Automazione del deploy di un cluster Ceph – Relazione di Tirocinio

Giacomo Biagioni
Matricola: 0001078126

5 Settembre 2025

Contents

Strumenti e tecnologie utilizzate

1.1 Multipass

Multipass è uno strumento sviluppato da Canonical che consente di creare e gestire macchine virtuali direttamente da linea di comando. È stato scelto per la sua compatibilità sia con sistemi Linux che Windows.

1.2 Microceph

MicroCeph è una distribuzione semplificata di Ceph sviluppata da Canonical, installabile tramite pacchetto `snap`. Il suo obiettivo è fornire un sistema di storage distribuito pronto all'uso, riducendo al minimo la complessità tipica dell'installazione di Ceph.

1.3 Samba

Samba è un software open source che implementa il protocollo SMB/CIFS, consentendo la condivisione di file e risorse di rete tra sistemi Linux/Unix e Windows. Grazie a Samba, è possibile accedere a directory e filesystem remoti come se fossero cartelle locali, garantendo compatibilità multipiattaforma.

1.4 Strumenti di supporto

In aggiunta agli strumenti principali, nel progetto sono stati utilizzati anche alcuni strumenti di supporto che hanno semplificato la configurazione delle macchine virtuali e del cluster.

1.4.1 Netplan: configurazione della rete e indirizzi IP statici

Netplan è un sistema di configurazione della rete per Ubuntu che utilizza file YAML per definire la configurazione di interfacce di rete. È stato utilizzato per gestire facilmente gli indirizzi IP statici, evitando problemi legati al cambio di IP dinamici (ad esempio dopo un riavvio), che avrebbero potuto compromettere il funzionamento del cluster microceph.

1.4.2 Cloud-init: automazione della configurazione iniziale delle VM

Cloud-init è uno strumento utilizzato per automatizzare la configurazione di sistemi al primo avvio. Permette di eseguire script, installare pacchetti, creare utenti e configurare servizi, senza dover intervenire manualmente sulla macchina.

Architettura

2.1 Struttura del codice

Il progetto è organizzato in moduli Python separati, ognuno con responsabilità specifiche:

- `main.py`: punto di ingresso principale, gestisce l'interfaccia a linea di comando e l'orchestrazione delle operazioni.
- `cluster_manager.py`: gestisce la creazione, configurazione e gestione del cluster Ceph.
- `multipass_manager.py`: interagisce con l'API di Multipass per la gestione delle VM.
- `hypervisor_check.py`: verifica la compatibilità e lo stato dell'hypervisor (KVM/QEMU su Linux, Hyper-V su Windows).
- `configs.py`: contiene le configurazioni predefinite e le impostazioni di default.
- `cloud-init`: script di inizializzazione per le VM, configurano i nodi Ceph all'avvio.

2.2 Workflow

Il flusso operativo del sistema segue questi passaggi principali:

1. **Verifica dell'ambiente**: `hypervisor_check.py` controlla la presenza e lo stato dell'hypervisor.
2. **Creazione delle VM**: `multipass_manager.py` avvia le VM utilizzando Multipass, configurando CPU, RAM e disco secondo le specifiche.
3. **Configurazione iniziale**: le VM eseguono gli script `cloud-init` per installare e configurare i pacchetti necessari. In questa fase viene anche configurata la rete, modificando il file `netplan` per assegnare IP statici ai nodi del cluster.
4. **Setup degli OSD**: `cluster_manager.py` configura i nodi Ceph per i componenti OSD (Object Storage Daemon), predisponendo i dischi e registrando gli OSD nel cluster. Per semplicità, sono stati utilizzati file di loopback come dischi virtuali.
5. **Configurazione del filesystem**: una volta che Ceph è attivo, viene configurato il filesystem distribuito.
6. **Accesso al filesystem**: i client possono montare il filesystem tramite Samba, a seconda della configurazione.

Utilizzo

3.1 Requisiti

Per poter eseguire correttamente il sistema, è necessario predisporre l'ambiente con i seguenti requisiti:

Requisito	**Dettagli**
Sistema operativo	Qualsiasi sistema compatibile con Multipass
Versione Python	3.8 o superiore, con i moduli standard necessari al progetto
Multipass	Installato e funzionante, con supporto alla virtualizzazione abilitato
Risorse per le VM	Quelle impostate come configurazione di default nel programma

Risorse utilizzate nella fase di testing

Parametro	**Valore utilizzato nei test**
Sistema operativo	Windows
Versione Python	3.13.7
Versione Multipass	1.16.0+win
Cluster di base	Composto da 2 nodi
Specifiche dei nodi	2 CPU, 2 GB RAM, 10 GB disco, Ubuntu 22.04

Esecuzione dello script

Per eseguire il deploy del cluster, è necessario lanciare lo script principale `main.py` con il comando:

```
python main.py deploy
```

Lo script supporta diverse opzioni configurabili da linea di comando. Di seguito sono riportate le principali opzioni disponibili:

- `--nodes`: Numero di nodi da creare nel cluster. Valore predefinito: 2.
- `--base-name`: Nome base per i nodi. Valore predefinito: `ceph-node`.
- `--cpus`: Numero di CPU per ciascun nodo. Valore predefinito: 2.
- `--ram`: Quantità di RAM per nodo. Valore predefinito: 2G.
- `--disk`: Dimensione del disco per nodo. Valore predefinito: 10G.
- `--os`: Versione del sistema operativo per le VM. Valore predefinito: 22.04.
- `--default`: Usa la configurazione di default senza richiedere parametri interattivi.

- `--with-client`: Crea anche una VM client alla fine del deploy.
- `--debug`: Abilita output dettagliato per il debug.

Esempi di comandi:

```
Deploy interattivo con richiesta parametri:
python main.py deploy

Deploy con valori di default:
python main.py deploy --nodes 5 --ram 4G

Deploy con parametri personalizzati:
python main.py deploy --nodes 5 --cpus 4 --ram 4G --disk 20G --with-client

Utilizzato in fase di testing:
python main.py deploy --default --with-client --debug
```

3.2 Accesso al filesystem

Una volta completato il deploy del cluster, il filesystem distribuito può essere montato e utilizzato dalle macchine client. La procedura varia a seconda del sistema operativo.

Linux

Su Linux, il metodo consigliato è tramite il supporto nativo del kernel, che garantisce le migliori prestazioni. La procedura generale è la seguente:

1. Assicurarsi che samba sia installato.
2. Configurare Samba per consentire l'accesso al filesystem dalla macchina client. Il blocco di configurazione seguente definisce lo share e i permessi necessari:

   ```
   Config usato:
   [<nome_dello_share>]
   path = <mount_point>
   browseable = yes
   read only = no
   valid users = <samba_username>
   create mask = 0755
   directory mask = 0755
   ```

3. Creare un punto di mount locale, ad esempio:

   ```
   sudo mkdir -p /mnt/cephfs
   ```

4. Montare il filesystem CephFS tramite il kernel:

```
sudo mount -t cifs //<IP_NODO_PRIMARIO>/<nome_dello_share> \
<mount_point> -o username=<username>,password=<password>
```

Nota: È possibile montare CephFS anche utilizzando `ceph-fuse`, un client in user space. Tuttavia, questa modalità è generalmente meno performante rispetto al montaggio tramite kernel ed è stata pertanto evitata nel progetto.

Windows

Grazie a Samba pure da windows possiamo connetterci facilmente

1. Assicurarsi che il client Linux del cluster stia condividendo il filesystem via Samba.
2. Sul PC Windows, aprire `Esplora Risorse` e selezionare "Connetti unità di rete".
3. Inserire l'indirizzo della condivisione Samba, ad esempio:

```
\\<IP_NODO_PRIMARIO>\<nome_dello_share>
```

4. Assegnare una lettera di unità e completare la connessione.

In questo modo, i client Windows possono leggere e scrivere su CephFS come se fosse un normale filesystem di rete, pur non avendo supporto diretto per CephFS.

Miglioramenti possibili

4.1 Salvataggio della configurazione corrente del cluster

Attualmente, lo script applica direttamente la configurazione del cluster e dei client senza mantenere una copia della configurazione precedente. Un miglioramento possibile sarebbe introdurre un meccanismo di *salvataggio della configurazione attuale* prima di applicare eventuali modifiche.

In questo modo, in caso di problemi durante l'applicazione della nuova configurazione, sarebbe possibile ripristinare rapidamente lo stato precedente, riducendo i tempi di intervento e minimizzando il rischio di interruzioni del servizio. Questo file di salvataggio potrebbe contenere informazioni come:

- Parametri delle VM (CPU, RAM, disco, OS)
- Configurazione del cluster Ceph (ruoli dei nodi, versioni dei servizi)
- Configurazioni delle condivisioni Samba

4.2 Configurazione della rete per l'accesso esterno al filesystem

Attualmente, tutte le VM client che accedono al filesystem fanno parte della stessa rete delle VM del cluster, e lo script non gestisce configurazioni di rete aggiuntive. Un possibile miglioramento sarebbe prevedere la configurazione di reti virtuali o bridge che permettano l'accesso al filesystem anche da host esterni o da subnet differenti.

Ciò potrebbe includere:

- Definizione di network bridge o NAT per le VM
- Configurazioni firewall per permettere il traffico verso i servizi Ceph/Samba
- Meccanismi di sicurezza aggiuntivi per l'accesso remoto

Questo permetterebbe di rendere il cluster più flessibile e fruibile in scenari reali in cui i client non si trovano sulla stessa rete fisica o virtuale delle VM.