Breaking the Silos with Ansible

Christophe Roux
Solution Architect

Swisscom
Enterprise Customers
Cloud & Data Center Services

Who am I?

Contact information:

Christophe Roux
Solution Architect

Swisscom
Cloud & Datacenter Services

Email: christophe.roux@swisscom.com
Phone: +41 58 223 38 31
Mobile: +41 79 477 20 85

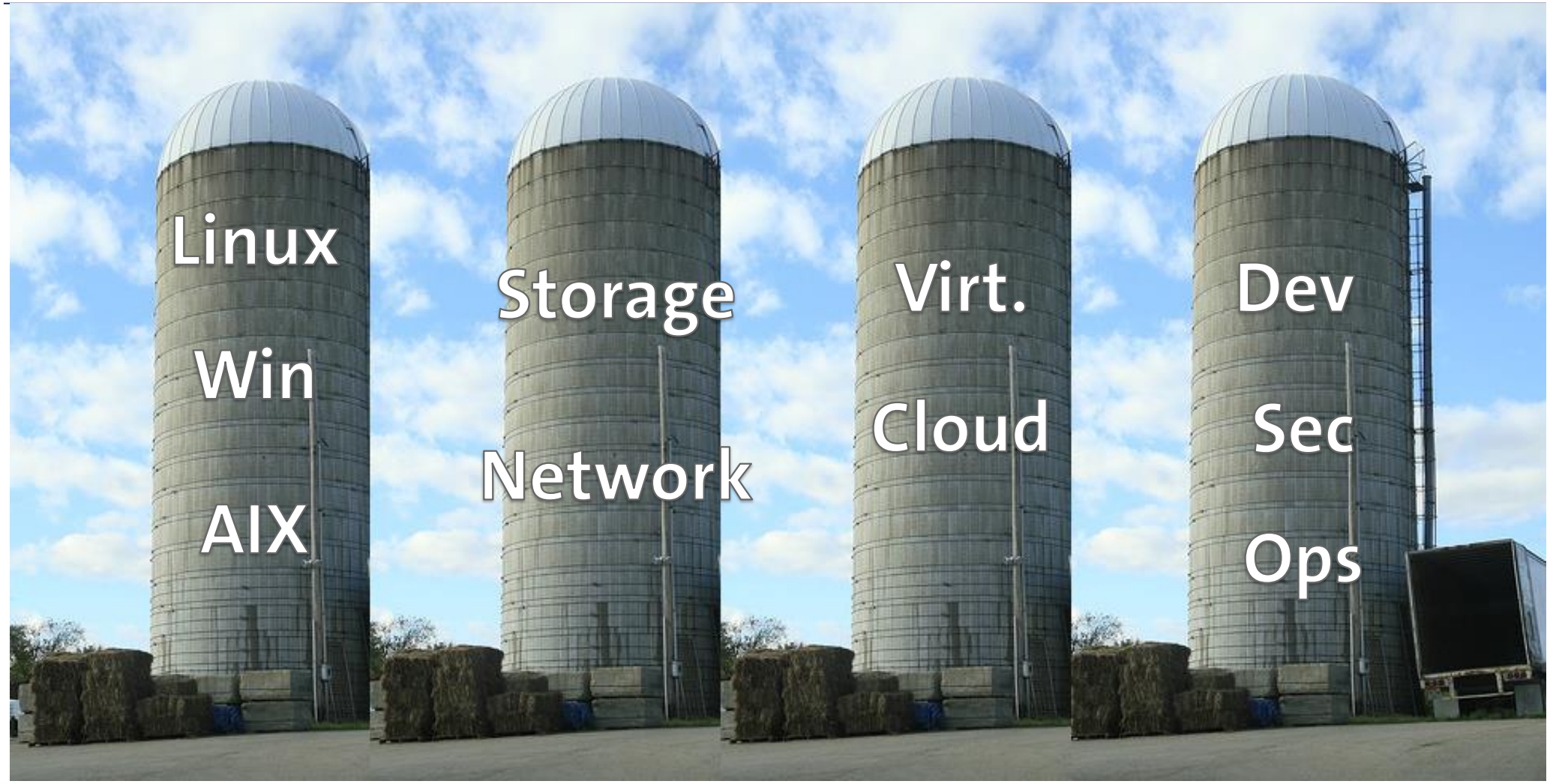
swisscom

Automation is key to Digital Transformation

> Digital business gives IT infrastructure and operations leaders a mandate to implement organization-wide automation...
>
> The benefits are compelling - automation improves accountability, efficiency and predictability, while reducing cost, variability and risk...

Gartner – The Next Frontier for IT
https://www.gartner.com/smarterwithgartner/automation-the-next-frontier-for-it-2/

Automation Silos



swisscom

Organization Automation

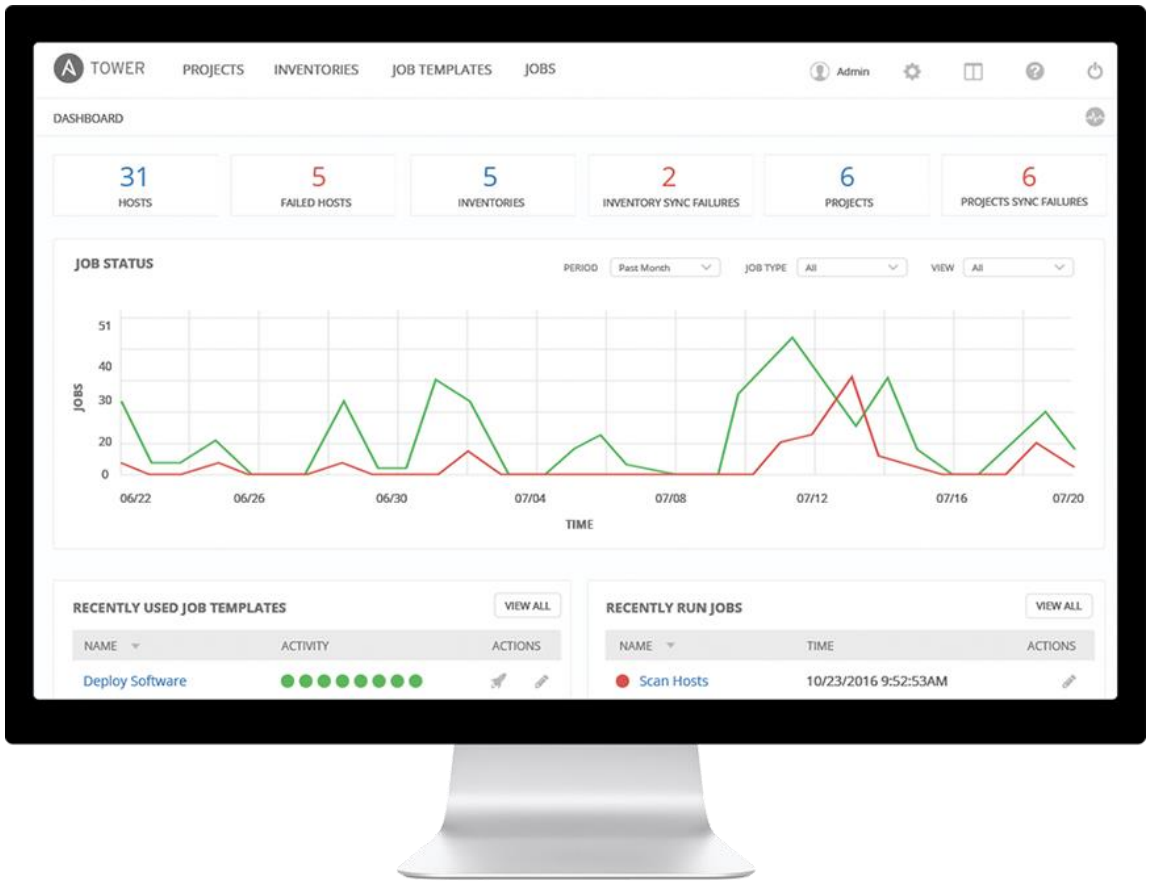
Isolated Scripts	Culture of Automation
⌄	⌄
Automates Functions	Automates Organizations

Automation at scale must be simple

AUTOMATION FOR EVERYONE

What is Ansible

- It's a **simple automation language** that can perfectly describe an IT application infrastructure in Ansible Playbooks.
- It's an **automation engine** that runs Ansible Playbooks.
- Ansible Tower is an **enterprise framework** for controlling, securing and managing your Ansible automation with a UI and restful API.



Principles of Ansible

SIMPLE

- Human Readable
- No special coding skills required
- Task executed in order
- Usable by every team

Get productive quickly

POWERFULL

- Application Deployment
- Configuration Management
- Workflow Orchestration
- Network Automation

Orchestrate the app lifecycle

AGENTLESS

- Agentless architecture
- Use OpenSSH & WinRM
- Lower vulnerability footprint
- Get Started immediately
- Pull based

More efficient & more secure

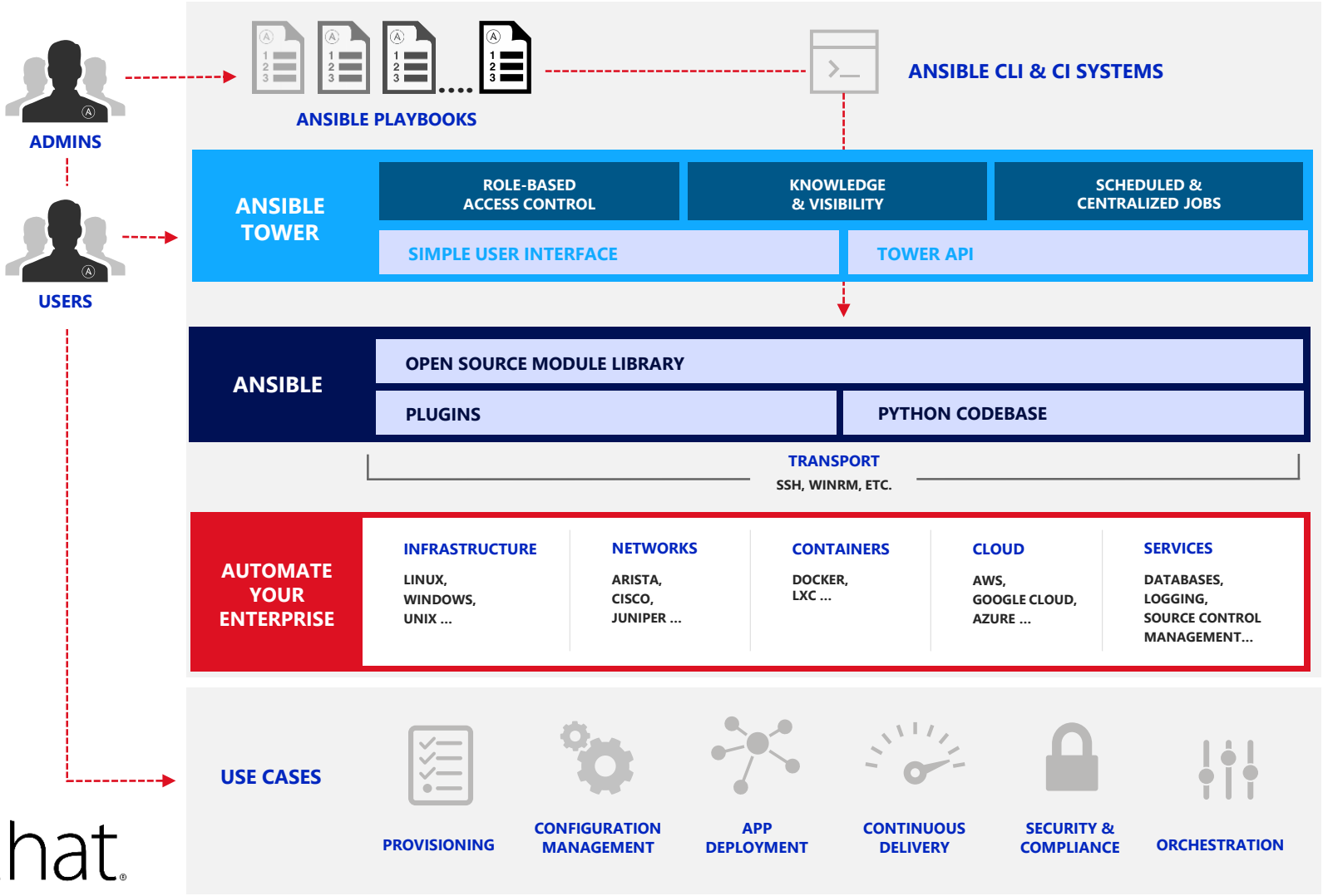
Ansible automates the technology you use

Cloud	Virt. & Container	Windows	Network	Chat	Monitoring
AWS	Atomic	ACL	Arista	Email	Nagios
Azure	CloudSTack	Files	Cisco	Jabber	Zabbix
Google	OpenSTack	Packages	Juniper	IRC	New Relic
OpenSTack	OpenShift	IIS	F5	Slack	Datadog
Rackpace	RHEV	Registry	Palo Alto		
	Vmware	Services	OpenSwitch		
		Users			
		Domains			
+ More	**+ More**	**+ More**	**+ More**	**+ More**	**+ More**

MORE THAN 1200 MODULES ..... And counting

swisscom

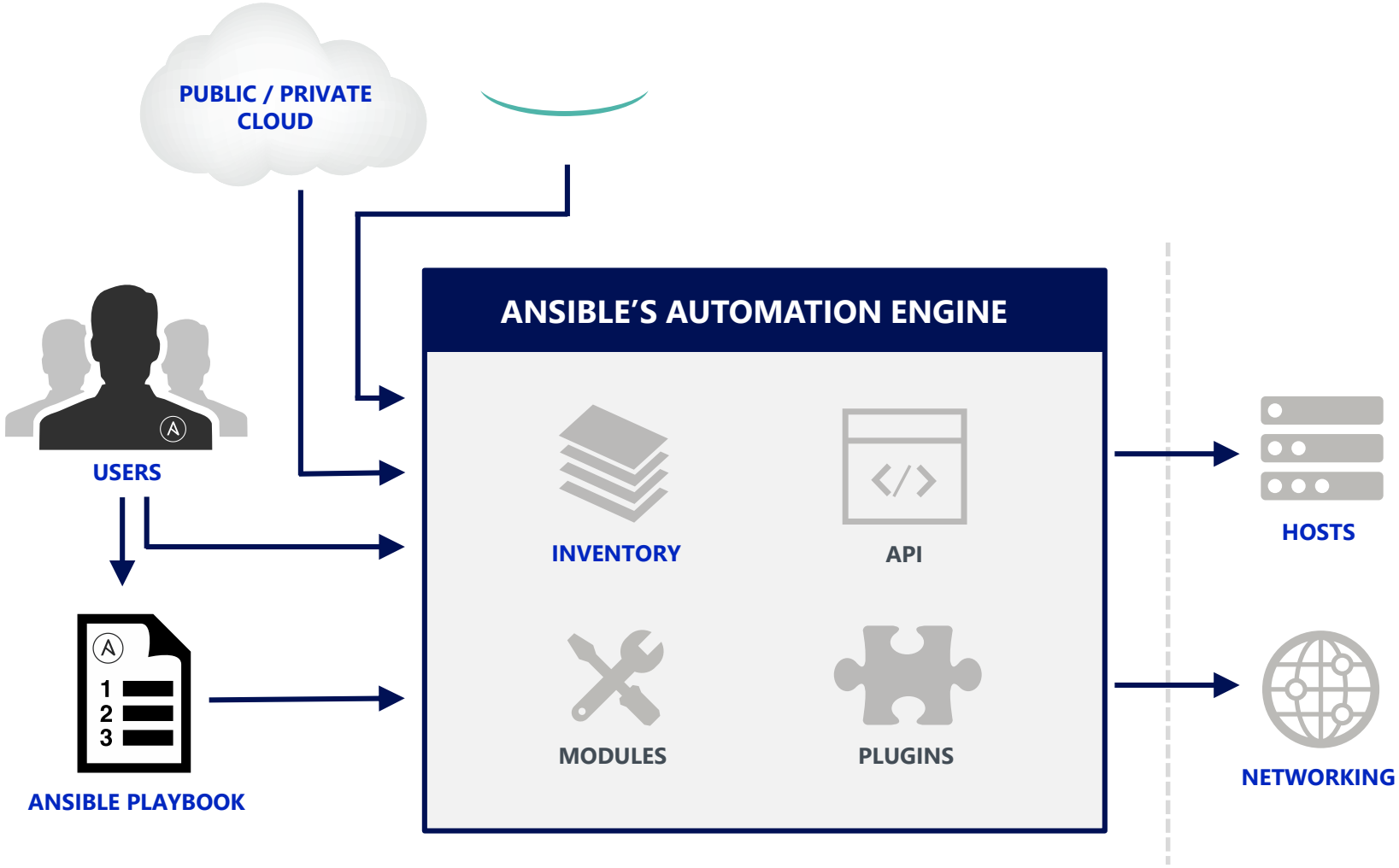
Ansible Architecture



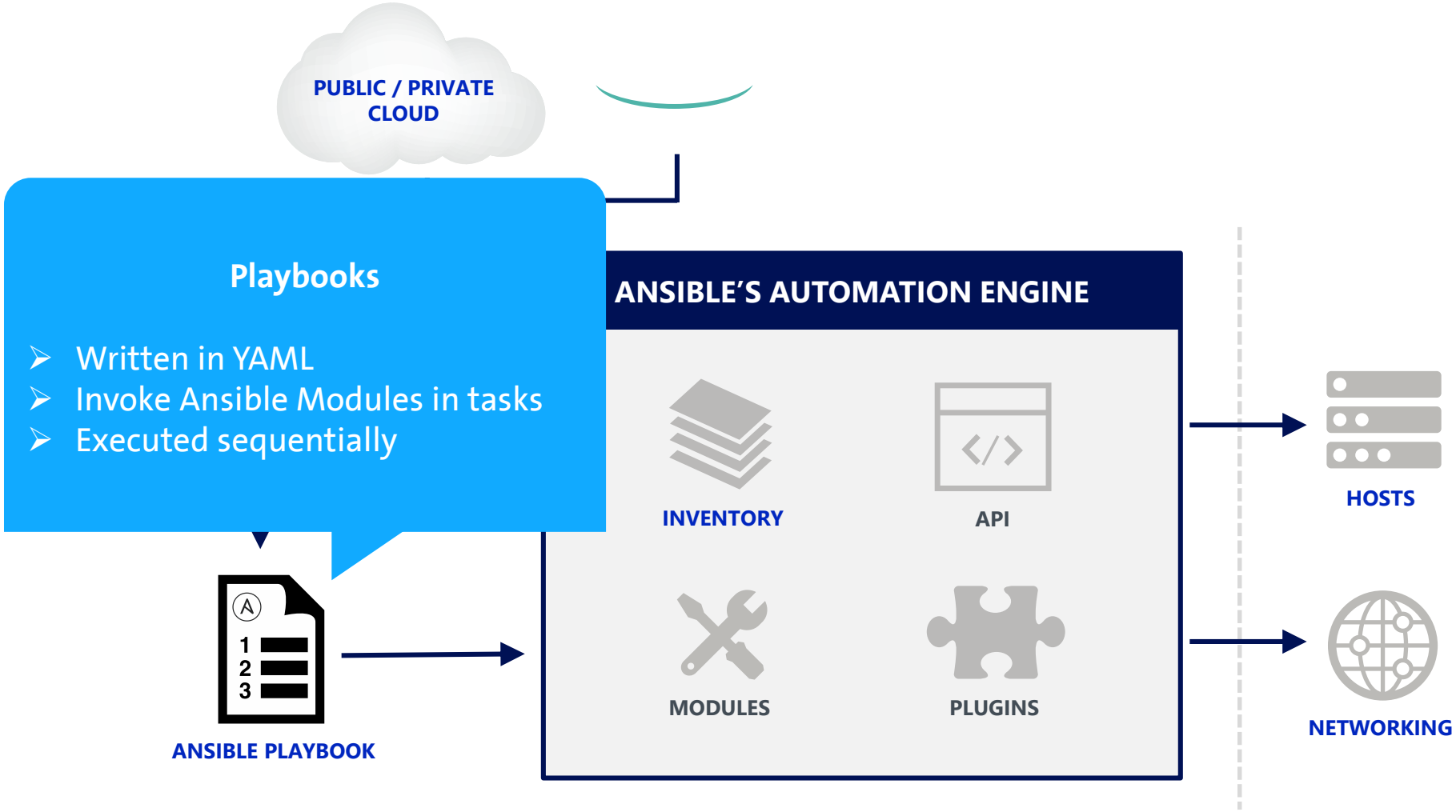
redhat

swisscom

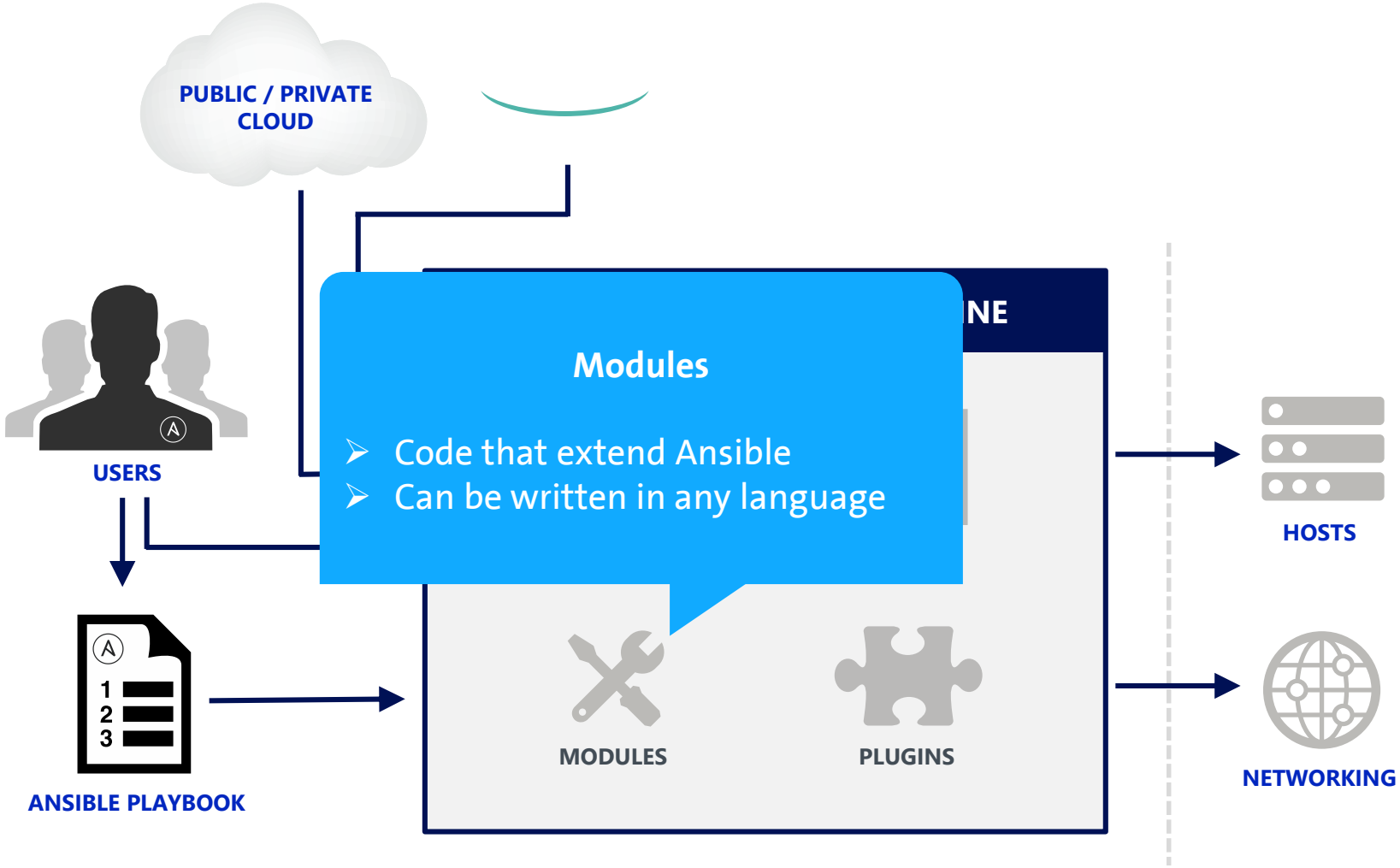
How Ansible Works



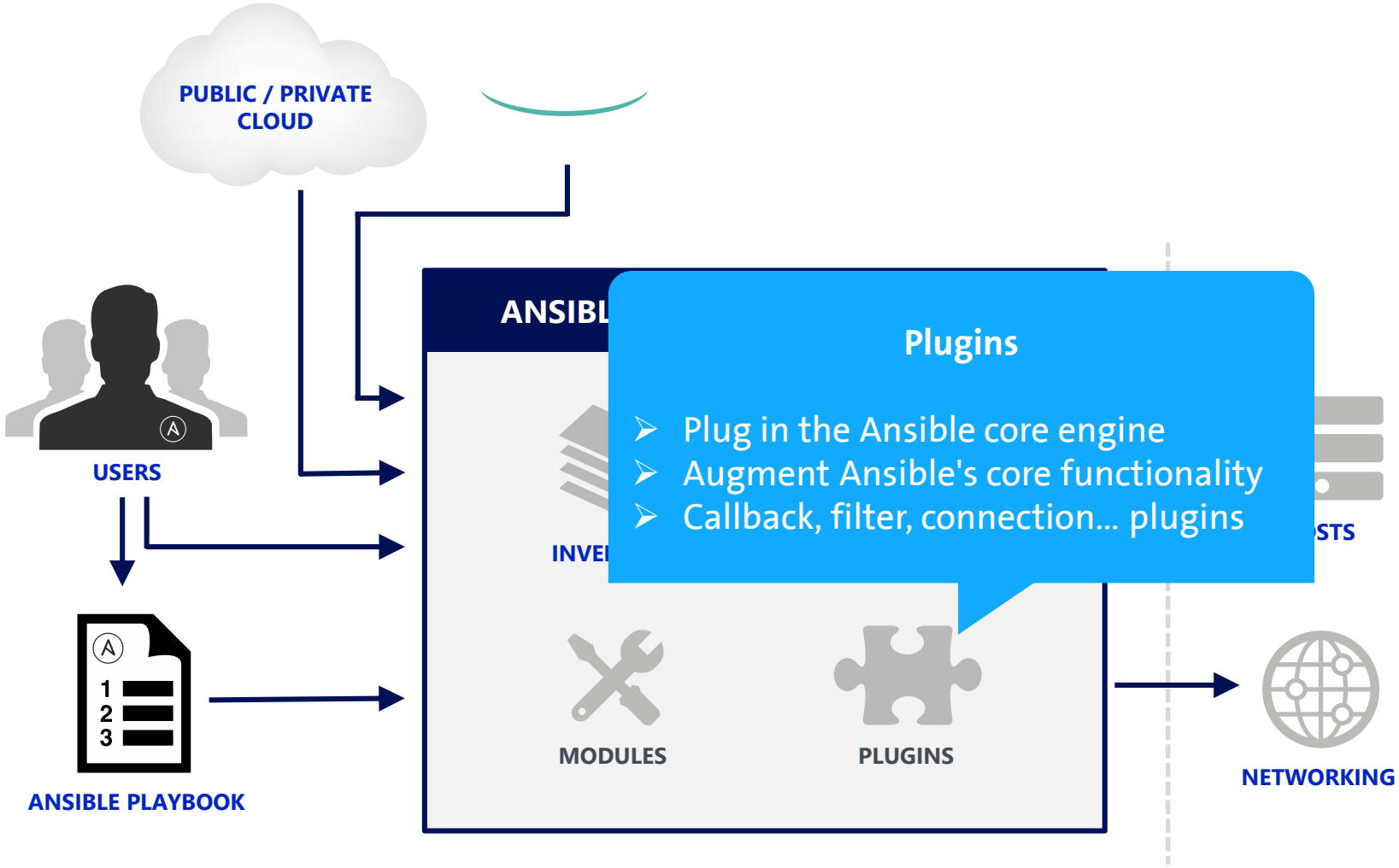
How Ansible Works



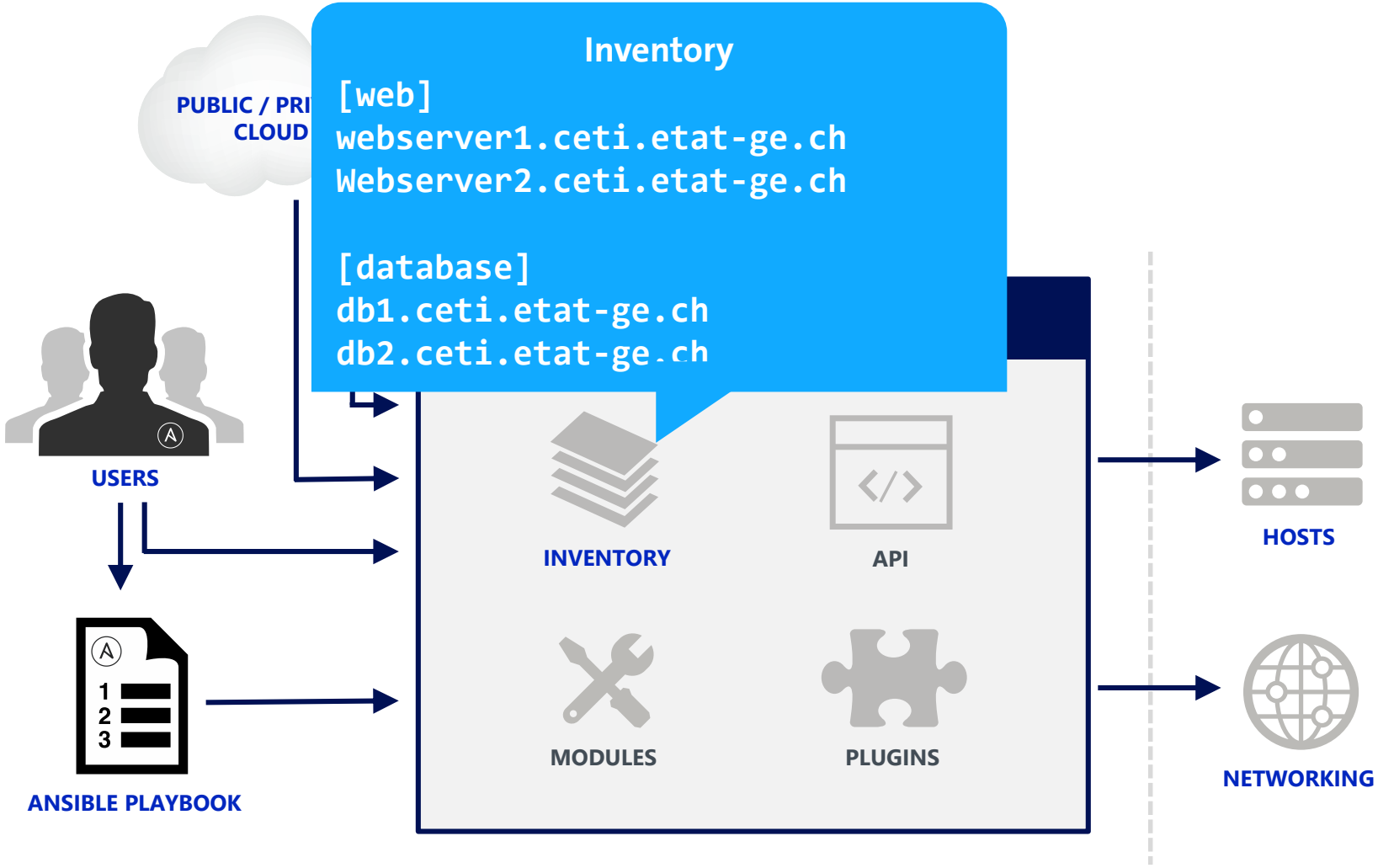
How Ansible Works



How Ansible Works



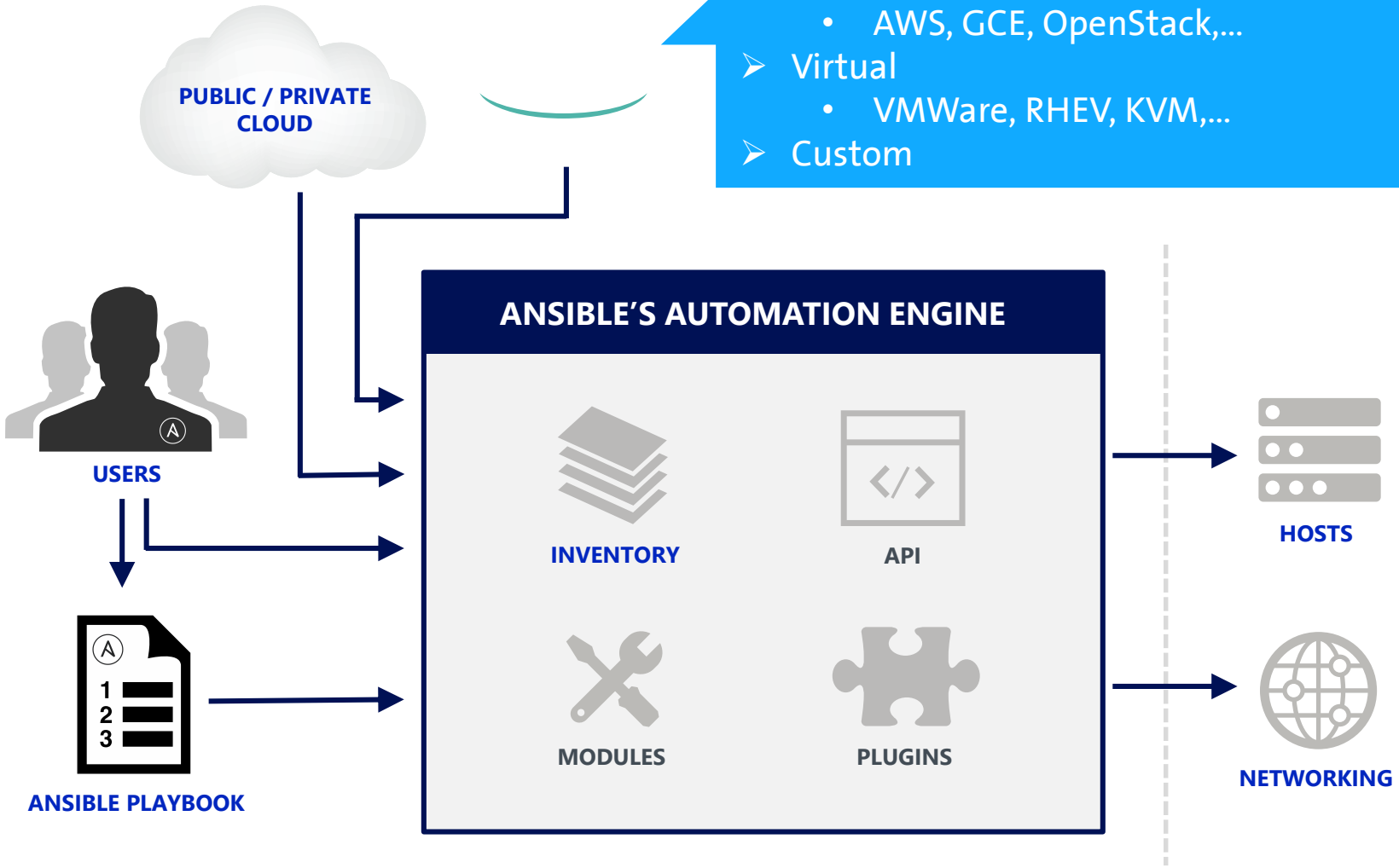
How Ansible Works



How Ansible Works

Dynamic Inventory

- Cloud
  - AWS, GCE, OpenStack,...
- Virtual
  - VMWare, RHEV, KVM,...
- Custom



DEMO

swisscom

Playbook Example


```
---
- name: install and start apache
  hosts: webserver
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

Playbook Example


```
---
- name: install and start apache
  hosts: webserver
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

Playbook Example


```
---
- name: install and start apache
  hosts: webserver
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

Playbook Example


```
---
- name: install and start apache
  hosts: webserver
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

Playbook Example


```
---
- name: install and start apache
  hosts: webserver
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: name=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

Playbook execution example

- Execute the playbook

```
# ansible-playbook -i ./hosts playbook.yml
PLAY [My First Playbook] ***************************************************

TASK [setup] ***************************************************************
ok: [www01.example.com]

TASK [Install apache] ******************************************************
changed: [www01.example.com]

PLAY RECAP *****************************************************************
www01.example.com          : ok=2    changed=1    unreachable=0    failed=0
```

Playbook execution example

> Re-run it

```
PLAY [My First Playbook] ***************************************************

TASK [setup] ***************************************************************
ok: [www01.example.com]

TASK [Install apache] ******************************************************
ok: [www01.example.com]

PLAY RECAP *****************************************************************
www01.example.com          : ok=2    changed=0    unreachable=0    failed=0
```

”

Idempotency:

An operation is idempotent if the result of performing it once is exactly the same as the result of performing it repeatedly without any intervening actions.

“

swisscom

Ansible Roles

- Package of closely related Ansible content that can be shared more easily than a play alone
- Improves readability and maintainability of large and complex plays
- Eases sharing, reuse and standardization of automation processes
- Enables Ansible content to exists independently of playbooks, projects or even organizations
- Provides some "magic" around "include"
  - Search path handling
  - Default values

Roles - How to use

- Add the roles to your playbooks

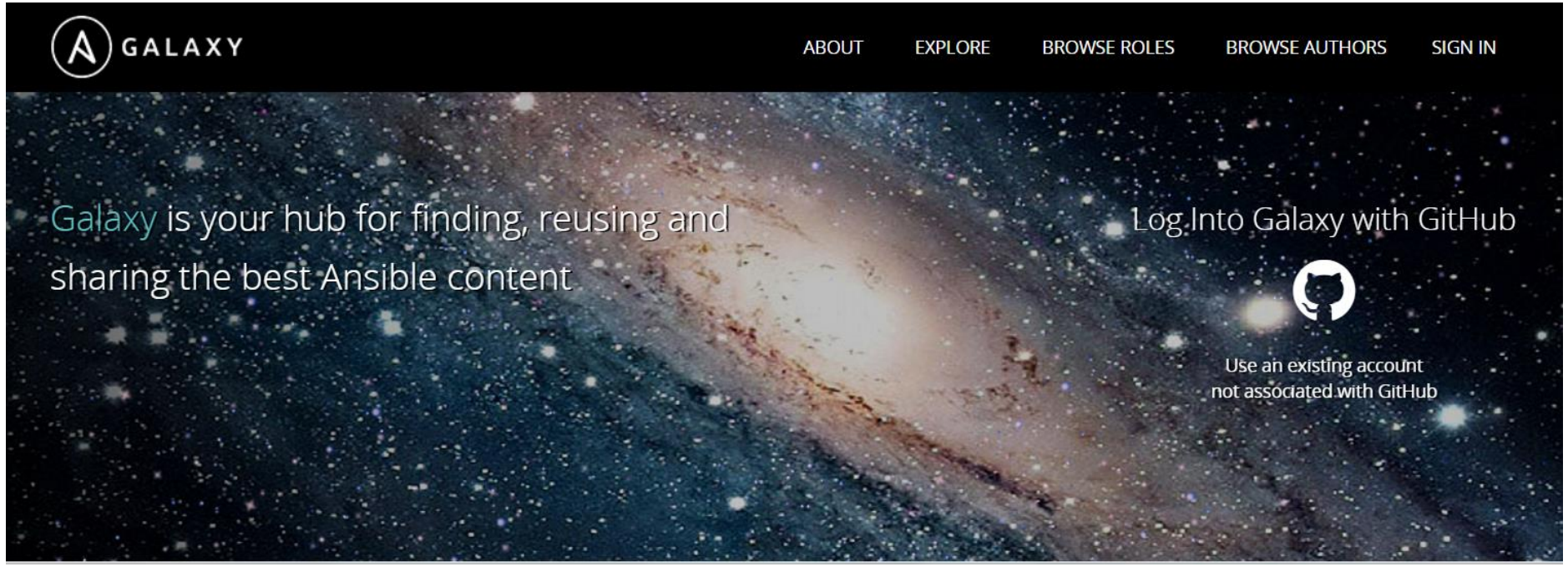
```
---
- hosts: web
  roles:
    - common
    - webtier
- hosts: db
  roles:
    - common
    - dbtier
```

- Parametrized Roles

```
---
- hosts: web
  roles:
    - common
    - role: myoracle
      sid: orcl1
      oracle_version: 11.2.0.4
    - role: myoracle
      sid: orcl2
      oracle_version: 12.2.0.1
```

Ansible Galaxy

- Hub for finding, sharing and reusing Ansible roles
- http://galaxy.ansible.com



Ansible Galaxy - Installing roles from Galaxy

- Single role install

```
$ ansible-galaxy install username.role
```

  - This will install the role in /etc/ansible/roles by default
  - or the value of role_path variable from Ansible config file

- Single Role install to specific location

```
$ ansible-galaxy install username.role -p <path_to_roles_directory>
```

- Multiple roles install

```
$ ansible-galaxy install -r roles.txt
```

```
# Roles.txt
user1.role1,v1.0.0
user2.role2,v0.5
user2.role3
```

Ansible Galaxy - Installing roles from version control repository

requirements.yml

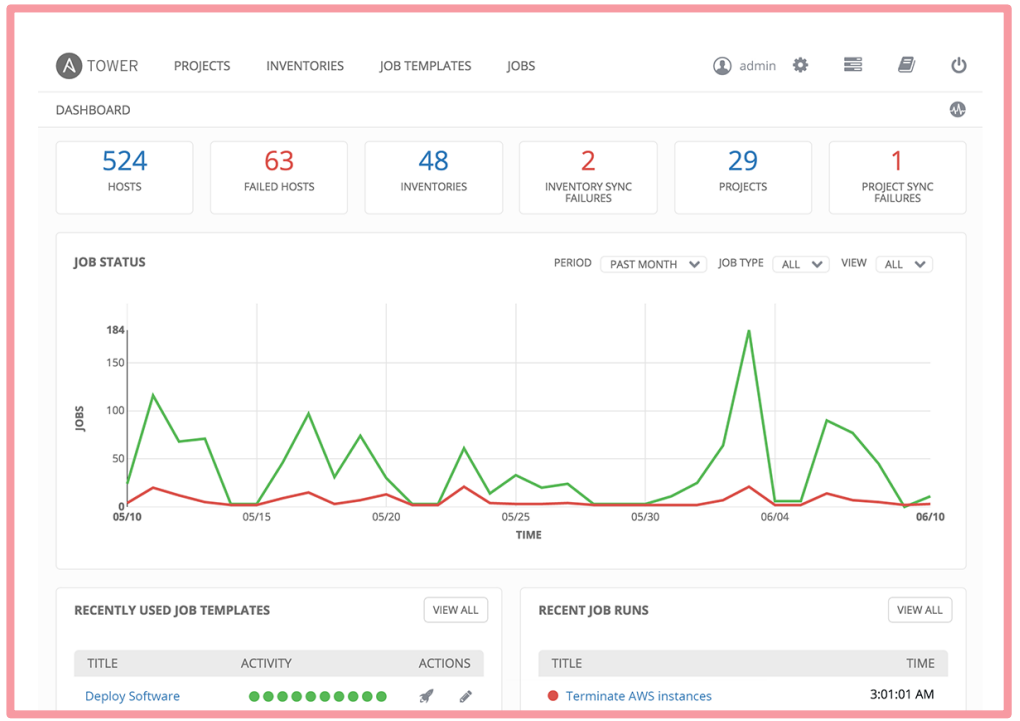
```
---
# From github
- src: https://github.com/bennojoy/nginx

# from github, overriding the name and specifying a specific tag
- src: https://github.com/bennojoy/nginx
  version: master
  name: nginx_role

# from github, overriding the name and specifying a specific tag
- src: https://github.com/bennojoy/nginx
  version: master
  name: nginx_role
```

Install

```
$ ansible-galaxy install -r requirements.yml
```

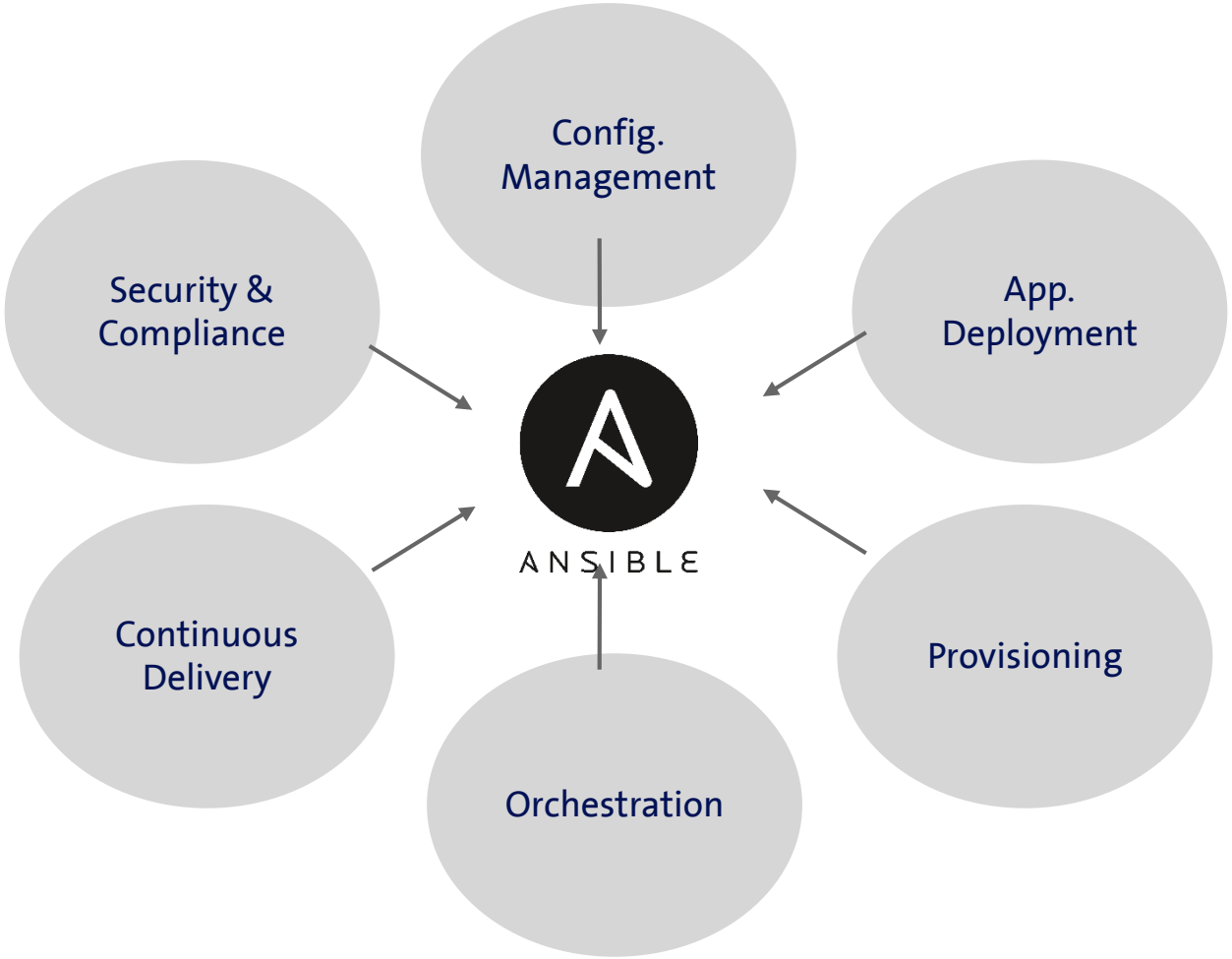
What is Ansible Tower?

- Ansible tower is an **enterprise framework** for controlling, securing and managing your Ansible automation with a **UI and RESTful API**.
- **Role-based access control** keeps environments secure, and teams efficient.
- Non-privileged users can **safely deploy** entire applications with **push-button deployment** access.
- All Ansible automations are **centrally logged**, ensuring **complete auditability and compliance**.

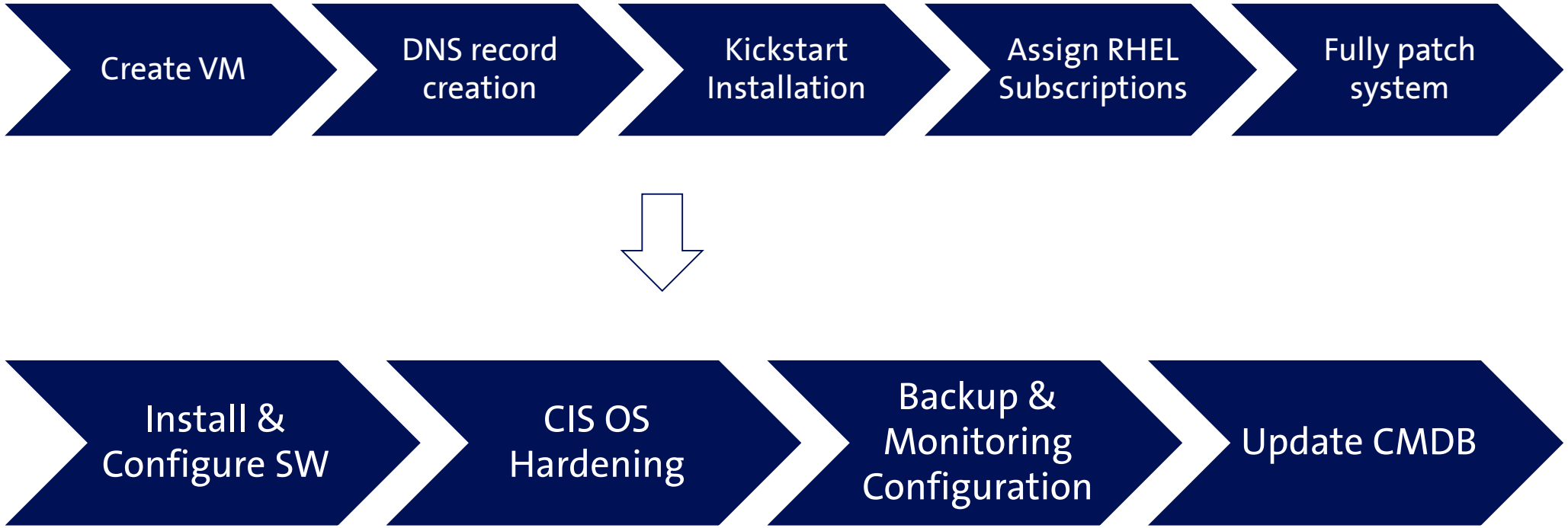


swisscom

Ansible Use Cases



Example Projet Ansible - RHEL server provisioning on Red Hat Virt.



What about Ansible and AIX

- Ansible Control Server must run on Linux

- Pre-requisites for AIX managed nodes
  - Openssh
  - Python 2.7 or Python 3

What about Ansible and AIX

- Lots of Ansible modules works out of the box
  - user, group, authorized_keys
  - template, copy, file
  - mount...
- Some specific to Linux don't
  - Yum, services,...
- But you can fall-back to "command" and "shell" modules
- Or write your own modules


```
- name: create dba group
  group:
    name: dba
    gid: 200
    state: present
- name: create croux user
  user:
    name: croux
    comment: Christophe Roux
    uid: 1200
    group: dba
    generate_ssh_key: True
    ssh_key_type: ecdsa
- name: deploy authorized key
  authorized_key:
    user: root
    key="{{ lookup('file',
         '/Users/croux/.ssh/id_rsa.pub') }}"
```

What about Ansible and AIX

- Some work is done by the community to add AIX specific modules

  https://github.com/kairoaraujo/ansible-aix-support

Pull requests list

List of pull requests already done to official Ansible

Type	Name	Pull Request link	Status	Version
module	aix_lvg	https://github.com/ansible/ansible/pull/30381	Open	devel (2.5.0)
module	aix_filesystem	https://github.com/ansible/ansible/pull/30810	Open	devel (2.5.0)
module	mksysb	https://github.com/ansible/ansible/pull/30460	Merged	devel (2.5.0)
module	installp	https://github.com/ansible/ansible/pull/30238	Open	devel (2.5.0)
utils	aix	https://github.com/ansible/ansible/pull/31546	Open	devel (2.5.0)
utils	distribution	https://github.com/ansible/ansible/pull/31754	Merged	devel (2.5.0)
module	aix_devices	https://github.com/ansible/ansible/pull/32290	Open	devel (2.5.0)
module	service	https://github.com/ansible/ansible/pull/33043	Open	devel (2.5.0)

AIX Specific Modules Examples


```
- name: Running a backup image mksysb
  mksysb:
    name: myserver
    storage_path: /repository/images
    exclude_files: yes
    exclude_wpar_files: yes

- name: Configure IP, netmask and set en1 up.
  aix_devices:
    device: en1
    attributes:
      netaddr: 192.168.0.100
      netmask: 255.255.255.0
      state: up
    state: present

- name: Creating a new file system without a previously logical volume.
  aix_filesystem:
    filesystem: /newfs
    size: 1G
    state: present
    vg: datavg
```

swisscom

Ansible and PowerVC

- Comprehensive support of OpenStack API
  - os_auth
  - os_server
  - os_server_facts
  - os_server_action
  - os_server_volume
  - …
- PowerVC exposes the OpenStack API
- Vanilla OS_* OpenStack modules should work!!


```
- name: Provision a New VM os_server:
  state: present
  auth:
    auth_url: https://openstackserver:5000/v3
    username: "{{ username }}"
    password: "{{ password }}"
    os-user-domain-name: Default
    os-user-domain-id: default
    os-project-domain-name: Default
    project_id: 1225tgfd54a54b28b332605
    project_name: default
    region_name: RegionOne
  image: 69853124-a87a-48bf-a15a-5863
  flavor: "{{ Flavour }}"
  name: "{{ Hostname }}"
  auto_ip: no
  nics:
    - port-name: "{{ Hostname }}-port"
  meta:
    hostname: "{{ Hostname }}"
  config_drive: true
  userdata: "{{ user_data.stdout }}"
  validate_certs: no
register: new_vm
ignore_errors: false
```

swisscom

Some Ansible Good Practices

- Version Control Everything
  - Playbooks
  - Roles
  - Inventory
  - Variables
- Use Roles for reusability and manageability
- Implement automated testing for your playbooks and roles

Ansible - Testing

- **Syntax check**

```
ansible-playbook site.yml --syntax-check
```

- Ansible Lint
- Ansible Rolespec
- Smoke tests in Playbooks

Ansible - Testing

- Syntax check
- **Ansible Lint**
  https://github.com/willthames/ansible-lint
- Ansible Rolespec
- Smoke tests in Playbooks

```
 $ ansible-lint examples/example.yml
[ANSIBLE0004] Git checkouts must contain explicit version
examples/example.yml:15
Task/Handler: git check

[ANSIBLE0004] Git checkouts must contain explicit version
examples/example.yml:18
Task/Handler: git check 2

[ANSIBLE0004] Git checkouts must contain explicit version
examples/example.yml:30
Task/Handler: using git module

[ANSIBLE0002] Trailing whitespace
examples/example.yml:13
    action: do nothing

[ANSIBLE0002] Trailing whitespace
examples/example.yml:35
    with_items:

[ANSIBLE0006] git used in place of git module
examples/example.yml:24
Task/Handler: executing git through command

[ANSIBLE0006] git used in place of git module
examples/example.yml:27
Task/Handler: executing git through command

[ANSIBLE0006] git used in place of git module
examples/example.yml:30
Task/Handler: executing git through command
```

Ansible - Testing

- Syntax check
- Ansible Lint
- **Ansible Rolespec**
  https://github.com/nickjj/rolespec
- Smoke tests in Playbooks

```
 assert_playbook_runs
assert_playbook_idempotent
assert_playbook_idempotent_long

assert_permission "/srv/users/testapp" "751"
assert_user_in_group "testuser" "audio"

assert_in_file "/srv/users/testapp/.ssh/id_rsa" "ssh"
assert_in_file "/etc/logrotate.d/testapp" "{.*}"

assert_url "https://${ROLESPEC_FQDN}"
assert_url "https://${ROLESPEC_FQDN}/sidekiq"

assert_monitoring "testapp"
assert_monitoring "sidekiq"
```

Ansible - Testing

- Syntax check
- **Ansible Lint**
- Ansible Rolespec
- **Smoke tests in Playbooks**

```
-
    name: Test if url is respoinding
    uri:
      url: http://localhost:8080/index.html
      return_content: true
    register: response
    until: "'Welcome' in response.content"
    retries: 10
    delay: 1
```

Questions?

Contact information:

Christophe Roux

Email: christophe.roux@swisscom.com
Phone: +41 58 223 38 31
Mobile: +41 79 477 20 85

swisscom