Ansible

Kurs DevOps – Wykład 3

Kuba Nowak

Uniwersytet Wrocławski

28 października 2025

Konfiguracja

- 1 komputer można skonfigurować ręcznie,

Konfiguracja

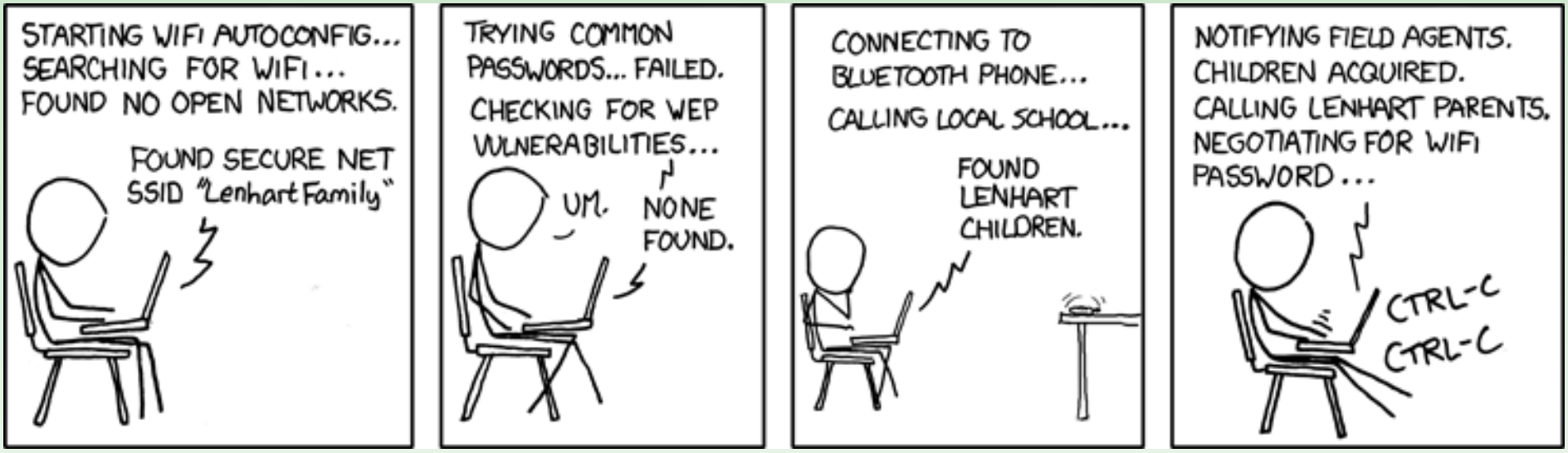
- 1 komputer można skonfigurować ręcznie,
- 5 komputerów można skonfigurować ręcznie,

Konfiguracja

- 1 komputer można skonfigurować ręcznie,
- 5 komputerów można skonfigurować ręcznie,
- 30 komputerów... też można skonfigurować ręcznie,

Konfiguracja

- 1 komputer można skonfigurować ręcznie,
- 5 komputerów można skonfigurować ręcznie,
- 30 komputerów... też można skonfigurować ręcznie,
- 1000 komputerów...?



Źródło: `https://xkcd.com/416/`

Orkiestracja

Przykładowe zadanie:

- Usuń wybrane serwery z load balansera
- Poczekaj aż serwery zakończą pracę
- Wyłącz monitoring
- Zamknij aplikacje na serwerze
- Zaktualizuj aplikację
- Zresetuj serwery
- Włącz monitoring
- Dodaj serwery do load balansera

Orkiestracja

Przykładowe zadanie:

- Usuń wybrane serwery z load balansera
- Poczekaj aż serwery zakończą pracę
- Wyłącz monitoring
- Zamknij aplikacje na serwerze
- Zaktualizuj aplikację
- Zresetuj serwery
- Włącz monitoring
- Dodaj serwery do load balansera

Wikipedia

Orchestration is the automated configuration, coordination, deployment, development, and management of computer systems and software.
`https://en.wikipedia.org/wiki/Orchestration_(computing)`

Infrastructure as a Code

Wikipedia

Infrastructure as code (IaC) is the process of managing and provisioning computer data center resources through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.
https://en.wikipedia.org/wiki/Infrastructure_as_code

- Zarządzalność
- Skalowalność
- Reprodukowalność
- Kontrola wersji

Opiszmy naszą infrastrukturę w plikach tekstowych, które następnie zostaną zinterpretowane ("wykonane") przez wyspecjalizowane oprogramowanie.

Push vs pull

push

Serwer inicjalizuje wysyłkę konfiguracji/poleceń do węzła niewolnego.

pull

Węzeł niewolny sam odpytuje serwer, czy jest nowa konfiguracja do wdrożenia.

Push vs pull

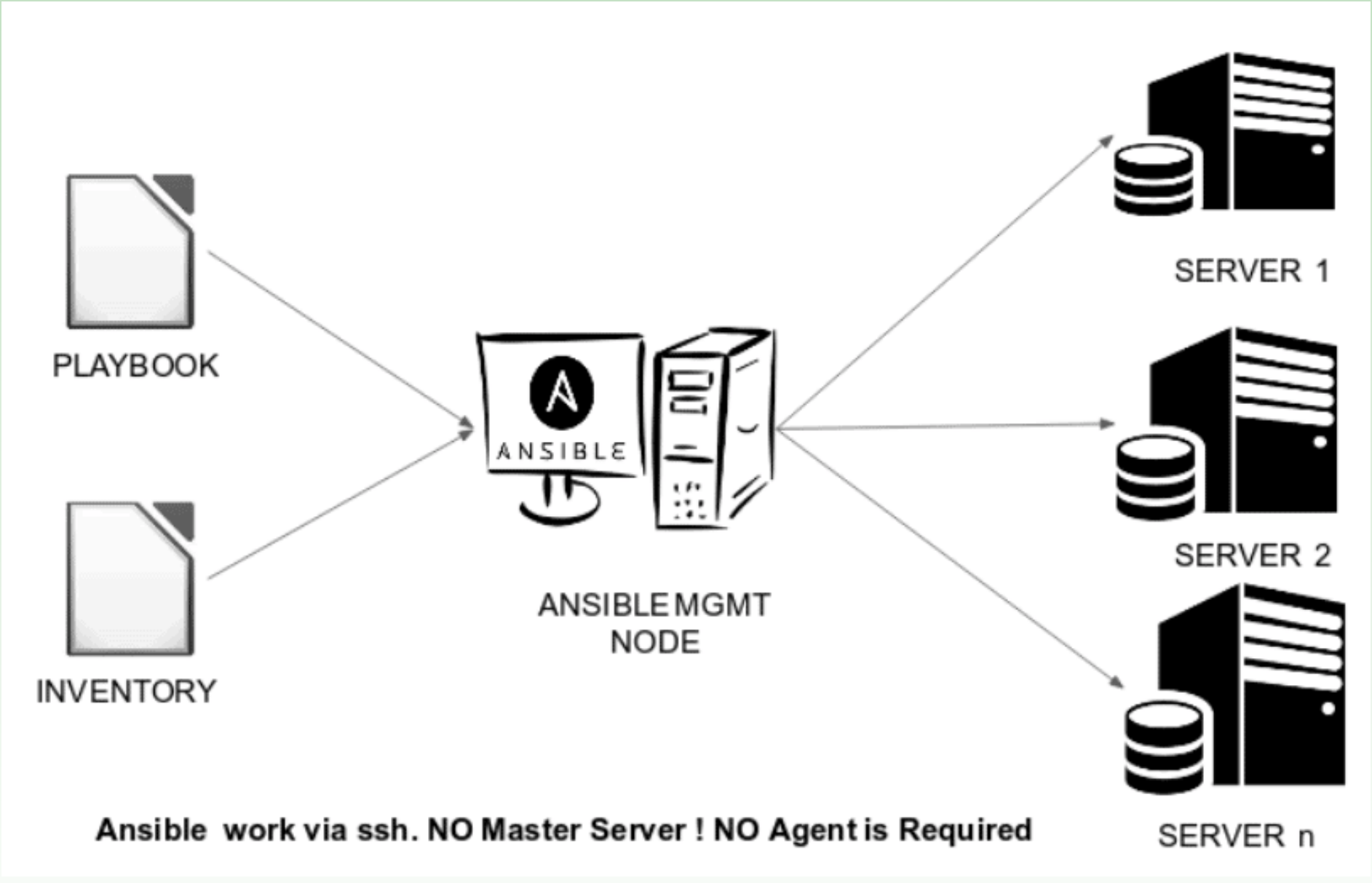
Push	Pull
Potencjalnie bezagentowy	Wymagany agent
Wydajność ograniczona serwerem	Potencjalnie nieskończone skalowanie

Przykładowe narzędzia do konfiguracji

- Puppet (2005)
- Chef (2009)
- Ansible (2012)
- Terraform (2014)
- OpenTofu (2023)

Przykładowe narzędzia do konfiguracji

- Puppet (2005)
- Chef (2009)
- **Ansible (2012)**
- Terraform (2014)
- OpenTofu (2023)



Źródło: https://k21academy.com/ansible/q-a-day1-live-session-review/

Ansible

- Bezagentowy
- Domyślnie model "push"
- Minimalne zależności: `ssh` oraz `python`
- Rozszerzalny/modularny
- Idempotentny

Zastosowania

Przykładowe zastosowania:

- Elminacja powtórzeń i uproszczenie workflow
- Zarządzanie i utrzymywanie systemu
- Ciągły deployment oprogramowania
- Zero-downtime rolling updates

Elementy systemu

Control node Węzeł, na którym jest zainstalowany Ansible i gdzie uruchamiane są polecenia.

Managed node Węzeł konfigurowany przez Ansible.

Inwentarz Lista węzłów zarządzalnych, zorganizowanych w grupy. Przechowywana na węźle zarządzającym.

Inwentarz

Definicja

Plik, bądź ich zbiór, opisujący węzły, którymi ma zarządzać Ansible. Minimalnie wymagany jest adres IP. Domyślnie w formacie INI bądź YAML.

Przykładowa zawartość:

- IP
- Port
- Użytkownik
- Zmienne specyficzne dla hosta
- Grupowanie (np. lokalizacjami, przeznaczeniem)

Grupy w inwentarzu

Zazwyczaj stosowane do określenia:

- **Co?** Aplikacja lub serwis uruchomiony na maszynach z danej grupy.
- **Gdzie?** Lokalizacja maszyny.
- **Kiedy?** Etap cyklu produkcyjnego, na którym maszyna jest używana.

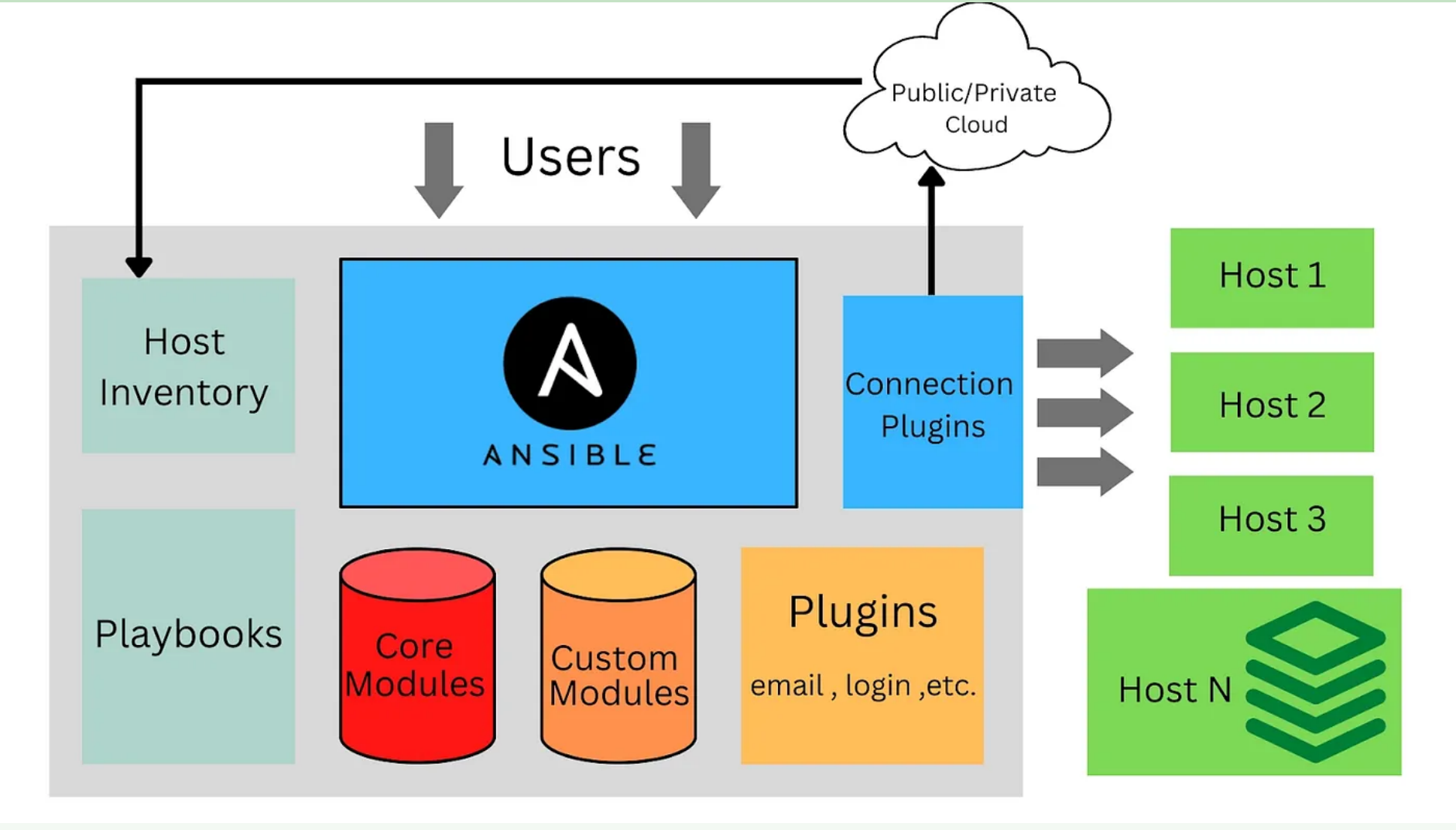
Jeden host może być w inwentarzu wiele razy, jako członek wielu różnych grup.

Grupy domyślne

- `all`
- `ungrouped`

Demonstracja 1

Prosty inwentarz



Źródło: `https://blog.devops.dev/writing-ansible-modules-with-support-for-diff-mode-cae70de1c25f`

Struktura konfiguracji

Playbook Lista ścieżek definiująca porządek wykonania operacji przez Ansible w celu osiągnięcia celu.

Ścieżka (ang. *play*) Uporządkowana lista tasków do wykonania na określonych węzłach.

Task Odniesienie do pojedynczego modułu, który ma zostać uruchomiony w celu wykonania operacji.

Moduł Kod bądź plik binarny uruchamiany przez Ansible na slave node. Moduły Ansible są zgrupowane w kolekcje z Fully Qualified Collection Name (FQCN) dla każdego modułu.

Rola Niezależny pakiet konfiguracyjny, który może być dystrybowany do innych użytkowników (czyt. biblioteka).

Handler Wyspecjalizowany task, który uruchamia się tylko wtedy, gdy zostanie wywołany przez jakiś wcześniejszy task.

Wykonanie

Procedura:

1. Generowanie parametrów (przy pomocy jinja2)
2. Podstawienie parametrów w modułach
3. Skopiowanie plików modułów (python) do maszyn zarządzanych
4. Wykonanie kodu na maszynach zarządzanych
5. Odczytanie wyniku i zamknięcie sesji ssh

Cechy:

- Synchronicznie
- Długie zadania mogą timeoutować na warstwie ssh
- Ograniczenie w liczbie na raz konfigurowanych hostów

`https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_strategies.html`

Demonstracja 2

Polecenia *ad-hoc*

Jinja2

Biblioteka służąca do generowania plików tekstowych:

- `{%...%}` Instrukcje (np. `for`, `if`, `set`, `call`)
- `{{...}}` Wypisywane wyrażenia (np. `"string"`; `[1,2,3] * 4`)
- `{#...#}` Komentarze

https://jinja.palletsprojects.com/en/stable/templates/

Filtrowanie zmiennych/programowalność

- Zmienne Ansible są dostępne jako zmienne Jinja
- Każdy string w pliku Ansible podlega interpretacji/podstawianiu przez Jinja.
- Podstawianie odbywa się na master węźle.
- Przy pomocy *lookup plugin* można odczytywać dane z systemu i podstawiać je z użyciem Jinja (np. `fileglob`, `file`, `ini`).
- Filtry pozwalają w sposób funkcyjny przetwarzać otrzymane dane (np. `range`, `sort`, `max`).

Filtrowanie zmiennych/programowalność

- Zmienne Ansible są dostępne jako zmienne Jinja
- Każdy string w pliku Ansible podlega interpretacji/podstawianiu przez Jinja.
- **Podstawianie odbywa się na master węźle.**
- Przy pomocy *lookup plugin* można odczytywać dane z systemu i podstawiać je z użyciem Jinja (np. `fileglob`, `file`, `ini`).
- Filtry pozwalają w sposób funkcyjny przetwarzać otrzymane dane (np. `range`, `sort`, `max`).

Testy i wykonanie warunkowe

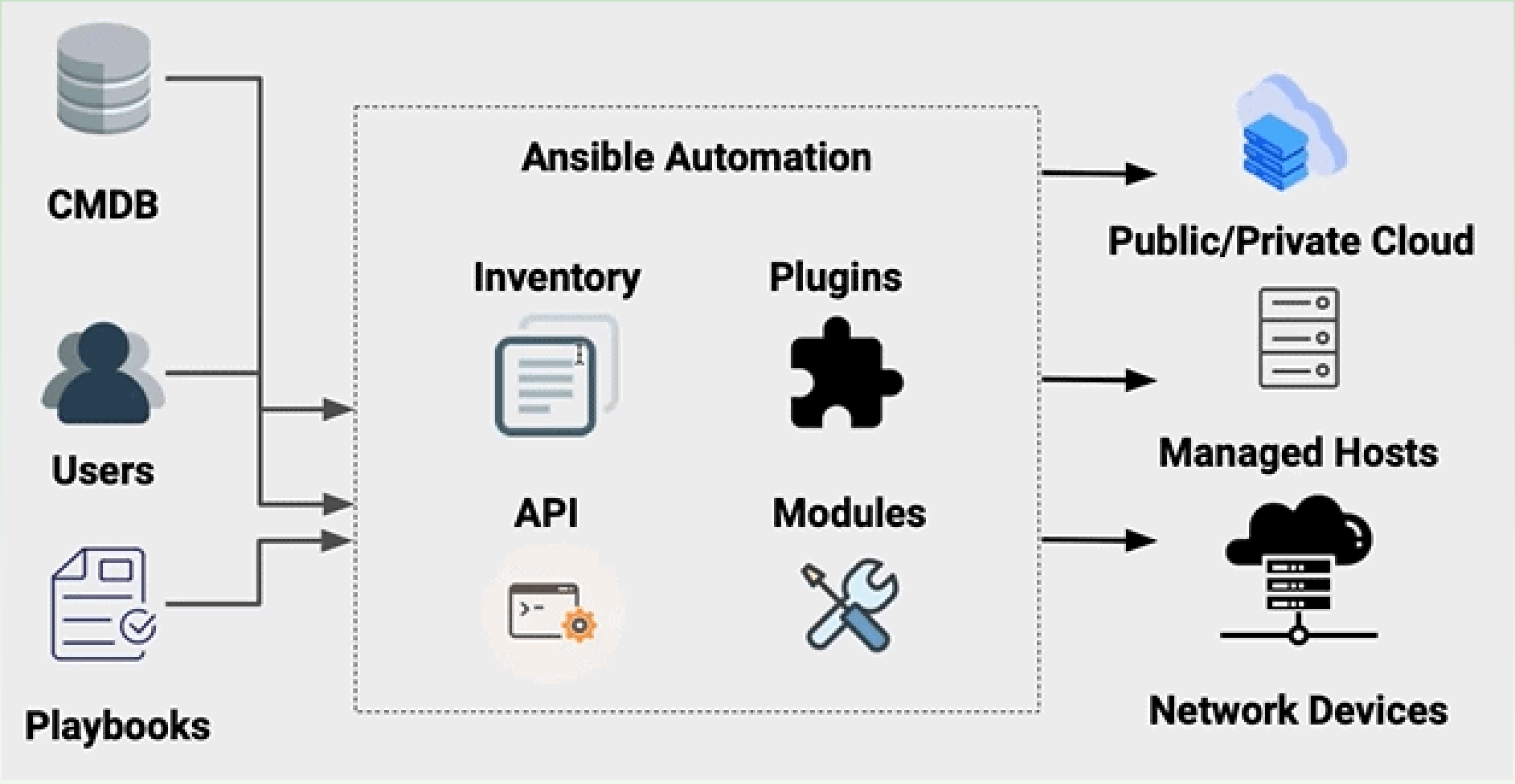
- Wyrażenia Jinja mogą się ewaluować do boola.
- Z użyciem `when` można pomijać część tasków.
- Więcej niż jeden `when` łączony jest z użyciem AND.
- Ewaluacja na hoście.
- By wykonać więcej niż jeden task pod `when` należy użyć `block` lub include

Demonstracja 3

Bardziej skomplikowany przykład: sortowanie rozproszone

Typy pluginów

Connection	Metody łączenia się ze zdalnymi hostami.
Inventory	Źródła opisu inwentarza.
Module	Rozszerza dostępne akcje w taskach.
Filter	Dodatkowe opcje przetwarzania danych w Jinja.
Lookup	Odczyt danych ze środowiska.
Strategy	Sposoby wykonania playbooka.
Become	Eskalacja uprawnień.



Źródło: https://k21academy.com/ansible/q-a-day1-live-session-review/

Demonstracja 4

Przykłady użycia pluginów w inwentarzu

include vs import

	Include_*	Import_*
Type of reuse	Dynamic	Static
When processed	At runtime, when encountered	Pre-processed during playbook parsing
Task or play	All includes are tasks	`import_playbook` cannot be a task
Task options	Apply only to include task itself	Apply to all child tasks in import
Calling from loops	Executed once for each loop item	Cannot be used in a loop
Using `--list-tags`	Tags within includes not listed	All tags appear with `--list-tags`
Using `--list-tasks`	Tasks within includes not listed	All tasks appear with `--list-tasks`
Notifying handlers	Cannot trigger handlers within includes	Can trigger individual imported handlers
Using `--start-at-task`	Cannot start at tasks within includes	Can start at imported tasks
Using inventory variables	Can `include_*: {{ inventory_var }}`	Cannot `import_*: {{ inventory_var }}`
With playbooks	No `include_playbook`	Can import full playbooks
With variables files	Can include variables files	Use `vars_files:` to import variables

Źródło:
`https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_reuse.html`

Inne konstrukty Ansible

- Handler i notify – pozwalają uzyskać obsługę podobną do wyjątków. Po zakończeniu ścieżki, jeśli został wywołany `notify` to zostanie *jednokrotnie* uruchomiony odpowiedni handler.
- `vars_promt` pozwala poprosić użytkownika o wpisanie danych wejściowych.
- `tags` służy do kategoryzowania zadań, tak by uruchomić tylko część z nich

Pułapki w które łatwo można wpaść

Zadania wykonywane na master węźle

Podstawianie, lookup, testy są zawsze wykonywane na master węźle (i w jego kontekście), a nie na hoście – ogranicza to liczbę zależności do zainstalowania na hoście.

Typy

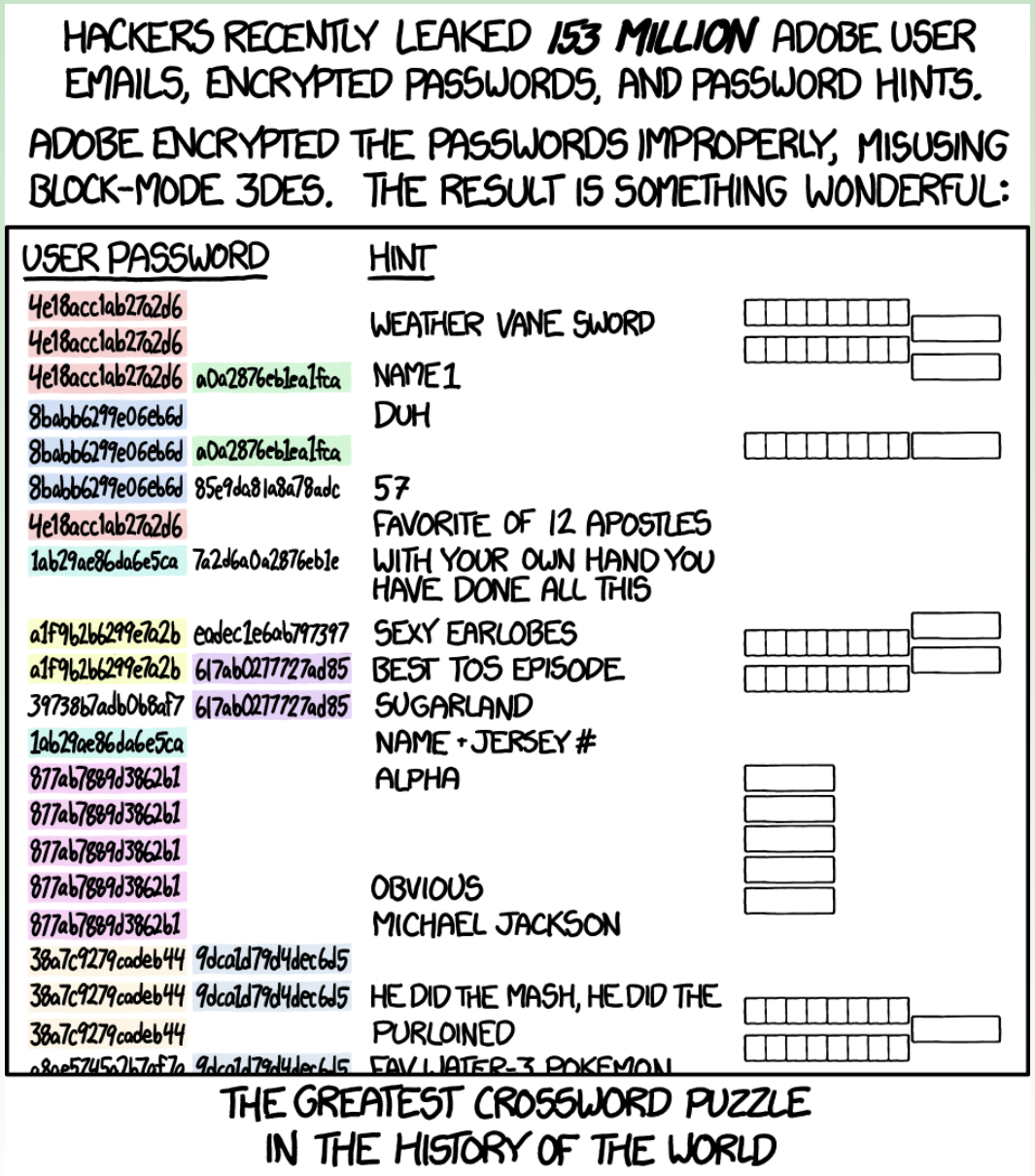
```
- name: check value of return code
  ansible.builtin.debug:
    var: bar\_status.rc

- name: check test for rc value as string
  ansible.builtin.debug:
    var: bar\_status.rc == "127"

- name: check test for rc value as integer
  ansible.builtin.debug:
    var: bar\_status.rc == 127
```

W jaki sposób bezpiecznie przechowywać hasła?

Ansible Vault:

- Szyfrowanie i deszyfrowanie wbudowanie w Ansible.
- Chroni tylko "data on rest".
- Hasło z pliku lub z terminala.
- `vault_id` używany jako wskazówka jakie hasło należy zastosować.

A jak przechowywać hasło do hasła?



Źródło: https://xkcd.com/1286/