

Capitolo 3 – Configurazione Iniziale dell'Ambiente di Lavoro

Dopo aver compreso architettura e componenti, il passo successivo è impostare correttamente l'ambiente operativo. In ambito sysadmin è fondamentale non lavorare in modo disordinato. Una struttura chiara permette scalabilità, manutenzione e integrazione con Git.

3.1 Non lavorare in /etc/ansible

Ansible può utilizzare:

- /etc/ansible/ansible.cfg
- /etc/ansible/hosts

Tuttavia, in ambienti professionali è sconsigliato lavorare direttamente lì.

Motivi:

- Difficoltà di versionamento
- Configurazione globale poco flessibile
- Problemi in ambienti multi-progetto

Best practice: creare una directory progetto dedicata.

3.2 Creazione Struttura Base Progetto

Esempio struttura iniziale:

```
mkdir -p ~/ansible/{inventory,playbooks,roles,group_vars,host_vars}
cd ~/ansible
```

Struttura consigliata:

```
ansible/
├─ ansible.cfg
├─ inventory/
│   ├─ production
│   ├─ staging
│   └─ dev
├─ playbooks/
├─ roles/
├─ group_vars/
└─ host_vars/
```

Descrizione:

- ansible.cfg → configurazione locale del progetto
- inventory/ → separazione ambienti
- playbooks/ → file operativi
- roles/ → componenti modulari
- group_vars/ → variabili per gruppi
- host_vars/ → variabili per host specifici

Questa struttura è scalabile e adatta a infrastrutture reali.

3.3 Creazione del File ansible.cfg Locale

Creare un file `ansible.cfg` nella root del progetto:

```
[defaults]
inventory = ./inventory/production
remote_user = ansible
host_key_checking = False
retry_files_enabled = False
```

```
roles_path = ./roles
forks = 10
timeout = 30

[privilege_escalation]
become = True
become_method = sudo
become_ask_pass = False
```

Spiegazione Parametri Principali

- **inventory**
Percorso dell'inventory di default.
- **remote_user**
Utente SSH utilizzato per la connessione.
- **host_key_checking**
Se disabilitato evita prompt interattivi (valutare in produzione).
- **retry_files_enabled**
Disabilita la creazione di file .retry.
- **roles_path**
Percorso locale dei ruoli.
- **forks**
Numero di host gestiti in parallelo.
- **become**
Abilita automaticamente l'escalation privilegi.

3.4 Ordine di Precedenza del File di Configurazione

Ansible legge il file di configurazione in questo ordine:

1. Variabile ambiente `ANSIBLE_CONFIG`
2. `ansible.cfg` nella directory corrente
3. `~/ansible.cfg`
4. `/etc/ansible/ansible.cfg`

Questo significa che il file locale nel progetto ha priorità rispetto a quello globale.

Verifica configurazione attiva:

```
ansible-config view
```

3.5 Configurazione SSH Corretta

Prerequisito fondamentale: accesso SSH senza password.

Generazione chiave (se non esiste):

```
ssh-keygen -t ed25519
```

Copia chiave sui nodi remoti:

```
ssh-copy-id ansible@server
```

Test connessione:

```
ssh ansible@server
```

Best practice:

- Non usare root diretto
 - Creare utente dedicato (es. ansible)
 - Consentire sudo controllato
-

3.6 Verifica Connessione con Ansible

Prima di scrivere playbook, verificare la connettività.

Esempio:

```
ansible all -m ping
```

Output atteso:

- SUCCESS
- Nessun errore SSH
- Nessun errore Python

Se compare errore Python, verificare presenza di python3 sul nodo remoto.

3.7 Preparazione per Ambienti Multipli

Separare ambienti evita errori critici.

Esempio:

```
inventory/  
├─ production  
├─ staging  
└─ dev
```

Esecuzione su staging:

```
ansible-playbook -i inventory/staging playbooks/site.yml
```

Non mescolare mai produzione e ambienti di test nello stesso inventory.

Conclusione

Una corretta configurazione iniziale è fondamentale per:

- Scalabilità
- Sicurezza
- Manutenibilità
- Integrazione con controllo versione

Nel prossimo capitolo entreremo nel dettaglio dell'Inventory: sintassi, gruppi, variabili e struttura professionale.

Revision #2

Created 2026-02-26 13:48:14 UTC by Pe

Updated 2026-02-26 13:51:37 UTC by Pe