Capitolo 2 – Architettura e Componenti di Ansible

Dopo aver compreso cos’è Ansible, è fondamentale capire come è strutturato e quali sono i suoi componenti principali.

Una corretta comprensione dell’architettura è essenziale per utilizzarlo in modo professionale in ambienti server.

2.1 Modello Architetturale

Ansible utilizza un modello **push-based**.

Questo significa che:

- Le configurazioni vengono inviate dal nodo di controllo
- I server remoti non eseguono processi agent in background
- Le operazioni vengono eseguite solo quando richiesto

Questo approccio riduce complessità, consumo risorse e superficie di attacco.

2.2 Control Node

Il **Control Node** è la macchina su cui è installato Ansible.

Caratteristiche:

- Da qui vengono eseguiti comandi e playbook
- Deve avere accesso SSH ai nodi gestiti
- Contiene inventory, playbook, ruoli e configurazioni

Requisiti:

- Linux o macOS (Windows supportato tramite WSL)
- Python installato
- Accesso SSH verso i nodi remoti

In ambienti enterprise il Control Node può essere:

- Una macchina amministrativa
- Un server dedicato all’automazione
- Un runner CI/CD

2.3 Managed Nodes

I **Managed Nodes** sono i server che vengono configurati da Ansible.

Requisiti minimi:

- SSH attivo
- Python installato (preferibilmente Python 3)
- Utente con privilegi sudo

Importante:
Non è richiesto alcun agente Ansible installato sul nodo remoto.

Questo è uno dei principali vantaggi rispetto ad altri strumenti di configuration management.

2.4 Inventory

L’**Inventory** è il file che definisce quali server devono essere gestiti.

Può essere:

- Statico (file ini o yaml)
- Dinamico (generato tramite script o integrazione cloud)

Esempio concettuale:

- Gruppo web
- Gruppo database
- Variabili specifiche per gruppo o host

L’inventory è il punto di partenza di qualsiasi esecuzione.

2.5 Moduli

I **Moduli** sono le unità operative di Ansible.

Sono piccoli programmi che:

- Installano pacchetti
- Gestiscono servizi
- Creano utenti
- Copiano file
- Modificano configurazioni

Quando eseguiamo un comando Ansible:

1. Il modulo viene copiato temporaneamente sul nodo remoto
2. Viene eseguito
3. Restituisce un output JSON
4. Viene rimosso

Questo processo è trasparente per l'amministratore.

2.6 Playbook

I **Playbook** sono file YAML che descrivono lo stato desiderato dell'infrastruttura.

Definiscono:

- Su quali host operare
- Quali task eseguire
- In quale ordine

Un playbook può:

- Installare pacchetti
- Configurare servizi
- Applicare template
- Riavviare demoni

Rappresentano il cuore dell'automazione con Ansible.

2.7 Idempotenza

Concetto fondamentale in ambito sistemistico.

Un’operazione è idempotente quando:

- Può essere eseguita più volte
- Produce sempre lo stesso risultato finale
- Non genera modifiche se lo stato è già corretto

Esempio:

- Se nginx è già installato → nessuna azione
- Se non è installato → viene installato

Questo garantisce:

- Sicurezza nelle riesecuzioni
- Stabilità in produzione
- Coerenza infrastrutturale

2.8 Flusso di Esecuzione

Quando si lancia un playbook:

1. Ansible carica il file di configurazione
2. Legge l’inventory
3. Stabilisce connessioni SSH
4. Esegue i task in ordine
5. Riporta lo stato finale (ok, changed, failed)

L’output fornisce:

- Numero host raggiunti
- Task modificati
- Eventuali errori

Conclusione

Ansible si basa su un’architettura semplice ma estremamente potente:

- Nessun agente
- Comunicazione SSH
- Stato dichiarativo
- Modularità

Comprendere questi elementi è fondamentale prima di passare alla configurazione operativa.

Nel prossimo capitolo entreremo nella configurazione iniziale dell’ambiente di lavoro: inventory, ansible.cfg e struttura progetto.

Revision #2
Created 2026-02-26 13:45:14 UTC by Pe
Updated 2026-02-26 13:52:02 UTC by Pe