**ctrlNods** e **ctrlClus**Nodes Control e Clusters Control

***2025 04 04 ver.1.20***

**• ORIGIN OF THE IDEA - CASE STUDY**

**• WHAT IT IS ABOUT**

**• PURPOSE**

**• DETAILED ARCHITECTURE**

**ctrlNods** e **ctrlClus  
Nodes Control e Clusters Control**

**ORIGIN OF THE IDEA - CASE STUDY**Nel 2022, su 2 datacenter di un cliente in ambito bancario.

Dopo 1 mese di continui allarmi di "simultanei down di nodi cassandra".  
  
  
**ctrlNods**   
Ho sviluppato un'architettura applicativa modulare per il monitoraggio dei nodi di cluster db**.** Messa in uso sui nodi di alcuni cluster cassandra di questo cliente.

*La causa scatenante l'allarme è stata individuata al primo evento successivo alla messa in uso di questo sistema di controllo.****QUAL'ERA IL PROBLEMA?****Di sicuro non era un problema di:*

1. *Settaggio di parametri dei nodi o del cluster cassandra, quindi del DB*
2. *Memoria*
3. *Network*
4. *Query*
5. *Funzionamento del Cluster*  
   *Dopo che trovi la causa capisci che la soluzione per trovarla è banale.*

**ctrlClus**

1. Creare delle insert select che corrispondono alle indagini per periodo
2. Impaginare i risultati di una 10na di query sql e delle query al punto 1.
3. Notificare i risultati degli eventi elaborati al punto 1

*Quanto descritto di colore grigio è da siluppare.*

**ctrlNods  
Nodes Control**

**WHAT IT IS ABOUT**   
A Bash script application installed on each node of the Cassandra cluster  
  
**PURPOSE**

1. Performs **CHECKS**

2. **LOGS** state changes

3. **SAVES events** to a local database (SQLite)

4. **SYNCHRONIZES** data to a central database for ctrlClus

5. **SENDS ALERTS** via Teams chat / email / SMS / email depending on the event

1. **I MODULI DI CONTROLLO:**
   1. **generic**
      * 1. **S\_DISK.sh**
        2. **S\_NMAP.sh**
        3. **S\_PING.sh**
   2. **cassandra**
      * 1. **S\_Balancing.sh**
        2. **S\_CL\_ClusterState.sh**
        3. **S\_CL\_QueryLatency.sh**
        4. **S\_CPU.sh**
        5. **S\_MEM.sh**
        6. **S\_QueryQueue.sh**
   3. **test**
      * 1. **S\_Partition.sh**
2. **LOGS WHAT**State changes:

* **ON/OFF**
* **AVAIABLE/UNAVAIABLE**
* **DOWN/UP**
* **UNDER\_100/UPPER\_100**
* **etc...**

**WHAT IT DOES WITH THE LOGS:**

* a) **Saves events to 2 files** (`data.log` and `data.sqlite`) located on a portion of disk in the RAM memory (ramdisk) of each client (mounted as tmpfs).
* b) Periodically synchronizes the **saved events to a central database** (MySQL, REDIS).

**ctrlNods  
architecture**

|  |  |
| --- | --- |
| **SETUP - LOG FILE HISTORY - SYNC** | |
| **00\_POSTreboot.sh** | ramdisk setup |
| **01\_updatefs.sh** | Log file history |
| 02\_syncDB.sh | Sync Local DB vs Centralized DB |

|  |  |
| --- | --- |
| **APPLICATION** | |
| **M\_chk.sh** | Application Launch |
| **M\_config.sh** | Settings |
| **M\_control.sh** | Application Logic |

|  |  |
| --- | --- |
| **CONTROL MODULES** | |
| **/modules/generic S\_\*** | Generic Modules |
| **/modules/cassandra/S\_\*** | Cassandra Modules |
| **/modules/test S\_\*** | Test Modules |

|  |  |
| --- | --- |
| **LOG FILE** | |
| **/log/**data.log | eventi in formato txt |
| **/log/**data.sqlite | eventi su db |
| UP\_$module\_$nod.ok (flag servizio UP) | flag file servizio up |

|  |
| --- |
| **BINARY FILE** |
| **/bin/\*** dd ping nmap sqlite3 sqldiff SchemaDB.sql *Script creazione DB locale (SQLite)* |

**ctrlNods  
generic modules**

*checks to localhost*

|  |  |
| --- | --- |
| **S\_DISK.sh** | |
| **PURPOSE** | I/O Disk |
| **CONTROLS** | */dev/zero of=/opt/ctrlNods/testfile\_100MB bs=1024 count=402400* |
|  | Se la velocità disco diventa > xx |

|  |  |
| --- | --- |
| **S\_CPU.sh** | |
| **PURPOSE** | stress java CPU |
| **CONTROLS** | *top -b -n 1 | grep java | awk '{print $9}'* |
|  | *Se la CPU supera xx%, il nodo potrebbe essere sovraccarico* |

*to checks against the other nodes of the cluster*

|  |  |
| --- | --- |
| **S\_PING.sh** | |
| **PURPOSE** | Ping STATUS |
| **CONTROLS** | Tutti i nodi fanno un ping di tutti i nodi |

|  |  |
| --- | --- |
| **S\_NMAP.sh** | |
| **PURPOSE** | Test service |
| **CONTROLS** | *7001 7199 9142* |
|  | 7001 TCP Comunicazione tra nodi con cifratura SSL abilitata  7199 TCP JMX (Java Management Extensions) per il monitoraggio e la gestione di Cassandra  9142 TCP Porta CQL per connessioni SSL/TLS |

**ctrlNods  
cassandra modules**

*checks to localhost*

|  |  |
| --- | --- |
| **S\_HINTS.sh** | |
| **PURPOSE** | Verifica la presenza di Hints files |
| **CONTROLS** | Data creazione di quale nodo per quale KEYSPACE per quale TABLES |

*to checks against the cluster nodes*

|  |  |
| --- | --- |
| **S\_QueryQueue.sh** | |
| **PURPOSE** | Thread pool delle query |
| **CONTROLS** | *nodetool tpstats | awk 'NR>1 && ($3+0 > 0 || $5+0 > 0) {print "Attenzione:", $1, "Pending:", $3, "Blocked:", $5}'* |
|  | *Pending → Query in attesa di essere elaborate Blocked→ Query bloccate indica un forte carico del nodo* |

|  |  |
| --- | --- |
| **S\_Balancing.sh** | |
| **PURPOSE** | Streaming e bilanciamento del cluster |
| **CONTROLS** | *nodetool netstats | grep -E 'Receiving from|Sending to'* |
|  | Controlla lo stato del trasferimento dati tra i nodi  Se vedi \*\*"Receiving from"\*\* o \*\*"Sending to"\*\*, significa che un nodo sta ricevendo o inviando dati.  Se lo streaming è troppo lungo, il nodo potrebbe essere sovraccarico. |

|  |  |
| --- | --- |
| **S\_CL\_ClusterState.sh** | |
| **PURPOSE** | Cluster State |
| **CONTROLS** | *nodetool status | grep -E '^D|^UJ|^UM|^UL'* |
|  | *D → Down (nodo non raggiungibile)*  *UJ → Joining (nodo si sta unendo, ma se persiste è un problema)*  *UM → Moving (nodo sta spostando i dati, verifica se dura troppo)*  *UL → Leaving (nodo sta lasciando il cluster, ma deve completare il processo)* |

|  |  |
| --- | --- |
| **S\_CL\_QueryLatency.sh** | |
| **PURPOSE** | Query Latency |
| **CONTROLS** | *nodetool proxyhistograms* |
|  | *Lo script definisce delle soglie massime per le latenze di lettura, scrittura e range. Se un valore supera questa soglia, viene segnalato come criticità.*  *#Estrazione dei valori: Lo script esegue il comando nodetool proxyhistograms e analizza l’output utilizzando grep e awk per estrarre i valori di latenza dei percentili 50%, 75%, 95% e 99% per ciascuna metrica di latenza.*  *#Funzione check\_criticality: La funzione confronta il valore di latenza con la soglia definita. Se il valore supera la soglia, stampa un messaggio di avviso.*  *#Controllo minimi e massimi: Lo script può anche verificare i valori minimi e massimi per ogni metrica di latenza, segnalando eventuali anomalie nei picchi di latenza.* |

|  |  |
| --- | --- |
| **S\_Partition.sh** | |
| **PURPOSE** | Large Partitions |
| **CONTROLS** | *nodetool tablehistograms keyspace table* |
|  | Cicla per tutte le tabelle di tutti i keyspaces Controlla la dimensione delle partizioni  Se ci sono partizioni molto grandi (> XXMB), possono rallentare le query. |

|  |  |
| --- | --- |
| **S\_MEM.sh** | |
| **PURPOSE** | Memoria e Garbage Collection |
| **CONTROLS** | *nodetool gcstats* |
|  | Controlla la frequenza del Garbage Collector  Min` / `Max` / `Mean`\*\* → Durata delle operazioni di GC  Count`\*\* → Numero totale di GC eseguiti  Se il \*\*tempo massimo del GC\*\* è elevato (> 1 sec), Cassandra potrebbe avere problemi di memoria |

**ctrlClus  
Clusters Control**

**DI COSA SI TRATTA**  
Un applicativo in python script installato su un server visibile da tutti i nodi del cluster.   
  
**SCOPO:**

Utilizza il DB CENTRALE

1. **ELABORA** i dati del database centrale che viene alimentato periodicamente da ciascun nodo di ciascun cluster.
2. **NOTIFICA** via chat teams/email/sms/email l'elaborazione degli allarmi ricevuti in un lasso di tempo.
3. **VISUALIZZA** i risultati delle domande che gli vengono chieste

* **Indisponibilità dei nodi:**  
  E caduto un nodo?
* Quando?
* Quante volte è successa la stessa cosa a quel nodo e agli altri?
* Perchè è successo? (rete, disco, ram, query, datacenter, computer, cpu)
* In quel momento altri nodi erano down?
* I settaggi sono corretti?