Sistemi Distribuiti e Cloud Computing - A.A. 2015/16
Progetto 3:
Elastic Distributed Shared Memory in the Cloud

Docente: Valeria Cardellini
Dipartimento di Ingegneria Civile e Ingegneria Informatica
Università degli Studi di Roma "Tor Vergata"
cardellini@ing.uniroma2.it

Requisiti del progetto

Lo scopo del progetto è realizzare, in un linguaggio di programmazione a scelta, un **sistema elastico per il caching in memoria nel Cloud**, utilizzando i servizi Cloud messi a disposizione da AWS Educate o IBM BlueMix o altro provider (inclusa una piattaforma di Cloud privata, gestita ad esempio con OpenStack). Sono ovviamente esclusi dall'utilizzo servizi e sistemi già disponibili per l'in-memory caching (ad es. Amazon ElastiCache).

Un sistema elastico di in-memory caching permette di memorizzare oggetti di dimensione limitata nella memoria RAM di un insieme di nodi, detti *cache server*, distribuiti localmente (o geograficamente), supportando lo scale-out e lo scale-in dello spazio di memorizzazione a disposizione delle applicazioni che lo utilizzano. Per raggiungere tale obiettivo, il sistema di in-memory caching sfrutta la natura elastica delle istanze di macchine virtuali (VM), che nel Cloud possono essere acquisite o rilasciate rapidamente. Il sistema di in-memory caching può così offrire un livello elevato di prestazioni, realizzando lo scale-out ed incrementando il numero di cache server su cui viene eseguito al crescere del carico di lavoro a cui viene sottoposto. Tuttavia, poiché il costo delle istanze di VM non è trascurabile, per minimizzare i costi occorre anche che, in caso di scarso utilizzo del sistema di in-memory caching, una o più VM su cui i cache server sono eseguiti siano deallocate, realizzando così lo scale-in. Lo scale-in si rivela essere un'operazione di non facile realizzazione, a causa della necessità di preservare e reintegrare gli oggetti memorizzati nei cache server che vengono eliminati.

Il sistema di in-memory caching elastico nel Cloud deve soddisfare i requisiti funzionali e non funzionali elencati di seguito.

- Supportare le operazioni di inserimento, aggiornamento, recupero e cancellazione di oggetti di dimensione limitata nel sistema di in-memory caching. Gli oggetti possono essere identificati tramite una chiave, come avviene ad es. in Memcached [3, 4] , Redis [5], Hazelcast [1]. Tali operazioni devono essere esposte alle applicazioni tramite un'opportuna API.
- Supportare almeno una politica di cache replacement per determinare quali oggetti rimpiazzare nella memoria RAM di un nodo.
- Supportare una politica di bilanciamento del carico tra i cache server.

- Supportare l'elasticità dei cache server componenti il sistema: si richiede di sfruttare la possibilità di allocare o deallocare velocemente istanze di VM nel Cloud e di fornire almeno una politica decisionale per lo scale-out e lo scale-in dei cache server.
- Garantire l'affidabilità, utilizzando tecniche di replicazione degli oggetti su molteplici cache server del sistema. Nella relazione dovranno essere spiegati i tipi di guasti che il sistema di in-memory caching realizzato è in grado di tollerare.
- Supportare un modello di consistenza degli oggetti replicati; la scelta del modello deve essere motivata nella relazione.
- Utilizzare almeno un servizio Cloud (escluso Amazon ElastiCache o simili), eventualmente usufruendo del grant di AWS Educate.
- Valutare sperimentalmente le prestazioni del sistema realizzato, considerando un workload composto sia da sole operazioni di lettura, sia da un mix di operazioni di lettura e scrittura. A tale scopo si può usare il tool memaslap [2] o altri tool di load testing e benchmarking di sistemi di in-memory caching. Si realizzi inoltre un testing funzionale delle varie operazioni supportate dal sistema di in-memory caching realizzato, presentandone i risultati nella relazione.

Si progetti il sistema di in-memory caching ponendo particolare cura al soddisfacimento dei requisiti sopra elencati e delle altre eventuali scelte effettuate dal gruppo. I componenti del sistema di in-memory caching devono essere eseguibili nello spazio utente e senza richiedere privilegi di root. Si richiede inoltre che il sistema di in-memory caching sia configurabile, ad es. tramite un file in cui sia possibile specificare i valori dei parametri di configurazione (tra cui, numero di VM iniziali, dimensione massima degli oggetti).

È possibile usare librerie e tool di supporto allo sviluppo del progetto; le librerie ed i tool usati devono essere esplicitamente indicati e brevemente descritti nella relazione.

Scelta e consegna del progetto

Il progetto è dimensionato per essere realizzato da un gruppo composto da **3** studenti.

Per poter sostenere l'esame nell'A.A. 2015/16, **entro il 15/5/2016** è necessario prenotarsi per il progetto, comunicando via email al docente le seguenti informazioni:

- nominativi ed email dei componenti del gruppo;
- progetto scelto.

Nel caso in cui il numero di prenotazioni per il progetto scelto abbia raggiunto la soglia massima prevista, sarà necessario effettuare una nuova scelta tra i progetti ancora disponibili.

Eventuali modifiche relative al gruppo devono essere tempestivamente comunicate al docente e concordate con il docente.

Per ogni comunicazione via email è necessario specificare *[SDCC]* nel subject della email. Il progetto è valido **solo** per l'A.A. 2015/16 e deve essere consegnato improrogabilmente **entro il 30/11/2016**. La prova d'esame scritta deve essere superata **entro la sessione autunnale 2015/16**.

Per iscriversi al programma AWS Educate, si raccomanda di seguire le istruzioni fornite a lezione ed inviate via email agli studenti iscritti al progetto.

La consegna del progetto deve avvenire circa dieci giorni prima della data (da concordare con il docente) in cui si intende sostenere la discussione del progetto. Il materiale relativo al progetto deve essere consegnato personalmente al docente.

La consegna del progetto consiste in:

1. CD-ROM o link a spazio di Cloud storage contenente:
   (a) tutti i sorgenti (opportunamente commentati) necessari per il funzionamento;
   (b) relazione (almeno in formato pdf);
   (c) eventuali test funzionali e non funzionali;
2. copia cartacea della relazione (senza codice).

La relazione contiene:

- la descrizione dettagliata dell'architettura del sistema di in-memory caching e delle scelte progettuali effettuate, opportunamente motivate; nel descrivere l'architettura, occorre usare una metodologia a scelta tra quelle studiate nei corsi di ingegneria del software;
- la descrizione dell'implementazione;
- la descrizione delle eventuali limitazioni riscontrate;
- l'indicazione della piattaforma software usata per lo sviluppo ed il testing del sistema di in-memory caching;
- i risultati del testing opportunamente commentati;
- un howto per l'installazione, la configurazione e l'esecuzione del sistema di in-memory caching.

Valutazione del progetto

I principali criteri di valutazione del progetto saranno:

1. rispondenza ai requisiti;
2. originalità;
3. organizzazione del codice (leggibilità, modularità, ...);
4. organizzazione, chiarezza e completezza della relazione.

Riferimenti bibliografici

[1] Hazelcast, 2016. `https://hazelcast.com`.

[2] Memaslap, 2013. `http://docs.libmemcached.org/bin/memaslap.html`.

[3] Memcached, 2016. `https://memcached.org`.

[4] R. Nishtala, H. Fugal, S. Grimm, M. Kwiatkowski, H. Lee, H. C. Li, R. McElroy, M. Paleczny, D. Peek, P. Saab, D. Stafford, T. Tung, and V. Venkataramani. Scaling Memcache at Facebook. In *Proc. of 10th USENIX Symposium on Networked Systems Design and Implementation*, NSDI '13, pages 385–398, Lombard, IL, 2013. https://www.usenix.org/system/files/conference/nsdi13/nsdi13-final170_update.pdf.

[5] Redis, 2016. http://redis.io.