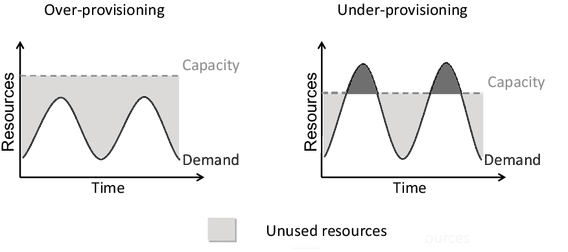
SDCC Progetto A1 - JDSys

Sistema di Storage Distribuito di tipo Chiave Valore

Danilo Dell’Orco  
 Facoltà di Ingegneria  
 Università di Roma Tor Vergata  
 Roma, Lazio, Italia  
 danilodellorcolp@gmail.com

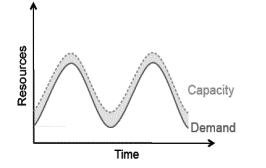
Jacopo Fabi  
 Facoltà di Ingegneria  
 Università di Roma Tor Vergata  
 Roma, Lazio, Italia  
 jacopo.fabi.1997@gmail.com

ABSTRACT

JDSys è un sistema di storage distribuito di tipo Chiave valore. Il sistema è distribuito nel senso che i dati salvati sono distribuiti su molteplici server, fornendo quindi un servizio ad alta disponibilità ed affidabilità. JDSys mira a funzionare su un'infrastruttura composta da diversi nodi, il cui numero preciso viene selezionato in modo da avere una distribuzione equa del carico di lavoro. Per questo il nostro sistema utilizza i servizi AWS, in modo da avere a disposizione una scalabilità teoricamente infinita. JDSys fornisce anche un’integrazione con il cloud storage S3 per aumentare anche la scalabilità al numero di dati. In particolare vengono migrate sul cloud tutte le entry scarsamente accedute, in modo tale da liberare spazio per entry più utilizzate. Oltre ad essere distribuiti, i dati vengono anche replicati tra i diversi server; questo garantisce una migliore scalabilità al numero di utenti e di dati.

1. INTRODUZIONE

Nel corso degli anni il traffico internet è cresciuto a dismisura. Ciò significa che un qualsiasi sistema che riceve richieste tramite la rete deve gestire una quantità di dati e di operazioni sempre maggiori. In un simile scenario è importante mantenere lo stesso livello di prestazioni ed affidabilità, a prescindere dal carico di lavoro in ingresso. Gli approcci tradizionali consistono nello di scalare verticalmente l’hardware dei server. Tuttavia questo approccio ha un costo esponenziale, ed ha quindi un upper bound al livello di prestazioni raggiungibili. Un altro problema da affrontare è quello del dimensionamento della propria infrastruttura, ovvero quante risorse allocare per fornire il servizio. In un approccio classico si potranno allocare soltanto un numero fisso di risorse durante l’esecuzione del servizio. Ciò vuol dire che, a fronte di un numero di richieste variabile, si potranno avere due scenari. Nell’approccio di under provisioning si allocano un numero di risorse che non riescono a far fronte a picchi di carico. Quindi si gestisce normalmente un carico medio, ma durante I picchi il servizio non sarà disponibilile. Nell’ over provisioning invece si forniscono un numero grande di risorse, in modo da poter gestire anche i picchi di carico. Questo garantisce che il sistema sia sempre disponibile, ma ci sarà anche un numero importante di sprechi in quanto bisognerà avere un'infrastruttura robusta anche durante il traffico medio.

Sfruttando invece un’infrastruttura cloud, è possibile offrire un servizio elastico, che permette di allocare risorse in base alla quantità di traffico che bisogna gestire. Questo approccio permette di minimizzare i costi, offrendo al contempo un servizio sempre disponibile.

L’altro vantaggio offerto dall’utilizzo dell’infrastruttura cloud è quello della scalabilità, poiché si può scalare teoricamente fino ad un numero infinito di macchine sfruttando la scalabilità orizzontale anziché quella verticale.

JDSys è un servizio di storage distribuito di tipo chiave-valore. Questo può ricevere richieste concorrenti, ed un traffico variabile. Per i motivi precedentemente descritti, JDSys utilizza Amazon Web Services a diversi livelli. Le istanze EC2 hostano i singoli data server, il Load Balancer garantisce una suddivisione equa delle richieste, mentre L’Auto Scaling permette di avere sempre il corretto numero di risorse attive in base al traffico in ingresso al sistema. Per garantire una ricerca efficiente della chiave all’interno del sistema, i diversi server sono organizzati in una overlay network ad anello (chord). Inoltre, per avere una maggirore scalabilità e disponibilità, i nodi implementano delle tecniche per la replicazione e riconciliazione dei dati.

Questo documento è strutturato come segue. La sezione 2 descrive il modello di dati utilizzato. La sezione 3 presenta la panoramica delle API client. La sezione 4 presenta la struttura e la progettazione del sistema. La sezione 4 descrive i test effettuati su JDSys. La sezione 5 conclude il documento, con le considerazioni finali sul sistema.

2. DATA MODEL

JDSys basa lo storage sul concetto di entry. Una entry è coppia chiave-valore. La Chiave rappresenta un identificativo univoco per l’entry, mentre possono esserci anche stessi valori associati a chiavi diverse. La chiave è rappresentata tramite una stringa, mentre il valore è un array di stringhe in quanto può contenere un numero variabile di argomenti.

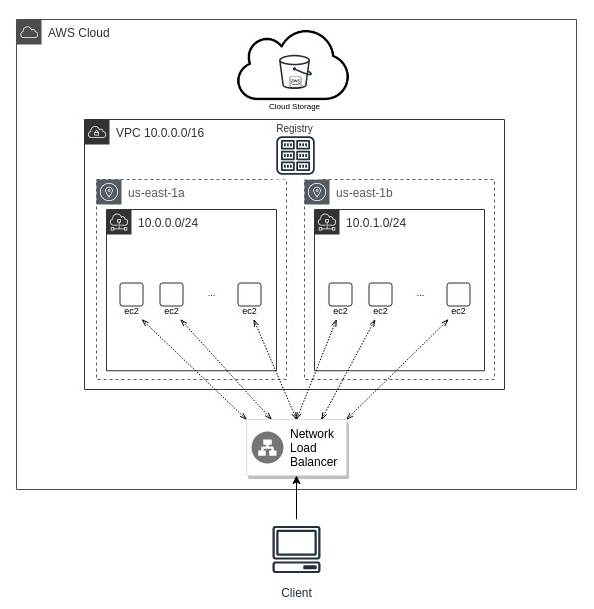
4. ARCHITETTURA DI SISTEMA

Un sistema di storage distribuito, oltre alla persistenza dei dati, deve offrire anche soluzioni scalabili e robuste per il bilanciamento del carico, il provisioning delle risorse, la sincronizzazione delle repliche, la gestione del sovraccarico, il trasferimento dello stato e la concorrenza, l'assegnazione delle richieste e l'instradamento delle richieste, il monitoraggio e gli allarmi del sistema. JDSys, fornisce una soluzione ad ognuno di questi aspetti, ed in questa sezione andiamo quindi a descrive nel dettaglio come sono stati affrontati.

Parleremo quindi in particolare della topologia di rete utilizzata, dei meccanismi di comunicazione, dei protocolli utilizzati, e per ognuno di questi punti verranno quindi descritti i principali aspetti implementativi.

4.1 Infrastruttura di Rete

L’intera infrastruttura di rete è stata definita sfruttando Amazon Web Service. Ciò ha permesso di avere un sistema distribuito I cui nodi dell’applicazione sono hostati sul cloud.

La Figura 1 mostra l’infrastruttura di rete di JDSys. I nodi del sistema sono hostati su delle istanze Amazon EC2 micro, che forniscono delle limitate capacità computazionali e di storage.

Risulta fondamentale in questo senso distribuire equamente il carico di lavoro tra questi nodi. A tale scopo, si utilizza l’ Elastic Load Balancer di AWS, che permette di distribuire le richieste in ingresso verso le istanze registrate nel Target Group. Un client dovrà quindi contattare il load balancer tramite il suo URL DNS, e la sua richiesta verrà redirezionata in automatico su uno dei nodi del Ring JDSys. Figura 1: Infrastruttura di Rete **JDSys**

Per motivi di sicurezza ed efficienza, tutte le istanze EC2 utilizzate sono state istanziate all’interno di un Virtual Private Cloud. In questo modo i nodi possono essere contattati dal Load Balancer, e possono comunicare tra loro tramite indirizzi IP privati, ma non potranno essere contattati direttamente dall’esterno.

Per simulare in maniera più fedele una latenza di rete, i nodi vengono istanziati in due zone di disponibilità differenti, che descrivono quindi due sottoreti IP differenti (10.0.0.0/24 e 10.0.0.1/24). Il service registry si trova anch’esso all’interno della VPC. A questa istanza è stato assegnato un IP Elastico, in modo che tutti i nodi possano contattarlo con successo.

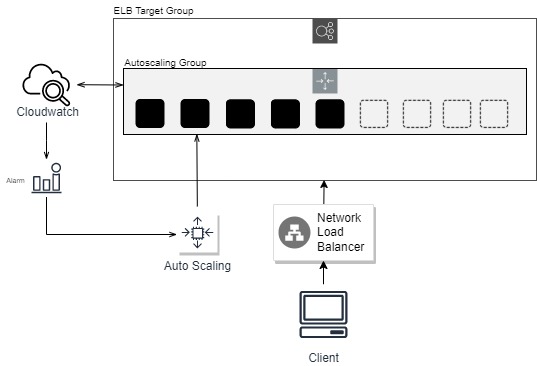
Fuori dal VPC troviamo invece un Bucket S3, che fornisce lo storage Cloud necessario per una maggiore scalabilità ai dati. I nodi interagiscono con S3 utilizzando le SDK di AWS, sfruttando quindi la rete internet pubblica in questo caso.

4.2 Autoscaling & Cloudwatch

Un aspetto fondamentale di JDSys è quello della scalabilità automatica. E’ stato utilizzato a tale scopo il sistema di Autoscaling fornito da AWS, che permette di definire un numero minimo e massimo di istanze da istanziare in modalità on-premise. In JDSys sono state utilizzate un minimo di 3 istanze attive, fino ad un massimo di 10. Si noti come il parametro massimo è impostato ad un valore “basso” in virtù delle limitazioni fornite da AWS Educate.

Per monitorare il traffico di richieste verso il sistema, si utilizza il servizio Cloudwatch. La metrica utilizzata è il numero di pacchetti in ingresso al gruppo di autoscaling, in quanto fornisce un indice indicativo sul numero di richieste e quindi anche sul numero di operazioni che ogni nodo dovrà gestire complessivamente.

Analizzando i dati forniti da questo servizio, sono stati ricavati i valori critici in termini di traffico. Si è osservato infatti che per un traffico medio di 30000 pacchetti, i nodi possono andare in saturazione (out of memory) non riuscendo più a gestire un numero elevato di richieste concorrenti. Analogamente si è osservato che Per traffico medio sotto le 8000 richieste, il sistema garantisce sempre la normale operatività.

Sono stati quindi definiti due allarmi, uno per lo Scale In ed uno per lo Scale Out, proprio sfruttando i dati ricavati in precedenza. Lo Scale Out consiste nella rimozione di un’istanza dal gruppo di autoscaling, e viene attivato quando il traffico medio in ingresso risulta superiore a 10000 pacchetti per più di 5 minuti. Si è deciso di mantenere questa soglia pari ad 1/3 di quella critica in modo da fornire al servizio di autoscaling il tempo necessario ad istanziare la macchina e a rendere UP & Running il JDSys Node.

4.3 Overlay Network

Uno dei requisiti di progettazione chiave per Dynamo è che deve scalare in modo incrementale. Ciò richiede un meccanismo per partizionare dinamicamente i dati sull'insieme di nodi (cioè host di archiviazione) nel sistema. Lo schema di partizionamento di Dynamo si basa su un hashing coerente per distribuire il carico su più host di archiviazione. Nell'hashing coerente [10], l'intervallo di output di una funzione hash viene trattato come uno spazio circolare fisso o "anello" (ovvero il valore hash più grande si avvolge intorno al valore hash più piccolo). Ad ogni nodo del sistema viene assegnato un valore casuale all'interno di questo spazio che rappresenta la sua “posizione” sull'anello. Ciascun elemento di dati identificato da una chiave viene assegnato a un nodo eseguendo l'hashing della chiave dell'elemento di dati per ottenere la sua posizione sull'anello, quindi percorrendo l'anello in senso orario per trovare il primo nodo con una posizione maggiore della posizione dell'elemento. A B C E D F G Chiave K I nodi B, C e D memorizzano le chiavi nell'intervallo (A, B) incluso K. Figura 2: Partizionamento e replica delle chiavi nell'anello Dynamo. Tabella 1: Riepilogo delle tecniche utilizzate in Dynamo e dei loro vantaggi. Problema Tecnica Vantaggio Partizionamento Hashing coerente Scalabilità incrementale Elevata disponibilità per le scritture Orologi vettoriali con riconciliazione durante le letture La dimensione della versione è disaccoppiata dalle velocità di aggiornamento. Gestione degli errori temporanei Quorum approssimativo e trasferimento suggerito Fornisce elevata disponibilità e garanzia di durata quando alcune delle repliche non sono disponibili. Ripristino da guasti permanenti Antientropia mediante alberi Merkle Sincronizza le repliche divergenti in background. Rilevamento di appartenenza e fallimento Protocollo di appartenenza basato su pettegolezzi e rilevamento di errori. Preserva la simmetria ed evita di avere un registro centralizzato per la memorizzazione delle informazioni sull'appartenenza e sulla vita del nodo. 12099 Quindi, ogni nodo diventa responsabile della regione nell'anello tra esso e il suo nodo predecessore sull'anello. Il vantaggio principale dell'hashing coerente è che la partenza o l'arrivo di un nodo interessa solo i suoi vicini immediati e gli altri nodi rimangono inalterati. L'algoritmo di hashing coerente di base presenta alcune sfide. Innanzitutto, l'assegnazione casuale della posizione di ciascun nodo sull'anello porta a dati e distribuzione del carico non uniformi. In secondo luogo, l'algoritmo di base è ignaro dell'eterogeneità nelle prestazioni dei nodi. Per affrontare questi problemi, Dynamo utilizza una variante di hashing coerente (simile a quella utilizzata in [10, 20]): invece di mappare un nodo su un singolo punto nel cerchio, ogni nodo viene assegnato a più punti nell'anello. A tal fine Dynamo utilizza il concetto di “nodi virtuali”. Un nodo virtuale sembra un singolo nodo nel sistema, ma ogni nodo può essere responsabile di più di un nodo virtuale. In effetti, quando un nuovo nodo viene aggiunto al sistema, gli vengono assegnate più posizioni (d'ora in poi, "token") nell'anello. Il processo di messa a punto dello schema di partizionamento di Dynamo è discusso nella Sezione 6. L'utilizzo di nodi virtuali presenta i seguenti vantaggi: • Se un nodo diventa non disponibile (a causa di guasti o manutenzione ordinaria), il carico gestito da questo nodo viene distribuito uniformemente tra i restanti nodi disponibili. • Quando un nodo diventa di nuovo disponibile o viene aggiunto un nuovo nodo al sistema, il nuovo nodo disponibile accetta una quantità di carico approssimativamente equivalente da ciascuno degli altri nodi disponibili. • Il numero di nodi virtuali di cui un nodo è responsabile può essere deciso in base alla sua capacità, tenendo conto dell'eterogeneità nell'infrastruttura fisica.

KEYWORDS

Insert keyword text, Insert keyword text, Insert keyword text, Insert keyword text

JDSys organizza i nodi secondo una struttura ad anello. Una qualsiasi richiesta di lettura/scrittura per una chiave viene instradata a qualsiasi nodo nel cluster Cassandra. Il nodo determina quindi le repliche per questa particolare chiave. Per le scritture, il sistema instrada le richieste alle repliche e attende che un quorum di repliche riconosca il completamento delle scritture. Per le letture, in base alle garanzie di coerenza richieste dal client, il sistema o indirizza le richieste alla replica più vicina oppure indirizza le richieste a tutte le repliche e attende un quorum di risposte.

3. CLIENT API

* JDSys memorizza gli oggetti associati a una chiave tramite una semplice interfaccia; espone quattro operazioni: get(), put(), delete() e append(). L'operazione get(key) individua le repliche degli oggetti associate alla chiave nel sistema di archiviazione e restituisce un singolo oggetto o un elenco di oggetti con versioni in conflitto insieme a un contesto. L'operazione put(key, context, object) determina dove devono essere posizionate le repliche dell'oggetto in base alla chiave associata e scrive le repliche su disco. Il contesto codifica i metadati di sistema sull'oggetto che è opaco per il chiamante e include informazioni come la versione dell'oggetto. Le informazioni di contesto vengono memorizzate insieme all'oggetto in modo che il sistema possa verificare la validità dell'oggetto di contesto fornito nella richiesta put. Dynamo tratta sia la chiave che l'oggetto fornito dal chiamante come un array opaco di byte. Applica un hash MD5 sulla chiave per generare un identificatore a 128 bit, che viene utilizzato per determinare i nodi di archiviazione responsabili del servizio della chiave.:
* *get(key)* ricerca la chiave all’interno del sistema e, se questa esiste, restituisce il valore ad essa associata.
* *put(key,value)* inserisce la coppia chiave-valore all’interno del sistema. Se la chiave già esiste, il suo valore viene aggiornato.
* *delete(key)* elimina la coppia chiave valore associata alla key specificata
* *append(key,arg1)* modifica l’entry specificata tramite la chiave key, aggiungendo l’argomento arg1

ACM Reference format:

FirstName Surname, FirstName Surname and FirstName Surname. 2018. Insert Your Title Here: Insert Subtitle Here. In *Proceedings of ACM Woodstock conference (WOODSTOCK’18). ACM, New York, NY, USA, 2 pages.* https://doi.org/10.1145/1234567890

1 Insert Heading Level 1

The updated template, user manuals, samples, and required fonts, all are available at the URL <https://www.acm.org/publications/proceedings-template>. It contains said information for all three versions of MS Word (Windows and 2 versions of Mac). There are also separate links to the user guide, which can be referred to by the user. This URL also contains some useful video links, which describe how to add the template, structure the paper, and generate the layout, in different clips. **Display Formula with Number**

 (1)

**Continuation part of Paragraph Text** The user must style this paragraph in **ParaContinue** style, which follows immediately after the **DisplayFormula** (numbered equation). The **DisplayFormula** style is applied only in case of a numbered equation. A numbered equation always has a number to its right. Insert paragraph text here. **Display Formula without Number**



The **DisplayFormulaUnnum** style is applied only in case of an unnumbered equation. An unnumbered display equation never contains an equation number to its right, and this unique property distinguishes it from a numbered equation.



Figure 1: Figure Caption and Image above the caption [In draft mode, Image will not appear on the screen]

**Theorem/Proof/Lemma.** Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement.

....Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract.

1.1 Heading Level 2

In the below paragraph, it is explained how alt-txt value is placed in **MS Word 2010**. To add alternative text to a picture in Word 2010, follow these steps:

1. In a Word 2010 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. Select the **Alt Txt** option from the left-side panel options.
4. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

Below are steps to place alt-txt value in **MS Word 2013/2016**. To add alternative text to a picture in Word 2013/2016, follow these steps:

1. In a Word 2013/2016 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. In the settings at the right side of the window, click on the "Layout & Properties" icon (3rd option).
4. Expand **Alt Txt** option.
5. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

*1.1.1 Heading Level 3.* Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

*1.1.1.1 Heading Level 4.*Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

ACKNOWLEDGMENTS

Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

REFERENCES

[1] Patricia S. Abril and Robert Plant, 2007. The patent holder's dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan, 2007), 36-44. DOI: <https://doi.org/>10.1145/1188913.1188915.

[2] Sten Andler. 1979. Predicate path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL '79)*. ACM Press, New York, NY, 226-236. DOI:https://doi.org/10.1145/567752.567774

[3] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. DOI:https://doi.org/10.1007/3-540-09237-4.

[4] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY..

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00