PIB: Una soluzione alla complessita' di installazione dei cluster moderni

Alberto Damo
Dipartimento di Matematica "Tullio Levi-Civita"
Universita' di Padova
Padova, Italia,
alberto.damo@studenti.unipd.it

Alessandro Pirolo
Dipartimento di Matematica "Tullio Levi-Civita"
Universita' di Padova
Padova, Italia,
alessandro.pirolo@studenti.unipd.it

August 29, 2024

Contents

1 Preambolo

Il mondo di oggi e' estremamente interconnesso e complesso, per questa ragione oramai ogni applicativo pensato per il grande pubblico deve essere in grado di garantire l'affidabilita' del servizio in ogni momento. Per risolvere questi problemi vengono spesso usati dei sistemi distribuiti in rete, detti anche **cluster**, che permettono di garantire la scalabilita' e l'affidabilita' del serverzio in ogni momento. Questi ultimi pero' sono estremamente complessi e, soprattutto, difficili da installare. Per rendersene conto e' sufficente seguire la procedura di configurazione di un cluster **Kubernetes** o **Ceph**. In entrambi i casi la procedura e' delicata e richiede molto tempo per essere conclusa a meno che non si utilizzino strumenti esterni per facilitare il lavoro, per quanto non siano sempre affidabili. Tale complessita' risiede, tra i tanti fattori, nell'architettura scelta per modellare il sistema, la quale, solitamente, e' di tipo **Orchestrator**. Tale modello offre numerosi vantaggi a un costo pero': i nodi che compongono il cluster non sono omogenei. Tale dettaglio, per quanto sembri insignificante, e' cio' che rende complessa la procedura di configurazione, aggiornamento e mantenimento del cluster. In questo paper tratteremo di come abbiamo creato un cluster che garantisca le richieste di affidabilita' necessarie permettendo inoltre una veloce e semplice installazione/configurazione.

2 Composizione di un cluster

Prima di poter analizzare quali siano i nostri obbiettivi, come li abbiamo raggiunti e del perche' siano state fatte determinate scelte e' necessario descrivere gli elementi che compongono un cluster e le loro caratteristiche:

- **nodi**: sono dei computer che si occupano di svolgere la computazione necessaria per mantenere il servizio attivo. Questi elementi possono essere fisici (servers dedicati) o virtuali (virtual machines, containers).
- **gestore dei nodi**: questo componente si occupa dell'aggiunta, rimozione e sostituzione dei nodi nel cluster. Puo' essere un **hypervisor**, nel caso i nodi siamo virtuali, o un **operatore** nel caso i nodi siano reali server. E' probabile che ci siano piu' gestori, in particolar modo se i nodi sono distanti tra di loro da un punto di vista geografico.
- **gestore di rete**: questo componente si occupa della gestione delle connessioni interne ed esterne al cluster. In particolar modo si occupa dell'assegnazione degli indirizzi di rete ai nodi (e' possibile che ogni nodo abbia piu' interfaccie di rete e quindi piu' indirizzi IP)
- **sistema operativo**: nonostante appartenga al nodo il sistema operativo deve essere progettato su misura per il cluster in quanto anche questo componente e' responsabile per la corretta operativita' dell'intera infrastruttura. In particolar modo dovra':
  - ottenere gli indirizzi di rete (e' possibile siano piu' di uno per ogni node)
  - eseguire una diagnostica interna sullo stato della macchina
  - eseguire eventuali programmi o daemon responsabili delle funzionalita' del cluster e degli applicativi

3 Obbiettivi

Come gia' accennato il nostro obbiettivo e' di realizzare un cluster che sia allo stesso tempo **affidabile** e **semplice da configurare/installare**.

In particolare il sistema dovra':

- permettere la **scalabilita'** del servizio aumentando o diminuendo il numero di nodi senza interrompere i servizi forniti.
- garantire la **disponibilita'** del servizio anche in caso di guasti sia a livello fisico (servers, rete), sia a livello software (crush del sistema operativo o dell'applicativo stesso).
- integrare dei tool per l'**interazione** dell'amministratore con il cluster.
- essere privo di **single point of failure**
- avere tutti i nodi del cluster uguali tra di loro a livello strutturale

4 Soluzioni

5 PIB

5.1 elementi

5.1.1 nodi

5.1.2 gestore dei nodi e della rete

5.1.3 sistema operativo

5.2 RAFT per il consenso distribuito

5.2.1 motivazioni

5.2.2 limiti

5.3 dettagli implementativi dell'applicativo

5.3.1 obbiettivi

5.3.2 struttura del codice

5.3.3 funzionamento

5.3.4 limiti

6 tool di configurazione

6.1 obbiettivi

6.2 utilizzo

6.3 limiti

7 valutazioni

7.1 raggiugimento obbiettivi

7.2 difficolta' riscontrate

7.2.1 alto livello di concorrenza e distribuzione

7.2.2 modularizzazione

7.2.3 implementazione del protocollo raft

7.2.4 organizzazione dei test

7.2.5 raccolta' dei log

7.2.6 automatizzare il processo di deployment (in produzione) del cluster

7.2.7 comprensione del cambio di configurazione del protocollo raft

7.2.8 riconoscimento dei casi di race condition