於費時
- 系統版本無法一致性
- 系統設定紊亂，管理不易
- 上Patch不易，無法升級
- 資安報表需求-TWGCB

Solution

- Red Hat Enterprise Linux
- Red Hat Satellite
- Red Hat Ansible Automation Platform
- Red Hat TAM Service

Why Red Hat

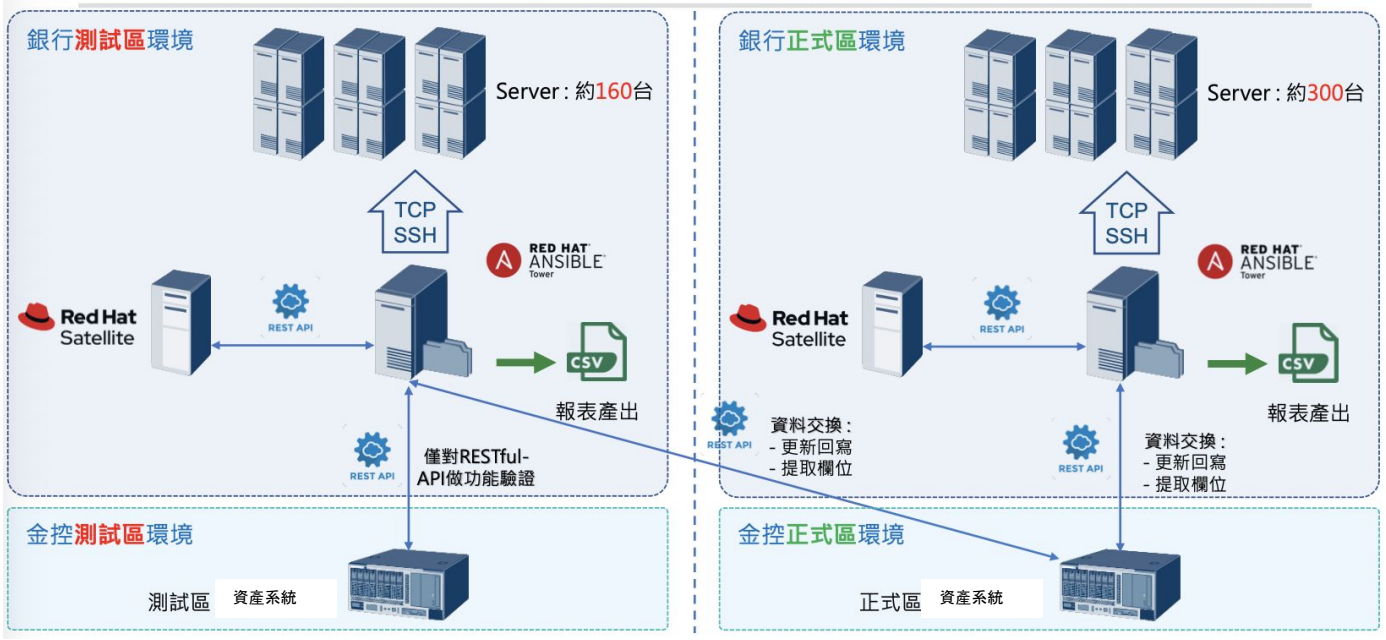
- Red Hat有強大的顧問團隊
- Red Hat 是自動化平台的領導者
- Red Hat 是最有經驗是Linux的專家

Results

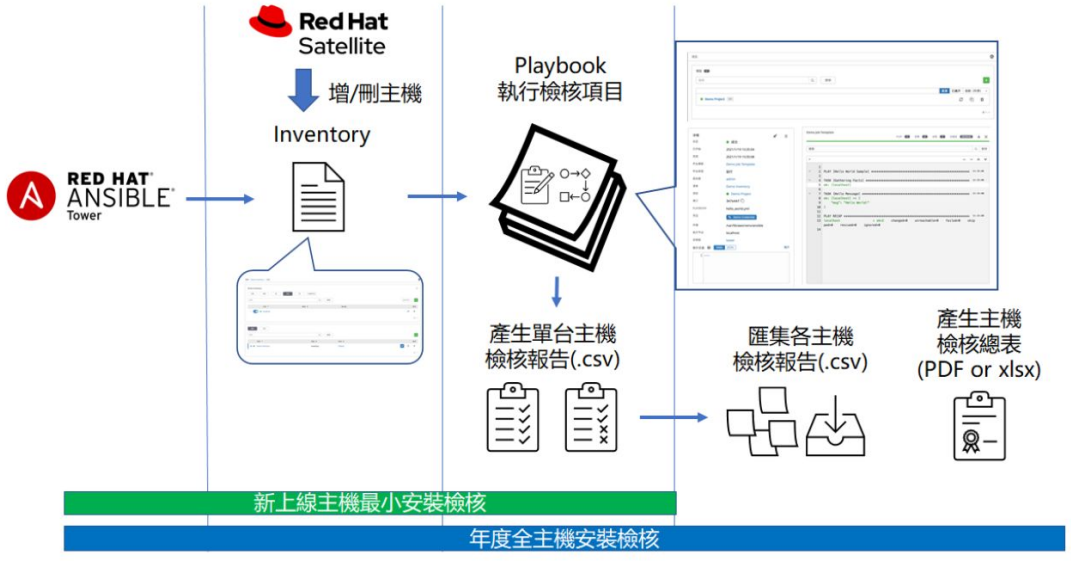
- 大幅縮短RHEL主機佈署時間，可同時佈署多台主機
- 系統標準化，協助infa團隊管理，實現**Infrastructure as Code (IaC).**
- 強化RHEL Security，簡化patch流程，執行定期更新
- 定期產出資安報表，合規檢 查

V0000000

某大型銀行合規自動化導入



Ansible 腳本執行流程圖 – 最小安裝檢核為例



解決方案

- Red Hat Ansible Tower
- Red Hat Satellite
- 客制化 Ansible 腳本撰寫
- Red Hat TAM Service

RedHat Ansible Tower 整合 Red Hat Satellite 與客戶自有的資產管理系統(ISMS)，以確保系統資 產資訊一致，並訂定日常維運腳本及自動化政策規則。

客戶原來情況

- 共有近 500 台 Linux
- 作業系統更新 Patch 耗時耗工，常要配合晚上或假日
- 因應資安稽核要求，除了定期進行合規檢 查作業外， 還需應對稽核臨時且緊急的安全性 查核。人工作業逐台清查耗時費力、且易於出錯及記錄。
- 資訊資產人工更新耗時並且正確性 /時效性有待確認

關鍵效益

- 資訊資產正確率提高，節省人工輸入作業時間

作業說明	人工作業所需時間	優化後時間
每年二次弱點修補時間	約800小時	約200小時
每年二次帳號Review時間	約200小時	約50小時
每年合規檢核作業時間	約60小時	約5小時

Red Hat
Ansible Automation
Platform

Ansible for Windows Automation

Improving speed, agility, and productivity
with open source solutions

v0000000

How to connect managed notes?

Not SSH

- WinRM (HTTP-based remote shell protocol)
  - Non-interactive logon
  - Different connection plugin
  - Requires `pywinrm` on control node
  - PSRP[1] support since Ansible 2.7
  - Faster, better
  - File transfer
  - Requires `pypsrp`
- Microsoft OpenSSH?

[1] Ansible uses the PowerShell Remoting Protocol (PSRP) on top of WinRM to execute PowerShell commands on a target. PSRP provides a faster direct connection to PowerShell.

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

Sample inventory file: winHosts

```
[win]
ad
192.168.0.215
[win:vars]
ansible_connection=winrm
# ansible_port=5986
ansible_user=Administrator
ansible_password=xxx
ansible_winrm_server_cert_validation=ignore
ansible_winrm_transport=credssp
ansible_become=false
```


Note: By default WinRM works only with Administrator. Refer to here and read 'Non-administrator accounts' section for more information.

More about WinRM

Supported 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

Complete discussion for these 5 options:
https://docs.ansible.com/ansible/latest/os_guide/windows_winrm.html

In this slide, we will use CredSSP for authentication.
ps. Below package may be necessary for control node (KB 3382521)
pip install pywinrm[credssp]

Prepare the Windows

Logon as Administrator, and run below in Windows PowerShell

C:\Users\Administrator\Documents> .\ConfigureRemotingForAnsible.ps1 -DisableBasicAuth and -EnableCredSSP

```
PS C:\Users\Administrator\Documents> .\ConfigureermotingForAnsible.ps1 -DisableBasicAuth -EnableCredSSP


cfg                 : http://schemas.microsoft.com/wbem/wsman/1/config/service/auth
lang                : en-US
Basic               : false
Kerberos            : true
Negotiate           : true
Certificate         : false
CredSSP             : true
CbtHardeningLevel   : Relaxed
```

First Attempt


```
$ ansible -m win_ping -i winHosts win
ad | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
$ ansible -m setup -i winHosts win
ad | SUCCESS => {
(omitted)
        "ansible_distribution": "Microsoft Windows Server 2016
Datacenter",
 (omitted)
```

WHAT CAN WE DO NEXT?

COMMANDS & SCRIPTS

Windows Command

- Simply executes a command
- Not run through shell → no shell variables, no shell specific commands
- Quite secure
- No real idempotency[1]

```
[pat@haumea-lan gitlab.dev] $ ansible -m win_command -i winHosts -a 'cmd.exe /c mkdir c:\temp\aaa'  win
ad | CHANGED | rc=0 >>

[pat@haumea-lan gitlab.dev] $ ansible -m win_command -i winHosts -a 'cmd.exe /c mkdir c:\temp\aaa'  win
ad | FAILED | rc=1 >>
A subdirectory or file c:\temp\aaa already exists.
non-zero return code
```

[1] idempotency: 當同樣的事做二次以上，第二次的派送不應發生變更。

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
  ansible.windows.win_shell: Write-Host Hello world

- name: run multi-lined shell commands
  ansible.windows.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
  ansible.builtin.script: /tmp/myscript.bat
```

/tmp/script.bat will be copied to managed node first and then execute it.

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++ Redistributable
  ansible.windows.win_package:
    path: http://download.microsoft.com/…/vcredist_x64.exe
    product_id: '{CF2BEA3C-26EA-32F8-AA9B-331F7E34BA97}'
    arguments:
    - /install
    - /passive
    - /norestart
```

Note: above software can be downloaded from here.

Application Installation with win_chocolatey


```
- name: Install multiple packages
  win_chocolatey:
    name:
    - git
    - notepadplusplus
    - windirstat
    state: present
```

Windows Feature


```
- name: Install IIS
  ansible.windows.win_feature:
    name: Web-Server
    state: present

- name: Install IIS with sub features and management tools
  ansible.windows.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)
- (starting from 2.5): transparent SYSTEM + auto reboot

Windows Updates

```
- name: Install only particular KB
  ansible.windows.win_updates:
    accept_list:
    - KB2267602
    - KB890830
    log_path: C:\ansible_wu.txt
```

For *.msu updates, win_hotfix can be considered.

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
```

CONFIGURATION MANAGEMENT & SERVICES

IIS

```
- community.windows.win_iis_website:
    name: Default Web Site
    physical_path: C:\Inetpub\WWWRoot

- community.windows.win_iis_webapplication:
    name: api
    site: acme
    state: present
    physical_path: C:\apps\acme\api
```


Red Hat

Registry

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

Registry

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

- name: merge registry data
  community.windows.win_regmerge:
    path: C:\autodeploy\myCompany-settings.reg
```

ACL

- More granular than Linux permissions
- More like SELinux ACLs

ACL

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

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

Windows Services

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

Windows Services


```
- name: ensure IIS is running
  ansible.windows.win_service:
    name: spooler
    state: running

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

DOMAINS & CREDENTIALS

Domains

- Enterprise identity management
- Makes auth complex
- Promote/demote Domain Controllers
- Joining/leaving domain is simple
- Manage basic domain objects

Domains


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

- name: add an AD user
  community.windows.win_domain_user:
    name: bob
    firstname: Bob
    surname: Smith
    password: xxx
    state: present
```

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
```

WINDOWS DSC

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.

What is DSC?

> Windows Management Platform built in

- Ships natively with Windows Server 2012 R2 and Windows 8.1 and newer
- Requires PowerShell v4 or greater

> Configuration based declarative model

- Define desired state in configuration
- DSC determines how to execute on target

> Push or Pull Architecture

```
configuration DNSServer
{
    Import-DscResource -module 'xDnsServer','xNetworking', 'PSDesiredStateConfiguration'

    Node $AllNodes.Where{$_.Role -eq 'DNSServer'}.NodeName
    {
        WindowsFeature DNS
        {
            Ensure  = 'Present'
            Name    = 'DNS'
        }

        xDnsServerPrimaryZone $Node.zone
        {
            Ensure    = 'Present'
            Name      = $Node.Zone
            DependsOn = '[WindowsFeature]DNS'
        }

        foreach ($ARec in $Node.ARecords.keys) {
            xDnsRecord $ARec
            {
                Ensure    = 'Present'
                Name      = $ARec
                Zone      = $Node.Zone
                Type      = 'ARecord'
                Target    = $Node.ARecords[$ARec]
```

Why Use Ansible & DSC Together?

Both declarative & end-state oriented	**Compliment each other**	**Rich community ecosystem for both**
Extend end-to-end use cases beyond Windows management	**Scale using Ansible lightweight architecture**	**Ansible Tower provides enterprise capabilities managing Windows**

Ansible Windows Modules or DSC Resources?

Reasons for using an Ansible module over a DSC resource:

- The host does not support PowerShell v5.0, or it cannot easily be upgraded
- The DSC resource does not offer a feature present in an Ansible module
- DSC resources have limited check mode support, while some Ansible modules have better checks
- DSC resources do not support diff mode, while some Ansible modules do
- Custom resources require further installation steps to be run on the host beforehand, while Ansible modules are in built-in to Ansible

Reasons for using a DSC resource over an Ansible module:

- The Ansible module does not support a feature present in a DSC resource
- There is no Ansible module available

Example: Ansible Modules vs DSC Resources


```
- name: Install IIS Web-Server
  win_feature:
    name: Web-Server
    state: present
    restart: True
    include_sub_features: True
    include_management_tools: True

- name: Create IIS site
  win_iis_website:
    name: Ansible
    state: started
    physical_path: c:\sites\Ansible

- name: Add HTTP webbinding to IIS
  win_iis_webbinding:
    name: Ansible
    protocol: http
    port: 8080
    ip: '*'
    state: present
```

```
- name: Install required DSC module
  win_psmodule:
    name: xWebAdministration
    state: present

- name: Install IIS Web-Server
  win_dsc:
    resource_name: windowsfeature
    name: Web-Server

- name: Create IIS site
  win_dsc:
    resource_name: xWebsite
    Ensure: Present
    Name: Ansible
    State: Started
    PhysicalPath: c:\sites\Ansible
    BindingInfo:
    - Protocol: http
      Port: 8080
      IPAddress: '*'
```

Example: win_dsc module vs Powershell


```yaml
- name: Install required DSC module
  win_psmodule:
    name: xWebAdministration
    state: present

- name: Install IIS Web-Server
  win_dsc:
    resource_name: windowsfeature
    name: Web-Server

- name: Create IIS site
  win_dsc:
    resource_name: xWebsite
    Ensure: Present
    Name: Ansible
    State: Started
    PhysicalPath: c:\sites\Ansible
    BindingInfo:
    - Protocol: http
      Port: 8080
      IPAddress: '*'
```

```
# Import the module
Import-DscResource -Module xWebAdministration,
PSDesiredStateConfiguration

    Node $NodeName
    {
        # Install the IIS role
        WindowsFeature IIS
        {
            Ensure          = 'Present'
            Name            = 'Web-Server'
        }

        xWebsite DefaultSite
        {
            Ensure          = 'Present'
            Name            = 'Ansible'
            State             = 'Started'
            PhysicalPath    = 'c:\sites\Ansible'
            DependsOn       = '[WindowsFeature]IIS'
            BindingInfo     =
MSFT_xWebBindingInformation
            {
                Protocol              = 'http'
                Port                  = '8080'
                IPAddress             = '*'
            }
        }
    }
```


Red Hat

Handle Credentials with `win_dsc` Module

- By default `win_dsc` module uses `SYSTEM` account
- You can use `PsDscRunAsCredential` attribute to run as another user:

```
- 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
```

Example DSC Resources

- Built-in:
  - Archive
  - File
  - Group
  - Package
  - WindowsFeature
  - And more..

- Custom resources provided by Microsoft and the community:
  - Domain Controller
  - IIS Web Site
  - SQL Server Cluster
  - Failover Cluster
  - DNS
  - And many more..

Wrap Up

Windows is a first class citizen within the Ansible ecosystem!

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


youtube.com/user/RedHatVideos


facebook.com/redhatinc


twitter.com/RedHat

Red Hat