Università degli Studi di Padova

Ridondanza dei dati in ambiente virtualizzato

Relazione finale di Tirocinio Breve

Laureando: Stefano Peraro
Relatore: Prof. G. Clemente

Dipartimento di Ingegneria dell'Informazione
Anno Accademico 2011-2012

Indice

1 PROGETTO 1
1.1 Premessa 1
1.2 Azienda 2
1.3 Obiettivi 3
1.4 Tecnologie adottate 5
1.4.1 Virtualizzazione 5
1.4.2 MySQL 11
1.5 Soluzione Adottata 18
2 REALIZZAZIONE 19
2.1 Architettura e server 19
2.2 Configurazione 21
2.3 Tecniche di ripristino 27
2.4 Alternative di realizzazione 29
2.5 Considerazioni finali 31
BIBLIOGRAFIA 32

1 Progetto

1.1 Premessa

Con questo tirocinio, svolto presso l'internet service provider Terralink, si è voluto progettare ed approfondire il funzionamento e la gestione di un sistema per la ridondanza dei dati integrandolo con i cambiamenti in corso apportati alla rete.

Ho preso parte inizialmente all'analisi dei sistemi di virtualizzazione installati da poco in azienda e del db in produzione e successivamente ho contribuito attivamente alla stesura e alla implementazione delle specifiche per la realizzazione dell'alta affidabilità.

Si è ritenuto superfluo riportare in forma esplicita la notevole quantità di documentazione studiata per l'ideazione del progetto, ma mi sono principalmente soffermato nel descrive la tecnologia utilizzata, le linee guida e l'installazione di tutto il sistema. Per i dettagli tecnici ho elencato nella bibliografia i siti di reiferimento consultati.

L'attenzione alla crescita aziendale e il desiderio di offrire un servizio sempre migliore ai clienti, ha portato Terralink alla necessità di effettuare un investimento alla propria struttura informatica.

Si è quindi reso necessario dotarsi di nuovo hardware, molto più performante rispetto a quello già presente, che potesse servire anche al compito di sostituire la maggior parte dei server divenuti ormai obsoleti.

Con l'occassione della ristrutturazione dell'intera architettura di rete si è colta l'occasione di attivare nuovi servizi e aggiornare le soluzioni adottate per quelli già implementati.

Svariati servizi che in precedenza venivano forniti attraverso hardware fisico sono passati ad un sistema centralizzato basato su macchine virtuali e di conseguenza è stato necessario rinnovare le modalità di erogazione dei servizi.

Fin da subito si è visto che il vecchio sistema di gestione dati MySQL, basato su hardware fisico dedicato poneva degli ostacoli che non avrebbero consentito a mettere in funzione correttamente un sistema per garantire l'alta affidabilità e avrebbe mantenuto tutto il sistema su un architettura non compatibile con l'innovazione in atto, si è quindi proceduto con la analisi dei sistemi da implementare.

La prima parte della relazione descrive lo studio teorico delle tecnologie utilizzate e le varie funzioni messe a disposizione da queste. Ci si sofferma sulle caratteristiche della virtualizzazione e dei suoi aspetti tecnici, vengono elencati i principali vantaggi che questo nuovo sistema stà apportando nei centri informatici, si analizza il software MySQL Cluster, le tecniche che sfrutta per garantire la ridondanza dei dati e quali prerequisiti necessita per un corretto funzionamento. Successivamente le funzionalità di replica date dal MySQL base vengono studiate per ottimizzare l'architettura dei dati e le varie modifiche che dovranno essere intraprese durante l'utilizzo del DB.

La seconda parte invece descrive l'implementazione pratica del sistema. Si elencano i passi da svolgere per una corretta installazione e configurarione, si analizzano alcuni problemi riscontrati e vengono descritte le azioni di ripristino in caso di bisogno.

In tutto il progetto ci si è indirizzati in soluzioni software open source. A livello di virtualizzazione si è installato XenServer, hypervisor di Citrix, mentre il sistema operativo delle macchine virtuali si basa su Debian 6.0 Squeeze.

Per consentire una migliore compatiblità dei dati ci si è mantenuti sul vendor MySQL, questo per non introdurre ulteriori difficoltà al progetto con il conseguente aumento dei tempo finale di realizzazione. Si sono dovute testare due soluzioni diverse per garantire la ridondanza dei dati. Per primo è stato realizzato e testato un sistema clusterizzato di database utilizzando la suite MySQL Cluster ma per motivi progettuali successivamente ci si è indirizzati verso le funzioni di replica di MySQL 5.5

1.2 Azienda

Terralink è una Internet Company con sede a Bassano del Grappa, ha iniziato la sua attività come Internet Service Provider nel 1996, tra i primi nel Veneto e nel nordest.

La società attualmente è composta da 12 persone suddivise tra reparto commerciale, amministrativo, tecnico sistemistico e di programmazione. Anche se di piccole dimensioni, ciò non toglie che l'offerta dei servizi resi ai clienti non sia di elevata qualità, grazie al fatto di avere al proprio interno personale certificato che periodicamente partecipa a corsi di aggiornamento. Offre soluzioni personalizzate in ambito sistemistico e di web design definendo sempre un progetto iniziale che a mano a mano che viene sviluppato si adatta alle problematiche che si incontrano, alle

nuove feature richieste e all'integrazione delle ultime tecnologie. Grazie a questo modo di lavorare ha guadagnato la fiducia in tutti quest'anni di oltre 2500 clienti alcuni dei quali legati fin dalla sua nascita, diventando per loro punto di riferimento in ambito IT.

Inoltre ha siglato una serie di accordi con partner tecnologici di importanza nazionale e mondiale per offrire connettività ad Internet di altissima qualità. Nel corso degli anni Terralink ha investito notevoli risorse per potenziare ulteriormente la velocità di banda disponibile verso Internet e migliorado qualità, sicurezza e affidabilità.

Attualmente fornisce i seguenti servizi:

- Connessioni ad Internet a larga banda di diverse tipologie: ADSL, HDSL, SHDSL, Fibra Ottica
- connessioni ad Internet a larga banda via radio con tecnologia Hyperlan,
- connessioni ad Internet su rete telefonica con tecnologia RTG, ISDN, GPRS, UMTS
- rete propria di Hotspot WiFi pubblici
- registrazione diretta di domini con qualunque suffisso (.it, .com, .net, .org, .info, .eu, .biz e tutti i domini geografici mondiali)
- servizi avanzati di posta elettronica, hosting e housing
- sistemi antivirus e antispam a livello desktop, server e centralizzato come ISP
- registrazione PEC (posta elettronica certificata),
- gestione integrata della sicurezza informatica con firewall, filtro dei contenuti, IDS, Virtual Private Network
- soluzioni di telefonia alternative a Telecom Italia, grazie al servizio VoIP
- soluzioni avanzate di Videoconferenza ed E-learning su rete IP.

L'azienda grazie alla capacità di monitorare sistemi di rete complessi, controlla costantemente la rete, il traffico e gli apparati, verifica lo stato dei nodi e le performance attraverso l'utilizzo di software applicativi, integrati con un sistema di notifica immediata per mezzo di messaggistica SMS ed e-mail.

1.3 Obiettivi

L'azienda rende disponibile uno spazio WEB più o meno grande, a seconda delle esigenze del cliente, presso i propri server ospitati presso il datacenter di Bassano del Grappa gestito direttamente dal personale tecnico.

Per offrire un servizio di qualità in un servizio di hosting si deve fornire elevata velocità di interconnessione, sicurezza e alta affidabilità.

Il datacenter dispone di collegamenti ridondati alla rete Internet, alimentazione continua grazie alla presenza di gruppi elettrogeni in grado di garantire la continuità del servizio anche in caso di assenza di fornitura prolungata da parte del gestore dell'energia elettrica, temperatura condizionata, sistemi antincendio, agenti di backup

per la salvaguardia dei dati. I locali del datacetenter sono dotati inoltre di sistemi anti intrusione, sistemi di controllo e allarme.

Il sistema di hosting è caraterizzato dalle seguenti specifiche:

- spazio Web residente su server equipaggiati con Windows 2008 Server con IIS7 per la gestione dei siti Web
- accesso FTP diretto
- traffico illimitato
- supporto ASP.NET, ASP, PHP, Perl, CGI, ODBC per accesso a DB Access, servizi database su piattaforme MySQL e Microsoft SQL Server.

Attualmente in Terralink il bisogno, di migliorare le prestazioni, ottimizzare le risorse, semplificare la gestione e la manutenzione del sistema, aumentare la sicurezza e soprattutto diminuire i costi di esercizio, ha portato ad implementare nuove tecnologie.

In questi ultimi anni, con l'avvento della virtualizzazione, si tende a modificare l'aspetto fisico e logico dei sistemi informatici aziendali. In precedenza ogni server era costituito da una macchina fisica avente uno specifico sistema operativo configurato appositamente per svolgere un determinato compito.

Utilizzando la virtualizzazione, invece, è ora possibile creare, all'interno di un server fisico, più server virtuali che, pur non essendo presenti fisicamente, hanno le stesse potenzialità e prestazioni di analoghi server fisici, e che sono completamente indipendenti tra loro offrendo quindi la possibilità di eseguire contemporaneamente sistemi operativi differenti, aumentando la flessibilità, ottimizzando l'utilizzo dell'hardware e la disponibilità delle applicazioni e delle risorse. Tramite il consolidamento dei server si solleva inoltre gli amministratori dell' IT dalle dispendiose operazioni di gestione, consente di migrare le risorse sottoutilizzate in ambienti dove servono, e di liberare quelle non più utilizzate. Le risorse sono assegnate alle varie applicazioni solo per la quantità necessaria permettendo così anche un ottimizzazione dei consumi energetici.

All'interno di questo progetto di consolidamento, il ruolo che andrò a svolgere sarà di rende sempre disponibile (High Available) il database MySQL della sezione web, in pratica rendere frubile anche in caso di guasto, sia software che hardware, i database a cui accedono i siti internet dei clienti.

L'obbiettivo è di strutturare un sistema integrabile con i cambiamenti in corso nella rete aziendale, quindi basato sulla virtualizzazione, in grado di conteplare tutti i servizi che attualmente sono disponibili. Inoltre grazie appunto all'aggiornamento della rete e alla possibilità di fruire di nuove risorse nella quantità necessaria, si vuole implementare l'alta affidabilità dei dati. Alla clientela in questa maniera si può offrire un servizio capace di garantire l' utilizzo dei dati senza down time in caso di disservizio, mentre al reparto IT permette di migliorare la gestione di carichi in caso di aumento del traffico, di intervenite in caso di guasto, di apportare modifiche, di

risolvere eventuali errori e di aggiornare il sistema senza togliere la disponibilità del servizio.

1.4 Tecnologie adottate

Nella scelta della soluzione ci si è indirizzati sull'utilizzo di strumenti di confidenza adattandoli alla nuovo contensto di rete. Le tecnologie usate erano già implementate nella veccia infrastruttura ma non venivano sfruttate correttamente. Perciò sono stati approfonditi gli aspetti fondamentali della virtualizzazione, delle varie tipologie e ambienti in cui può operare e degli aspetti positivi che essa può apportare al sistema.

Ai fini della scelta progettuale si è anche esaminato in dettaglio le problematiche sorte nella ridondanda del DB. Si non analizzati le varie alternative di implementazione delle funzionalità messe a disposizione dal MySQL, si sono studiati i prerequisiti necessari all'implementazione e le necessarie modifiche da apportare ai dati.

1.4.1 Virtualizzazione

Data una risorsa hardware, si definisce virtualizzazione la creazione di una o più versioni astratte della risorsa fisica, dette anche Virtual Machine.

Le Virtual Machine si pongono tra SO e l'hardware, consentendo l'accesso alle risorse in maniera concorrente. Ogni Virtual Machine ha il suo sistema operativo con le relative applicazioni, come se fossero installate su macchine fisicamente separate.

I sistemi di virtualizzazione offrono ai sistemi operativi installati sulle Virtual Machine la condivisione della cpu, della memoria, delle periferiche e della connessione di rete.

Ad oggi esistono varie tipologie di virtualizzazione, che analizziamo qui di seguito.

Ambienti di virtualizzazione

La virtualizzazione è vista come la possibilità di far eseguire contemporaneamente sulla stessa macchina fisica diversi sistemi operativi, ma l'ambiente virtuale che ne permette l'esecuzione, indipendentemente da quale sistema operativo poi si scelga di utilizzare, è quello che svolge tutte le funzioni di interfacciamento tra hw e sw.

Un ambiente virtuale non è altro che un insieme di componenti hardware e periferiche, implementate via software, conosciute anche con il nome di Virtual Machine.

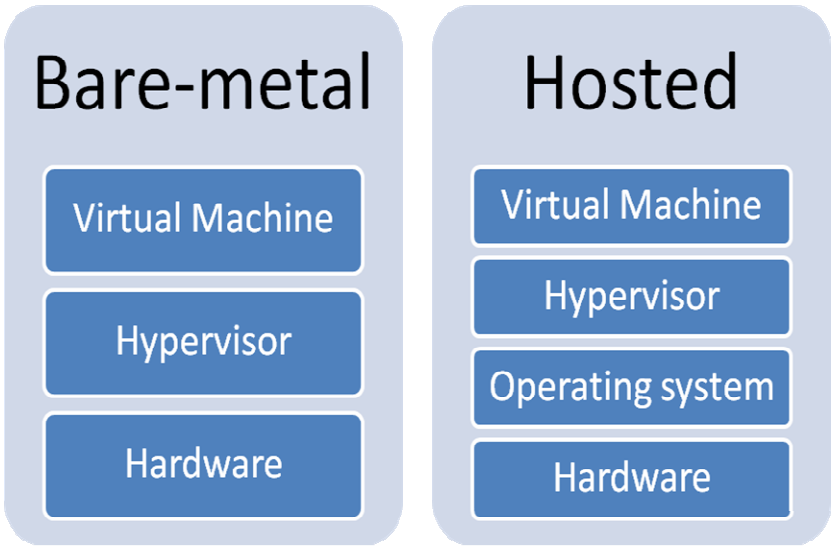
Una Virtual Machine può ospitare l'esecuzione di un sistema operativo ed una o più applicazioni che eseguono sopra di esso. Ogni Virtual Machine esegue sopra ad uno strato di virtualizzazione chiamato tecnicamente hypervisor (supervisore) che gestisce CPU, memoria e periferiche.

Esistono due tipi di hypervisor :

- Bare-metal o nativo
- Hosted o ospitato

Un hypervisor hosted è situato al di sopra di un sistema operativo detto ospitante (Windows, Linux, ecc...). In questo caso si parla di Hosted Virtualization.

Un hypervisor bare-metal è invece situato direttamente sopra l'hardware della macchina e ne possiede il controllo diretto e privilegiato. In questo caso si parla anche di virtualizzazione nativa.



SCHEMA VIRTUALIZZAZIONE HOST E BARE METAL

L'approccio hosted alla virtualizzazione consiste nell'installare l'applicazione di virtualizzazione all' interno di un sistema operativo host (ospistante). In questo modo sarà il sistema operativo fisico ad interfacciare mediante driver lo strato fisico sottostante, estendendo di fatto la compatibilità dell'applicazione a tutto l'hardware supportato dall'SO stesso. Il rovescio della medaglia sono prestazioni relativamente basse, in quanto le chiamate privilegiate effettuate dalle VM dovranno prima essere processate dal virtualizzatore, poi dall'SO e dai relativi driver.

L'approccio alla virtualizzazione mediante hypervisor bare-metal consiste nell'installare direttamente sulla macchina fisica uno strato hardware detto hypervisor, che controllerà le risorse fisiche che a loro volta gestiranno le VM con i relativi SO. In questo modo si avranno prestazioni nettamente superiori, in quanto le chiamate privilegiate verranno gestite dall'hypervisor stesso in maniera estremamente performante.

Questa scelta operativa limita però di molto il supporto hardware (dovranno essere scritti driver appositi), obbligando il produttori di hypervisor a processi di certificazione di hardware compatibile, che dovrà essere utilizzato in fase di deployment del prodotto.

Attraverso la virtualizzazione nativa è possibile ottenere prestazioni di gran lunga migliori rispetto alla virtualizzazione hosted, perché ci sono meno strati per arrivare all'hardware.

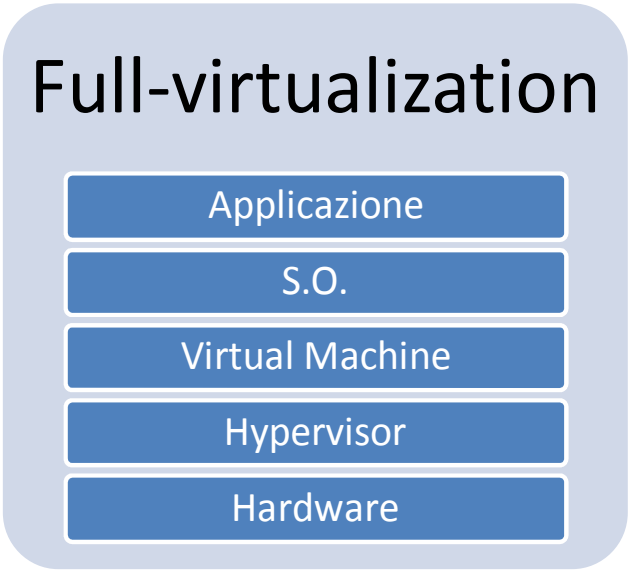
Entrambe le tipologie, permettono di eseguire VM al loro interno e dal punto di vista del filesystem vengono considerate come un insieme di file (file del disco fisso, file della RAM, file per lo swap, file di configurazione, ecc...).

Tecniche di virtualizzazione

Esistono tre differenti tecniche di virtualizzazione che si sono affermanti

• **Full virtualization**: consente l'esecuzione di sistemi operativi in un ambiente totalmente ricreato ed isolato dal sistema operativo che lo ospita. Tale ambiente è realizzato da un apposito software che si occupa di replicare l'hardware necessario traducendo le istruzioni eseguite dal sistema ospite in qualcosa che il sistema operativo ospitante sia in grado di eseguire. Come è possibile intuire la traduzione di ogni singola istruzione e molto onerosa a livello di potenza di calcolo, per ovviare a ciò è necessario eseguire la maggior parte delle istruzioni in modo nativo.

Questi sistemi posso utilizzare sia hypervisor hosted che bare metal e dipendono quindi dalla presenza di un sistema operativo o di un hypervisor che permetta l'esecuzione della VM il quale genera e gestisce le macchine virtuali. La virtualizzazione completa è quel tipo di virtualizzazione che consente alle VM di eseguire sistemi operativi completi e non modificati. Ciò è ottenuto tramite la completa simulazione in software di risorse hardware.



Full Virtualization Bare Metal

• **Hardware Assisted Virtualization**: Questa tecnica si appoggia a speciali funzioni della CPU (Intel VT e AMD-V) e permette l'esecuzione direttamente in hardware di alcune chiamate della macchina virtuale.

E' possibile dotare, un VM della capacità di sfruttare queste estensioni consentendo una virtualizzazione completa ma con un degrado di prestazioni minimo, paragonabile o in alcuni casi superiore a quello offerto dalla para-virtualizzazione. Ciò è ottenuto attraverso la parziale implementazione in hardware di complessi meccanismi quali la gestione delle istruzioni sensibili non privilegiate o la traduzione degli indirizzi di memoria delle macchine virtuali a indirizzi fisici, che normalmente deve gestire la VM.

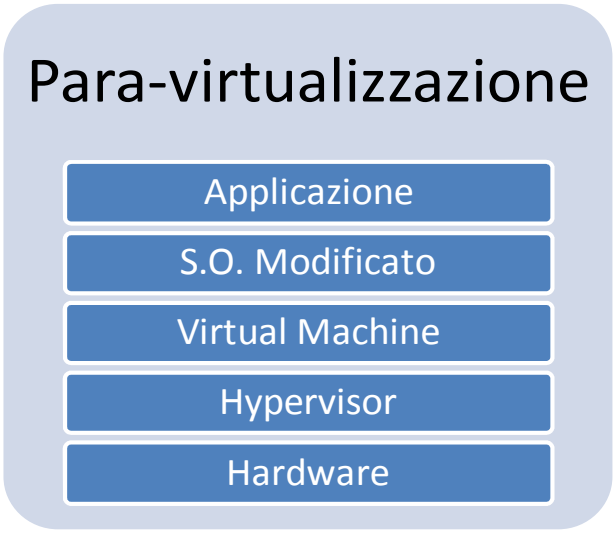
Tuttavia, questo tipo di virtualizzazione è strettamente dipendente dalle estensioni hardware di cui si è parlato, e quindi l'utilizzo è riservato ad architetture che le implementano. La diffusione di questo tipo di cpu ormai ha raggiunto livelli di mercato notevoli, è normale trovarle supportate dai vari prodotti di virtualizzazione unite alle altre tecniche.

• **ParaVirtualization** (OS Assisted Virtualization): è una tecnica in cui il sistema operativo virtualizzato riesce a comunicare direttamente con l'hypervisor. Questo comporta in alcuni casi la necessità di modificare il kernel del sistema operativo da virtualizzare. Il vantaggio nell' usare la Paravirtualization consiste nella maggior velocità di esecuzione delle applicazioni utente.

La para-virtualizzazione nasce in risposta al degrado di performance dovuta alla traduzione delle varie richieste tra più strati nella virtualizzazione completa.

La para-virtualizzazione evita la traduzione esplicita delle istruzioni attraverso l'impiego di un ambiente bare-metal, che fornisce un'interfaccia software simile a quella di un sistema operativo. I sistemi guest in esecuzione sulle macchine virtuali possono usufruire di tale interfaccia per avere un accesso filtrato alle risorse hardware che vengono comunque gestite a basso livello dall'hypervisor.

Per poter utilizzare tale interfaccia, i sistemi operativi guest devono essere opportunamente modificati in modo da sostituire le componenti che interagiscono con l'hardware fisico nell'esecuzione non virtualizzata, con meccanismi che sfruttino l'interfaccia fornita dall'hypervisor.



Paravirtualizzazione

I vantaggi offerti dalla virtualizzazione sono molteplici.

Maggiore sfruttamento dell'hardware

Molto spesso i servizi in esecuzione su una determinata macchina non sono sufficienti a saturare la capacità delle risorse che essa possiede. Attraverso l'esecuzione di più VM sulla stessa macchina, il numero di servizi offerti può crescere fino a ridurre l'inutilizzo delle risorse.

Isolamento

Si può sfruttare l'esecuzione di più VM per isolare ambienti sicuri, da altri potenzialmente a rischio. Ad esempio, servizi poco affidabili e da esporre al pubblico possono essere implementati su determinate VM, senza compromettere la sicurezza dell'intero sistema.

Controllo delle risorse

Ad ogni macchina virtuale si possono assegnare risorse in base a precisi criteri. Ciò evita che alcune applicazioni possano sfruttare eccessivamente CPU, memoria od altro a discapito di altre applicazioni che vedrebbero la propria funzionalità ridotta.

Test

Attraverso la virtualizzazione si può disporre di ambienti di complessità variabile, anche distribuiti, senza la necessità di dotarsi di più hardware. Ciò consente il test di molte applicazioni che necessitano di una tale complessita d'ambienti, come ad esempio applicazioni parallele o peer-to-peer. Inoltre le applicazioni possono essere testate per più sistemi operativi che possono essere ospitati dalle VM.

Facilità d'amministrazione

L'implementazione di una macchina virtuale richiede molto meno sforzi da parte degli amministratori. Inoltre le macchine virtuali possono essere fermate, messe in pausa, riavviate o clonate su altre macchine fisiche senza dover interrompere il funzionamento della macchina reale. Ciò consente di utilizzare la virtualizzazione per implementare meccanismi di fail-over o di bilanciamento del traffico molto più efficaci rispetto alla semplice replicazione hardware.

Al momento esistono diversi software che gestiscono la virtualizzazione. Nel corso degli anni alcuni si sono sviluppati e innovati cambiando di molto l'approccio della realizzazione e gestione delle infrastrutture di rete sul mercato, altri si sono specializzati nel deployment delle virtual appliance e del virtual desktop, diventando leader di mercato, mentre altri si sono limitati ad alcune nicchie di mercato, altri sono stati inglobati mentre altri sono andati in disuso. Ecco un elenco dei prodotti attivamente presenti nel mercato.

- KVM (Qumranet)
- Parallels Desktop / Workstation / Server (Parallels)
- QEMU (Fabrice Bellard)
- Oracle VM VirtualBox (Oracle)
- Hyper V (Microsoft)
- VSphere ESXi (WMware)
- Xen (Citrix)

XenServer

Nel mercato sono presenti diverse soluzioni di virtualizzazione ognuna con diversi pregi e difetti. L'azienda dove ho svolto il tirocinio sta attuando una completa ristrutturazione della rete interna e dei servizi forniti ai clienti, la scelta tra i vari prodotti presenti sul mercato è caduta su Citrix XenServer, in quanto offriva un prodotto completo di virtualizzazione in ambiente nativo gratuitamente. Inoltre garantiva compatibilità sia con prodotti predisposti alla virtualizzazione, cioè aventi come processore le nuove CPU Intel-VT AMD-V, sia con hardware un po' datato non compatibile utilizzando altri prodotti. Di contro utilizzando la versione gratuita non si può avere supporto diretto con la casa produttrice, ma lo studio del manuale messo a disposizione e l'utilizzo in caso estremo del forum, con la presenza di esperti ha sostituito degnamente questa mancanza.

XenServer è un prodotto nato negli ultimi anni, dopo l'acquisto avvenuto nel 2007 da parte di Citrix. All' inizio Xen era un progetto creato e utilizzato all'università di Cambridge, dove è stata implementata per la prima volta la paravirtualizzazione. Solo nel 2005 si sono visti sul mercato i primi prodotto Xen compatibili anche con le nuove CPU. Piano piano conquistando sempre più clienti e migliorandosi negli anni è diventato il rivale principale dell'altro hypervisor più conosciuto, VMWare. Negli ultimi tempi, tramite l'acquisto da parte di Citrix il prodotto è cresciuto notevolemente nei servizi offerti e nella qualità dei prodotti sviluppati.

XenServer è il sistema operativo dei server fisici utilizzati, riesce a bilanciare le risorse necessarie tra i due host in modo dinamico e automatico, oltre a fornire

un'incredibile quantità di tool necessari per la gestione e il monitoraggio delle macchine virtuali attive. Il sistema operativo che attualmente viene più utilizzato in azienda per le macchine virtuali dove andranno a risiedere i DB MySQL è Debian 6.0.

Per queste macchine virtuali, dove girano servizi vitali per la produttività aziendale, è indispensabile effettuare periodicamente ad intervalli ben stabiliti dei backup. Grazie a XenCenter è possibile effettuare backup dell'intera macchina virtuale in modo che se dovessero accadere degli inconvenienti software (aggiornamenti falliti, errori, etc) che non permettono più l'avvio del sistema operativo si può in brevissimo tempo rimettere in produzione (non necessariamente sullo stesso hardware) il servizio funzionante al momento del precedente backup.

Il controllo centralizzato di tutta l'infrastruttura virtuale è reso possibile grazie a Citrix XenCenter. In sostanza è la parte che si occupa di gestire le varie funzionalità della piattaforma XenServer. Tramite l'interfaccia principale è possibile accedere ad una serie di funzioni che permettono l'amministrazione dell'intera infrastruttura. C'è la possibilità, ad esempio, di accedere alle periferiche dati connesse e poter monitorare l'uso complessivo dello spazio di archiviazione disponibile e controllare tutto ciò che riguarda la connessione. Si può inoltre monitorare in tempo reale l'uso delle risorse che una determinata macchina virtuale richiede, in questo modo se un server necessita di più ram o più potenza di calcolo è possibile modificarne i valori rapidamente e con semplicità. E' così possibile configurare in modo ottimale l'uso effettivo delle risorse che sono fisicamente disponibili e grazie a questo non viene più sprecata potenza di calcolo come succedeva con la gestione classica.

1.4.2 MySQL

Il codice di MySQL viene sviluppato fin dal 1979 dalla ditta TcX dataconsult, adesso MySQL AB, ma è solo dal 1996 che viene distribuita una versione che supporta SQL, prendendo spunto da un altro prodotto: mSQL. Il codice di MySQL è di proprietà della omonima società e viene distribuito con la licenza GNU GPL oltre che con una licenza commerciale.

Il 16 gennaio 2008 Sun Microsystems ha acquistato la società per un miliardo di dollari, stimando il mercato del database in 15 miliardi di dollari. Il 20 aprile 2009 alla stessa Sun Microsystems è stata proposta l'acquisizione da parte di Oracle per 7,4 miliardi di dollari.

MySQL è il database relazionale open source più diffuso al mondo e costituisce oggi la soluzione ottimale per chi è in cerca di un database veloce, flessibile, affidabile e soprattutto gratuito

MySQL è un RDBMS, ossia un sistema di gestione per database relazionali. MySQL si occupa della strutturazione e della gestione a basso livello dei dati stessi, in modo da velocizzarne l'accesso, la modifica e l'inserimento di nuovi elementi. L'acronimo SQL significa Structured Query Language ed indica il linguaggio standard di

interrogazione dei DataBase. Possiede delle interfacce per diversi linguaggi, compreso un driver ODBC, due driver Java e un driver per Mono e .NET. MySQL è diventato uno dei sistemi per la gestione di database più importanti al mondo. Dai piccoli progetti di sviluppo ad alcuni tra i siti più famosi e più prestigiosi sul Web, MySQL si è dimostrato una soluzione solida e veloce per tutti i tipi di necessità di archiviazione di dati.

Un Database (traducibile in italiano come "base di dati") non è un altro che un insieme di dati logicamente correlati fra loro. I Data Base Management System (DBMS) sono invece i prodotti software in grado di gestire i database e le loro caratteristiche sono:

- capacità di gestire grandi quantità di dati
- condivisione dei dati fra più utenti e applicazioni
- utilizzo di sistemi di protezione e autorizzazione per l'accesso ai dati stessi

La scelta di utilizzare questo DBMS in azienda è dovuto appunto per la sua diffusione a livello mondiale nelle soluzioni di hosting dei siti internet. La tipica struttura che negli anni si è andata a consolidare è la LAMP (Linux-Apache-MySQL-PHP).

All'interno del mio progetto si sono dovute analizzare le soluzioni più adatte per fornire la ridondanza dei dati offerte da MySQL. La scelta è ricaduta sul MySQL Cluster 5.1.19 pacchetto del tutto indipendente dal MySQL 5.5, e sulle funzioni di replica di quest'ultimo.

La neccessità di studiare inizialmente la clusterizzazione e successivamente adottare un'altra tecnica è stata dovuta dal fatto che venivano richieste troppe risorse per il funzionamento corretto del sistema. Risorse che nell' evoluzione del servizio sarebbero aumentate notevolmente, si è quindi scelto di adottare una soluzione alternativa la quale richiedeva risorse piu modeste garantendo la stessa affidabilità.

MySQL Cluster 7.1.19

Un cluster di rete collega macchine indipendenti a lavorare insieme in modo coordinato. La configurazione del cluster varia notevolmente a seconda delle scelta delle tecnologie e dello scopo a cui è preposto il sistema.

MySQL Cluster ha un'architettura shared-nothing.

In un'architettura shared-nothing, ogni nodo è indipendente e autonomo. Su ogni nodo del cluster esiste una copia fedele e aggiornata della base di dati, ogni nodo scrive localmente le informazioni. Per cui, in caso di caduta di un qualsiasi nodo, non vi è alcuna perdita di informazioni, inoltre un nodo può essere clonato, anche per il ripristino, a partire da un altro nodo attivo, o da un backup.

MySQL Cluster 7.1.19, non è MySQL configurato su un cluster di computer, ma è un prodotto a sé stante, costruito per soddisfare i requisiti di alta affidabilità e parallellismo. Attraverso l'utilizzo di macchine standard, si vuole garantire l'affidabilità con prestazioni elevate. Questa soluzione è infatti stata sviluppata,

tenendo in considerazione i settori nei quali si necessita di mantenere sempre on-line i propri servizi. A questo scopo, è stato sviluppato un prodotto che promette di garantire 5/9 di disponibilità (99.999%), con un tempo di failover inferiore al secondo. Grazie a queste caratteristiche, si stima che tra down-time programmato e non, in un anno, il database può fermarsi al massimo per 8 secondi. Inoltre, poiché è una soluzione scalabile, garantisce la possibilità di adeguare il sistema a carichi crescenti, con tempi di risposta bassi, quasi in tempo reale, sfruttando fino a 255 nodi.

L'architettura di MySQL Cluster 7.1.19

MySQL Cluster è un RDBMS, ad alta affidabilità, alte prestazioni, costruito usando l'architettura shared-nothing e una interfaccia SQL standard. Il sistema è costituito da nodi, distribuiti su macchine diverse, anche dislocate geograficamente, per assicurare la continuità del servizio, nel caso in cui un nodo o la rete siano compromessi. MySQL Cluster usa uno storage engine, che provvede alla memorizzazione dei dati sui nodi, abilitando l'accesso attraverso query SQL standard.

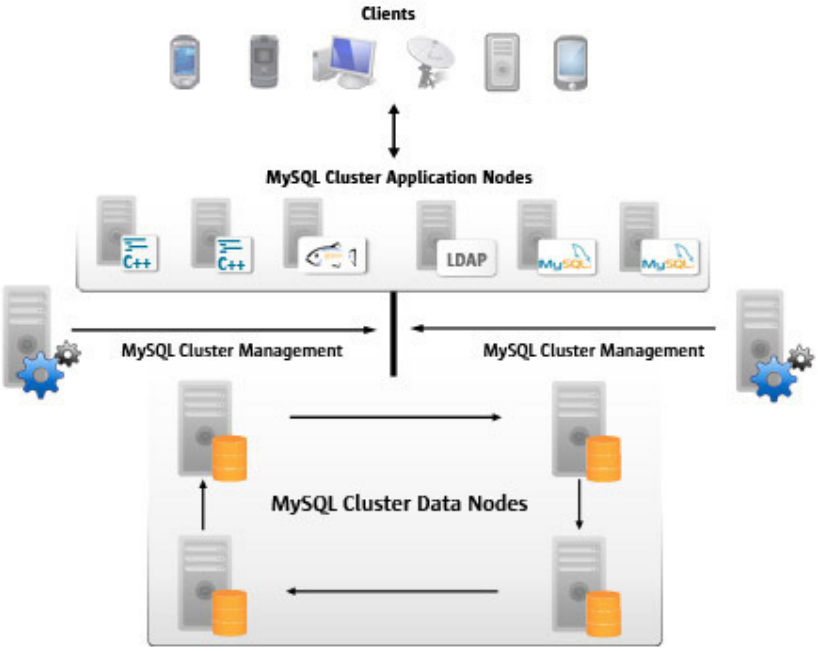
MySQL Cluster tollera i fallimenti di diversi nodi e si riconfigura al volo per mascherarli. Le capacità di self-healing, la trasparenza nella distribuzione dei dati e del loro partizionamento applicativo, permettono di avere un modello di programmazione semplice, il quale consente agli sviluppatori di includere facilmente l'alta affidabilità all'interno delle applicazioni, senza aver a che fare con codice a basso livello.

MySQL Cluster è costituito da 3 componenti principali:

Nodi SQL o Application Node (MySQL): sono nodi MySQL Server, che consentono l'accesso al cluster tramite le classiche query SQL. Si può sfruttare la metodologia scale-out per incrementare prestazioni, capacità e scalabilità della soluzione.

Nodi Dati: sono i nodi nei quali sono memorizzati effettivamente i dati e che gestiscono le transazioni. La continua disponibilità e un throughput elevato sono garantiti mediante la replicazione delle partizioni dei dati. Ogni data node è capace di gestire interrogazioni e transazioni, comunicando con gli altri.

Nodo di Management: viene utilizzato per l'amministrazione e la configurazione del cluster. E' destinato a prendere decisioni in caso di problemi al cluster: ad esempio, nel caso si verifichi un fenomeno di caduta di un link di rete, decide quali nodi devono essere considerati attivi e quali fuori dal cluster. Di solito se ne prevede uno, poiché il cluster può funzionare a prescindere dal fatto che sia sempre attivo o meno, comunque, per evitare che esso diventi un possibile single point of failure, anche questo nodo può essere replicato.



Architettura di MySQL Cluster

Il DB engine : NDB

NDB (Network Database) è il DB engine introdotto in MySQL Cluster: esso si occupa di distribuire i dati sui nodi del cluster.

I dati sono automaticamente ripartiti tra i server, con una frammentazione del tipo orizzontale, ovvero suddividendo le tabelle in *k* insiemi di tuple, dove *k* è il numero di nodi del sistema. Questa operazione viene effettuata operando con un algoritmo, totalmente trasparente all'utente. Ognuna delle partizioni è allocata su un nodo primario e replicata su un nodo secondario. I nodi che possiedono gli stessi dati (frammenti primari e secondari), sono raggruppati insieme, in questo modo, si avrà sempre una copia dei dati disponibile, per cui si ha la possibilità di elaborarli in parallelo, oppure di risolvere un problema dovuto alla caduta di un nodo.

La replicazione dei dati in un gruppo avviene in modo sincrono. La memorizzazione dei dati sui nodi può avvenire sia su disco che in memoria centrale. In quest'ultimo caso, le performance ovviamente sono altissime.

Vantaggi di MySQL Cluster

L'architettura basata sui nodi di MySQL Cluster è stata accuratamente progettata per l'alta disponibilità del sistema, se un nodo è soggetto a un fallimento, le transazioni possono essere redirezionate ad un altro nodo. I dati all'interno di un nodo sono replicati, in modo che il sistema possa tollerare fallimenti di un nodo. I nodi di management possono essere spenti e poi riaccesi senza conseguenze sulle attività degli altri nodi.

Il sistema, disegnato in questo modo, è reso altamente affidabile poiché le probabilità che un nodo rappresenti un single point of failure, sono portate al minimo: il sistema continua a funzionare anche in presenza di un fallimento di uno dei nodi che lo compongono.

Inoltre, vengono utilizzate altre tecniche, col fine di aumentare l'affidabilità e la disponibilità del sistema:

- i dati sono replicati in modo sincrono tra i nodi, portando al minimo i tempi di failover.
- i nodi sono eseguiti su diverse macchine, permettendo l'operatività anche in presenza di guasti hardware.
- i nodi sono realizzati col paradigma shared-nothing: ogni nodo ha il suo storage locale.
- i single points of failure sono ridotti al minimo: il sistema può tollerare la caduta di qualunque nodo nel sistema.

Le applicazioni che fanno uso di MySQL Cluster, ne ricavano i seguenti benefici:

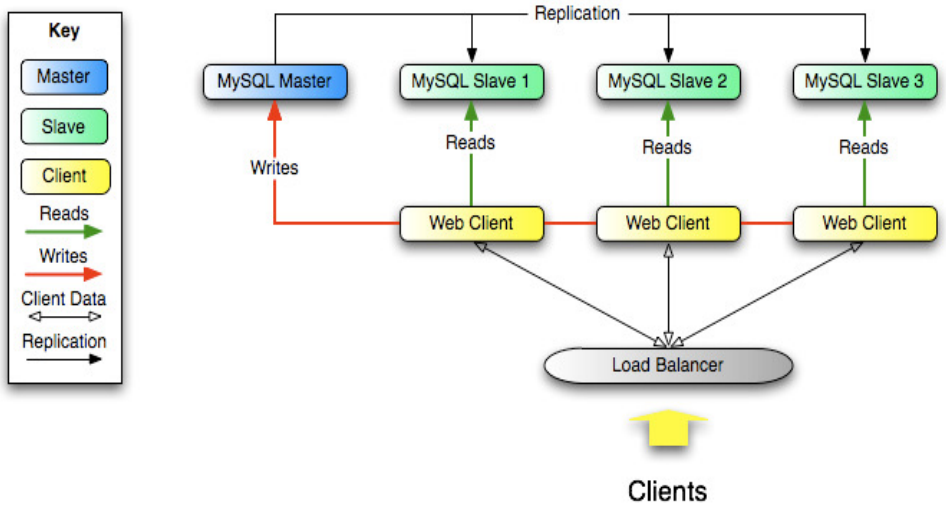
- indipendenza dei dati: l'applicazione può essere scritta senza conoscere nulla sulla distribuzione dei dati. L'engine di memorizzazione ha cura di gestire tutti i dettagli a basso livello, come la replicazione e il failover automatico.
- trasparenza di rete e di distribuzione: l'applicazione non dipende né dai dettagli operazionali della rete, né dalla distribuzione dei dati tra i nodi.
- trasparenza di replicazione e di partizione: l'applicazione può essere scritta senza curarsi della replicazione e indipendentemente da come sono partizionati i dati.
- una interfaccia SQL standard: per accedere ai dati, non bisogna utilizzare linguaggi di basso livello, anche se non sono escluse API native.

La replicazione

Un database ad alta affidabilità è in grado di sopportare la caduta di uno qualsiasi degli apparati che costituiscono il sistema. Tipicamente, per raggiungere tale obiettivo, si utilizzano tecniche di replicazione, sia dei processi che dei dati, in modo che, se una componente smette di funzionare, l'integrità del sistema non viene compromessa. Tale concetto è normalmente applicato a tutti i tipici servizi di rete, i quali devono essere ridondanti al fine di garantire che, in caso di interruzione dei processi principali, continuino ad essere erogati. In un database distribuito, avere la replicazione dei processi significa che se per una qualsiasi ragione un processo su di una macchina dovesse cadere, il database resterebbe comunque accessibile, grazie ad un processo di backup, che ne prende il posto, localizzato sulla macchina stessa o su una differente. La replicazione dei dati, invece, presuppone che un dato non sia memorizzato in una singola posizione: quindi, se questo si corrompe o diventa indisponibile, si ha la possibilità di accedervi e ripristinarne il contenuto in base ad una copia di backup. Oltre ad un efficiente sistema di backup, se le copie sono su macchine on line, le richieste di lettura possono essere esaudite accedendo ad una ognuna di queste, bilanciando il carico del sistema.

La replicazione in MySQL

La replicazione fornita da MySQL pricipalmente è del tipo master-slave: sul server master vengono eseguite le query di scrittura (inserimento, update, cancellazione), ma non sono vietate le operazioni di lettura. Sui server di tipo slave dovrebbero essere eseguite esclusivamente le operazioni di lettura, ma per default le scritture non sono disabilitate.



Schema Master - Slave

Il master riceve le connessioni relative alle modifiche della base di dati e scrive le modifiche localmente sul suo disco e in un file di log, accessibile ai client slave. E' l'unico a conoscere l'identità di tutti gli slave e ne riceve le connessioni. In caso di fallimenti, deve essere sostituito, e devono essere avvertiti gli slave, i quali dovranno resettare la loro memoria locale relativa agli aggiornamenti.

Gli slave leggono il file di log del master ed eseguono le operazioni lì riportate aggiornando il database locale, mantenendo una memoria locale per ricordare fino a che punto si sono letti gli aggiornamenti dal master.

Il rapporto tra master e slave è così definito: un master può avere più slave, mentre uno slave può avere un solo master.

La replicazione dei dati dunque avviene in modo asincrono, ovvero una transazione applicata sul master, può attendere un po' di tempo prima di essere visibile dagli slave. Tale modalità si contrappone ad una replicazione sincrona, dove l'aggiornamento viene propagato in automatico su tutti i nodi.

Per default, MySQL sfrutta la replicazione basata su istruzioni laddove le istruzioni SQL (non le effettive modifiche dei dati) siano replicate dal master agli slave. La replicazione basata su istruzioni fa parte del server MySQL dalla versione 3.23.

Un vantaggio della replicazione basata su istruzioni è che, in alcuni casi, una quantità minore di file finisce per essere scritta nei file di log, per esempio quando gli

aggiornamenti o le cancellazioni riguardano molte righe. Nel caso delle istruzioni semplici che riguardano solo poche righe, la replicazione basata su righe può occupare meno spazio.

La replicazione basata su istruzioni presenta alcuni svantaggi,in particolare, non può supportare le istruzioni non deterministiche, come ad esempio la funzione dell'ora corrente.

Introdotta con MySQL 5.1, la replicazione basata su righe registra le modifiche delle singole righe delle tabelle. Con la replicazione basata su righe, il master scrive i messaggi nel log binario che indica quali righe delle tabelle siano state modificate. Ciò è simile alle forme più tradizionali di replicazione che si trovano negli altri RDMS. In generale, la replicazione basata su righe richiede meno lock su master e slave, il che significa che è possibile ottenere una concorrenza più elevata. Uno svantaggio della replicazione basata su righe è che solitamente genera più dati da registrare. Per esempio, se un'istruzione modifica 100 righe in una tabella, con la replicazione basata su righe dovranno essere registrate 100 modifiche, mentre con la replicazione basata su istruzioni dovrà essere replicata una singola istruzione SQL.

A partire da MySQL 5.1.8, utilizzando il logging misto, il formato di binary logging può essere modificato in tempo reale a seconda dell'evento che si sta registrando. Quando è attivo il logging misto, si usa per default la replicazione basata su istruzioni, ma in alcuni casi si passa automaticamente alla replicazione basata su righe.

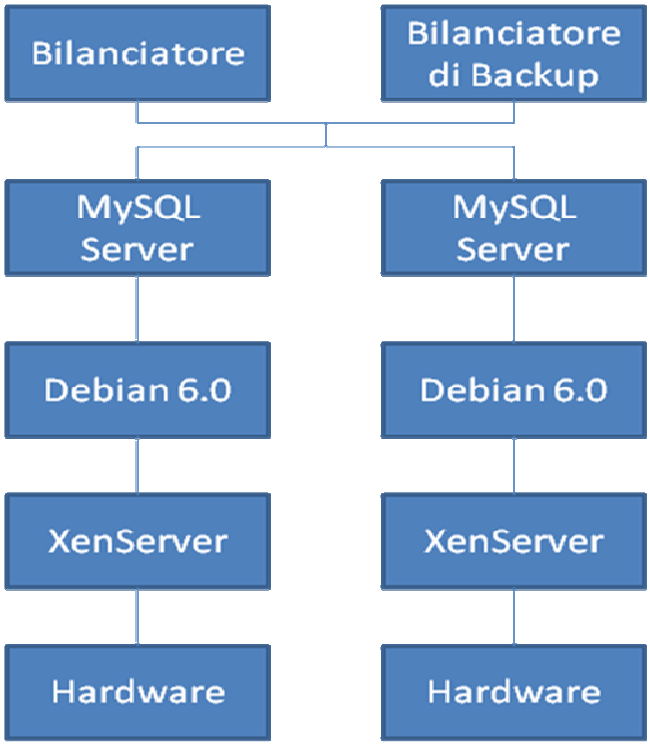
Per quanto riguarda la scalabilità della soluzione, il carico del sistema può essere bilanciato effettuando uno scale-out dell'architettura: se ho bisogno di aumentare le capacità del sistema, posso aggiungere più nodi slave (o anche master), questi sistemi non devono necessariamente avere caratteristiche computazionali elevate. Tipicamente, invece, per incrementare le prestazioni di un sistema si provvede ad aggiungere risorse fisiche ad un nodo quando, queste non sono sufficienti a supportare il carico di lavoro (aggiungo ram, processori…): tale metodo è detto scale-up e si contrappone a quello descritto precedentemente.

Come descritto fino ad ora, la tipica soluzione di replica, utilizza un master e uno slave. Sul master sono consentite letture e scritture mentre sullo slave vengono effettuate solo letture. Nel mio caso invece si utilizzerà una soluzione master-master. In una configurazione master-master, due server vengono combinati in coppia, affinché agiscano reciprocamente da master e slave. Sebbene questa configurazione offra il beneficio di poter scrivere su uno qualsiasi dei sistemi con la certezza che la modifica sarà ad un certo punto replicata, aumenta il grado di complessità delle attività di impostazione, configurazione e amministrazione.

1.5 Soluzione Adottata

La soluzione finale adottata è la replicazione. Tramite di essa si è potuto clonare su due server distinti i dati per garantirne l'uso da parte delle applicazioni. Grazie alla replicazione si è potuto installare inoltre un sistema affidabile, molto veloce nel trasferimento dati e quando richiesto facilmente scalabile, si sono aggiornate le tecnologie consentendo così anche un monitoring più dettagliato del traffico e introdotto la ridondanza del servizio sia in hw che in sw. Con una capacità di calcolo più elevata si è migliorata anche l'elaborazione delle query.

Il sistema adottato implementato è schematizzato nell' immagine seguente.



Schema Sistema Finale

Al di sopra dell nostro server MySQL si è installato un bilanciatore principale e uno di backup che indirizzano le richieste ai due server. Nel caso uno dei due server diventasse irraggiungibile le richieste verranno indirizzate tutte verso l'altro garantendo così la continuità del servizio.

2 Realizzazione

Con riferimento alle tecnologie discusse nel capitolo precedente, si va ora a descrivere le azioni necessarie per l'effettiva realizzazione del sistema. Si espongono le nuove risorse hardware utilizzate, le prestazioni degli host e le caratteristiche della nuova area di storage. Sucessivamente è descritto come installare l'hypervisor XenServer e come configurarlo correttamente. Sono inoltre riportati i dettagli dei file usati per impostare correttamente le due tecniche di ridondanza: MySQL Cluster 7.1.19 e MySQL 5.5. Vengono anche esposte le istruzioni necessarie per un corretto ripristino dell'alta affidabilita in caso di guasto.

2.1 Architettura e server

La virtualizzazione è una tecnologia software ampiamente sperimentata che sta trasformando il mondo IT e cambiando dalle fondamenta il modo di utilizzare le risorse informatiche. La virtualizzazione consente di eseguire più sistemi operativi e applicazioni contemporaneamente nello stesso computer, aumentando in tal modo l'utilizzo e la flessibilità dell'hardware, superando il vecchio concetto un singolo sistema operativo per una singola applicazione su un solo PC.

Come conseguenza di ciò risulta evidente il ritorno alla centralizzazione della potenza di carico, cioè ad avere maggiore potenza di calcolo all'inteno della sala server che verrà poi assegnata a chi ne necessita.

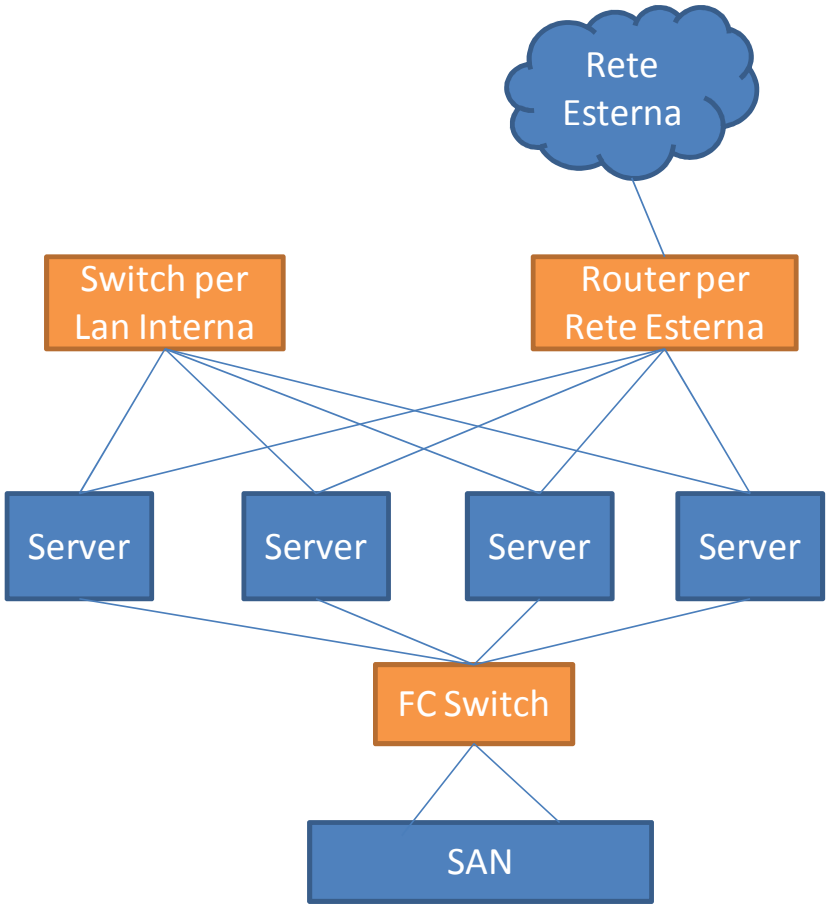
Per tale motivo, è possibile eseguire contemporaneamente diversi sistemi operativi e applicazioni in un singolo computer. La virtualizzazione sostanzialmente consente di trasformare l'hardware in software. Più macchine virtuali condividono le risorse hardware senza interferire tra loro.

La scelta di virtualizzare quasi la totalità dei server presenti in azienda ha posto la necessità di dotarsi di nuovo hardware in grado di garantire il funzionamento di tutti i servizi senza rallentamenti per gli utenti finali. La scelta è caduta su IBM BladeCenter. Il sistema è costituito da due IBM BladeCenter Chassis H all'inteno dei quali risiedono su entrambi due BladeCenter server H22. Ogni server è equipaggiato con doppio processore Intel Xeon E5630 4C a 2.40 GHz, 8 MB di cache, 8 GB di

RDIMM 1066MHz , 1 disco SAS da 250 GB da 15000 giri/min. Le connessioni di rete sono integrate nel Chassis con 8 porte a tecnologia Gbit ethernet.

Affiancato a questo server si è reso necessario dotarsi di una struttura per l'archiviazione dei dati molto veloce e ridondante in caso di guasto. Per soddisfare questa necessità si è ricaduti sull' installazione di una SAN. Grazie alla SAN si viene ad avere un elevato trhoughtput di comunicaizone con i server e in caso di bisogno si attivano i canali ridondanti per garantire la continua accessibilità ai dati.

In questo caso ci si è dotati di IBM System Storage DS3500 Express dotato di 12 HardDisk da 300GB per una spazio totale di archiviazione di 3.4 TB. I dischi sono configurati in raid 5. Sia i server che la SAN sono dotati di unità UPS in grado di mantenere il funzionamento in caso di mancanza di energia elettrica. Tra i server e la SAN si è installato un switch FC, per consentire ai due sistemi di comunicare tra loro. Lo schema seguente rappresenta una versione semplificata di ciò descritto sopra.



Schema della rete

2.2 Configurazione

Installazione e configurazione di Xen Server

L'installazione di XenServer si è svolta regolarmente seguendo la documentazione. Si è proceduto con la formattazione delle macchine fisiche e con l'installazione dell' hypervisor. Successivamente si sono configurati tutti quei servizi per mettere in rete il sistema quali IP, DNS, NTP, SSH. A questo punto tramite un PC di controllo nel quale è stato installato XenCenter si è verificato se il server è visibile da remoto. Tramite l'interfaccia sono stati installati anche gli aggiornamenti disponibili sul sito di Citrix.

Si è creato un template (che può essere considerato il file di configurazione della VM) predisposto per la nostra distribuzione linux, secondo le linee guida. XenServer è compatibile con Debian 6.0 installando solamente la versione multiarch con installazione dalla rete. Anche se un pò limitante come requisito non sono sorti nessun tipo di problemi. Al termine dell'installazione è stata controllata se la VM fosse raggiungibile da XenCenter , si è aggiornato il SO in modo tale da avere una versione totalmente aggiornata del sistema e di seguito configurato Debian 6.0 per essere visibile in rete locale.

Per fare ciò sono stati impostati i file `host` e `interface` con i dati corretti e installati SSH e NTP per rendere il sistema accessibile in modalità sicura dall'esterno e sincronizzarlo con la sua controparte.

Innanzitutto, per non perdere il lavoro fin qui fatto, ma anche per avere delle copie per eseguire dei test utili a sviluppare al meglio il nostro sistema, si è provveduto ad effettuare delle copie di backup.

Il sistema a questo punto è configurato con il seguente schema:

due server fisici nei quali sono installati due XenServer 6.0, sopra ai quali è installata per ogni uno una VM con SO Debian 6.0. Le Virtual Machine sono visibili tra di loro, dall'esterno e da XenCenter. Questo mi consente di procedere con la configurazione tramite l'installazione di MySQL.

Installazione e configurazione di MySQL Cluster 7.1.19

La struttura del cluster è formata da due nodi MySQL, da due nodi Dati, e due nodi di Management.

Sia i due nodi MySQL, sia i due nodi Dati, sono stati configurati su i due server virtualizzati, mentre per i nodi di management si è provveduto ad utilizzare due VM già predisposte alle funzioni di monitoraggio per altre applicazioni. I nodi di management, sono utilizzati entrambi per far partire il cluster la prima volta, successivamente, posso essere spenti entrambi o utilizzati solo quando ci è necessario.

L'istallazione del software MySQL Cluster 7.1.19 è stata svolta seguendo le istruzioni trovate nella documentazione in rete. Ogni computer costituente il cluster deve avere il corretto eseguibile installato. I nodi MySQL: `mysqld`, i nodi Dati: `ndbd`, i nodi di Management: `ndb_mgmd`, `ndb_mgm`. Nel mio caso i file necessari per eseguire i

nodi Dati e MySQL sono installati, in entrambi i server. Terminata l'installazione si è passato alla creazione dei file di configurazione.

Ogni nodo Dati o MySQL necessita di un file `my.cnf`. Il `my.cnf` è stato salvato all'indirizzo `/etc`, dove solitamente sono contenuti i file di configurazione delle applicazioni installate su Debian 6.0.

```
[mysqld]
# Options for mysqld process:
ndbcluster
 # run NDB storage engine
ndb-connectstring=192.168.1.11,192.168.1.12 # location of management server
[mysql_cluster]
# Options for ndbd process:
ndb-connectstring=192.168.1.11,192.168.1.12 # location of management server
```

La dicitura `[mysqld]` serve per indicare che le opzioni successive sono per il server MySQL, mentre la dicitura `[mysql_cluster]` serve per specificare le opzioni dei nodi Dati.

Entrambi i tipi di nodi necessitano di specificare l'indirizzo dei nodi Management. Questo viene fatto settando `l'opzione ndb-connectstring` con l'indirizzo corretto, se i nodi management sono più di uno, deve contenere tutti gli indirizzi.

Come si è visto dalla descrizione del MySQL Cluster utilizza l' NDB engine. Questa specifica viene inserita nelle opzioni del nodo MySQL con l'opzione `ndbcluster`.

Ogni nodo di Management necessita di un file `config.ini`. In questo caso lo si trova in `/var/lib/mysql-cluster`. All'interno si trovano le informazioni per quante repliche mantenere, quanta memoria per dati e indici su ogni nodo allocare, dove trovare i nodi dati, dove salvare i dati sul disco su ogni nodo dati, dove trovare i nodi MySQL.

```
#DEFAULT NODI
[ndbd default]
NoOfReplicas=2
LockPagesInMainMemory=1
DataMemory=512M
IndexMemory=512M
MaxNoOfTables=2048
MaxNoOfOrderedIndexes=2048
MaxNoOfAttributes=2048

# GESTIONE
[ndb_mgmd]
# Management process options:
hostname=192.168.1.11
# Hostname or IP address of MGM node
datadir=/var/lib/mysql-cluster
# Directory for MGM node log files
NodeId=1

[ndb_mgmd]
# Management process options:
hostname=192.168.1.12
# Hostname or IP address of MGM node
datadir=/var/lib/mysql-cluster
# Directory for MGM node log files
NodeId=2
```

```
# NODI

[ndbd]
# Options for data node "A":
hostname=192.168.1.21
# Hostname or IP address
datadir=/usr/local/mysql/data
# Directory for this data node's data files
NodeId=3

[ndbd]
# Options for data node "B":
hostname=192.168.1.22
# Hostname or IP address
datadir=/usr/local/mysql/data
# Directory for this data node's data files
NodeId=4

#SERVER
[mysqld]
# SQL node options:
hostname=192.168.1.21
NodeId=5

[mysqld]
# SQL node options:
hostname=192.168.1.22
NodeId=6
```

Ci sono diverse sezioni con le relative opzioni che si possono utilizzare nel file di configurazione. Di seguito verranno descritte quelle che ho utilizzato.

`[ndbd default]`: Specifica le opzioni comuni a tutti i nodi dati

`NoOfReplica`: Opzione per indicare il numero di repliche in ogni gruppo di nodi. Nel mio caso ho solo un gruppo con i dati dei nodi replicati su entrabi.

`LockPagesInMainMemory`: Serve per bloccare nei sistemi Linux lo swap dei dati su Disco, preservando il degrado delle prestazioni.

`DataMemory`: Quantità di spazio disponibile per salvare i record di tutti i database.

`IndexMemory`: Quantità di spazio disponibile per salvare gli indici per accedere ai record di tutti i db.

`MaxNoOfTables`: Numero massimo di tabelle per database.

`MaxNoOfOrderedIndexes`: Numero massimo di chiavi per database.

`MaxNoOfAttributes`: Numero massi di attributi per database.

`[ndb_mgmd]`: Specifica le opzioni per i nodi di management.

`Hostname`: Indirizzo IP o hostname del nodo di management.

`NodeId`: Id numerico univoco su tutto il cluster per identificare il nodo.

`Datadir`: Directory per i file di log per i nodi management.

`[ndbd]`: Specifica le opzione per i nodi dati.

`Hostname`: Indirizzo IP o hostname del nodo dati.

`Datadir`: Directory per i file dati per i nodi dati.

`NodeId`: Id numerico univoco su tutto il cluster per indentificare il nodo.

`[mysqld]`: Specifica le opzioni per i nodi MySQL

`Hostname`: Indirizzo IP o hostname del nodo dati.

`NodeId`: Id numerico univoco su tutto il cluster per indentificare il nodo.

A questo punto, si è fatto partire per primo il nodi Management tramite il comando

```
ndb_mgmd –f /var/lib/mysql-cluster/config.ini
```

nel quale viene specificato quale file di configurazione caricare.

Come già detto prima i nodi di Management devo essere entrambi funzionanti, quindi è stato lanciato lo stesso comando anche sull'altro nodo.

Su ogni nodo dati si è eseguito il comando:

```
ndbd
```

mentre per ogni nodo MySQL è stato impartito il comando:

```
mysqld_safe &
```

Per testare se il tutto si è avviato correttamente in uno dei nodi managament è stato fatto partire il client per la gestione del cluster tramite:

```
ndb_mgm
```

Tramite il comando `SHOW` viene mostrato la configurazione del cluster:

```
root@gestione1> ./ndb_mgm
-- NDB Cluster -- Management Client --
ndb_mgm> SHOW
Connected to Management Server at: localhost:1186
Cluster Configuration
---------------------
[ndbd(NDB)]     2 node(s)
id=3    @192.168.1.21  (Version: 5.1.16-ndb-7.1.19, Nodegroup: 0, Master)
id=4    @192.168.1.22  (Version: 5.1.19-ndb-7.1.19, Nodegroup: 0)

[ndb_mgmd(MGM)]2 node(s)
id=1    @192.168.1.11  (Version: 5.1.19-ndb-7.1.19)
id=2    @192.168.1.12  (Version: 5.1.19-ndb-7.1.19)

[mysqld(API)]   2 node(s)
id=5    @192.168.1.21  (Version: 5.1.19-ndb-7.1.19)
id=6    @192.168.1.22  (Version: 5.1.19-ndb-7.1.19)
```

Ora il cluster è funzionante e si può iniziare ad introdurre i dati necessari.

Installazione di MySQL replica

E' stata implementata la struttura master-master installando su entrambi i server MySQL 5.5.

Per prima cosa è stato configurato il file `my.cnf` situato in `/etc` in entrambi i server.

```
[mysqld]
log-bin=mysql-bin
server-id=1
innodb_flush_log_at_trx_commit=1
```

```
sync_binlog = 1
replicate-same-server-id = 1
auto-increment-increment = 2
auto-increment-offset = 1
relay-log = /var/lib/mysql/relay-log
max_relay_log_size = 250M
relay-log-index = /var/lib/mysql/relay-log-index
relay-log-info = /var/lib/mysql/relay-log.info
master-info-file = /var/lib/mysql/master.info
log-bin = /var/log/mysql/mysql-bin.log
#relay-log-index = /var/lib/mysql/slave-relay-log.index
expire_logs_days = 10
max_binlog_size = 500M
report-host = master1
```

Come nel caso di MySQL Cluster ci sono varie opzioni, questa volta però è stato configurato solo il server MySQL.

`server-id` : Opzione che identifica univocamente ogni server

`log-bin`: Prefisso del nome del file dove vengono salvate le query che modificano i dati da parte del master per essere inviate allo slave e replicate

`max_binlog_size`: La massima dimensione che un binlog file può raggiungere prima di essere ruotato.

`log-bin-index`: Nome del file dove è salvato il log-bin-index.

`relay-log`: Bin-log ricevuti dal master prima di essere esguiti vengono salvati qui dallo slave

`relay-log-index`,`relay-log-info`,`master-info-file`: In questi file lo slave mantiene le informazioni sullo stato di lettura dei bin-log e di esecuzione dei relay.

`skip-slave-start`: Utile in fase di configurazione, vado a commentarla prima della messa in produzione, evita lo start del processo slave all'avvio del demone MySQL, utilizzato in fase di testing.

`auto-increment-increment` e `auto-increment-offset`: Sono le due opzioni più importanti per il buon funzionamento della replica master-master in caso di presenza di tabelle con campi auto-increment.

`expire_logs_days`: Numero di giorni da mantenere i file binlog

`report-host`: Nome dello slave visualizzato dal master.

Sono stati fatti partire entrambi i server MySQL tramite il comando `mysqld_safe &` e dopo aver configurato l'utente amministratore è stata lanciata la console tramite il comando `mysql`.

Successivamente si è creato l' account per la replica in entrambi i server:

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'replica'@'%' IDENTIFIED BY
'replica'.
```

sostituendo al simbolo % il nome dei due server slave denominati master1 e master2. Per rendere utilizzabili da subito gli account, dalla console di MySQL è stato eseguito il comando `FLUSH PRIVILEGES;`

Si è preso nota della situzione dei bin-long del master1 salvandomi il nome del file e la posizione eseguendo il comando `SHOW MASTER STATUS;`.

```
*************************** 1. row ***************************
            File: mysql-bin.000001
        Position: 4
    Binlog_Do_DB
Binlog_Ignore_DB:
```

Passando alla console del master2 sono stati introdotti i valori di configurazione per collegarlo al master1 come slave.

```
mysql> CHANGE MASTER TO   MASTER_HOST='master1',   MASTER_USER='replica',
MASTER_PASSWORD='replica',   MASTER_PORT=3306,   MASTER_LOG_FILE='mysql-
bin.000001',   MASTER_LOG_POS=4,   MASTER_CONNECT_RETRY=10;
```

Per finire si è fatto partire lo slave del server master2:

```
mysql> START SLAVE;
```

A questo punto per controllare che tutto è andato a buon fine si è visualizzato lo stato dello slave.

```
Slave_IO_State: Waiting for master to send event
                  Master_Host: slave1
                  Master_User: replica
                  Master_Port: 3306
                Connect_Retry: 10
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 4
               Relay_Log_File: master1-relay-bin.000004
                Relay_Log_Pos: 253
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB
          Replicate_Ignore_DB
           Replicate_Do_Table
       Replicate_Ignore_Table
      Replicate_Wild_Do_Table
  Replicate_Wild_Ignore_Table
                   Last_Errno: 0
                   Last_Error
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 4
              Relay_Log_Space: 557
              Until_Condition: None
               Until_Log_File
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File
           Master_SSL_CA_Path
              Master_SSL_Cert
            Master_SSL_Cipher
               Master_SSL_Key
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error
               Last_SQL_Errno: 0
               Last_SQL_Error
  Replicate_Ignore_Server_Ids
             Master_Server_Id: 2
```

Nel caso di errori, vengono segnalati nel campo `Last_IO_Error`.

Stessa cosa viene fatta per configurare il master2 come master e il master1 come slave. Controllando che tutto sia andato a buon fine si può iniziare ad introdurre i dati, che verranno replicati in entrabi i server.

Software Utilizzato

Ecco un elenco del software utilizzato per la realizzazione delle due alternative. Si trattano di soluzioni Open Source disponibili nel periodo di svolgimento del tirocinio. Al momento della messa in produzione del sistema sono state tutte aggiornate all'ultima release, principalmente per motivi di sicurezza e non funzionali.

- Citrix XenServer 6.0
- Citrix XenCenter 6.0
- Debian 6.0 Squeeze
- MySQL Cluster 7.1.19
- MySQL 5.5

2.3 Tecniche di ripristino

Per entrambe le soluzioni adottate si sono studiate delle tecniche per il ripristino del sistema in caso di guasto, in modo tale da avere delle linee guida essenziali ed efficaci. In fase di progettazione mi è stato espressamente richiesto dal settore IT, di non sottovalutare questo aspetto, in quanto, quando avviene un disservizio improvviso, la maggior parte delle volte non si ha il tempo di far intervenire il tecnico specializzato in quel ambito, quindi si necessita di avere della documentazione accessibile da tutto il reparto. Solitamente in questi casi i problemi non sono unici ma spesso collegati, quindi un soluzione repentina del problema è preferibile, lasciano la ricerca delle cause ad una momento successivo.

Soluzione in caso di guasti

La caduta di un nodo può derivare da guasti software e hardware. Due soluzioni sono implementabili per recuperare il funzionamento della replica.

Nel caso un nodo non fosse più raggiungibile per vari motivi, ma comunque al server MySQL non viene apportata nessuna modifica, al riavvio della macchina, in automatico il server MySQL, si aggiorna con la sua controparte ed entrambe le repliche sono correttamente allineate.

Nel caso invece che il nodo caduto necessiti di reinstallare il server MySQL, bisogna procedere come segue:

- Configurare nuovamente il file my.cnf situato in `/etc`.
- Fermare lo slave nel master ancora in piedi e bloccare le scritture nei database.

```
mysql>STOP SLAVE;
mysql>FLUSH TABLE WITH READ LOCK;
```

- Creare il dump del database, recuperare il nome del file e l'indice del binlog.

```
shell> mysqldump --all-databases > dump.sql
mysql> SHOW MASTER STATUS;
```

- Riconsentire le scritture nel master.

```
mysql>UNLOCK TABLES;
```

Si può ora procedere con l'inserimento del dump nel nuovo master,

```
shell> mysql < dump.sql
```

e con la creazione dell'utente per accedere allo slave.

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'replica'@'%' IDENTIFIED BY
'replica'.
```

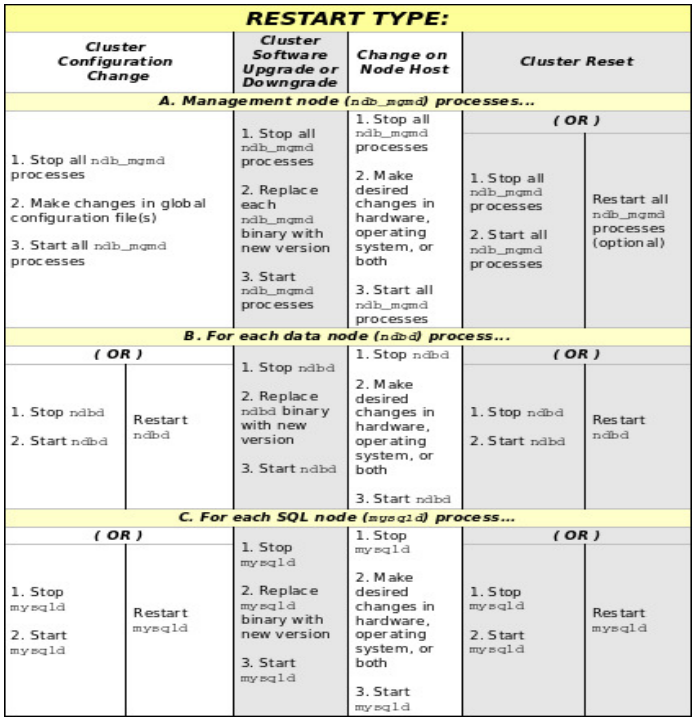
In entrabi i server per configurare correttamente gli slave è necessario reintrodurre i parametri corretti del nome del file di bin-log e l'indice del bin-log.

```
mysql> CHANGE MASTER TO   MASTER_HOST='master1',   MASTER_USER='replica',
MASTER_PASSWORD='replica',   MASTER_PORT=3306,   MASTER_LOG_FILE='MySQL-
bin.000001',   MASTER_LOG_POS=4,   MASTER_CONNECT_RETRY=10;
mysql> START SLAVE;
```

A questo punto entrambi i server sono di nuovo in replica e funzionanti. Eventuali modifiche apportate dopo il dump verranno automaticamente riversate nel nuovo server in replica.

Rolling Restart

MySQL Cluster mette a disposizione il Rolling Restart del Cluster. In pratica consiste nel bloccare e far partire o ripartire ogni nodo a turno, consentendo così al cluster di rimanere sempre operativo. L'operazione di Rolling Restart è utile nel caso la configurazione del cluster cambi, per fare upgrade o downgrade del sistema, per apportare modifiche all' hardware, per fare il reset del sistema. La seguente immagine presa dal sito MySQL riassume il Rolling Restart di ogni tipo di nodo, nei vari casi.



Rolling Restart

2.4 Alternative di realizzazione

Nell'ambito del percorso di studio della soluzione da adottare ci si scontra sempre tra le richieste fatte in ambito di definizione del progetto e le risorse disponibili per la realizzazione definitiva dello stesso. Anche nel mio caso all'inizio ci si è indirizzati verso una strada, ma come vedremo qui di seguito ho dovuto abbandonarla per percorrerne un'altra.

Entrambe le alternativa da me adottate consentono la ridondanza dei dati e l'alta affidabilità di questi.

MySQL Cluster è un prodotto che garantisce la disponibilità di accesso ai dati con un affidabilità quasi pari al 100%, recentemente sono state aggiornate e rese più performante le richieste al database, consentendo una risposta 70 volte più veloce rispetto alla versione precedente, ma per consentire e garantire queste performace così elevate, MySQL Cluster mantiene totalmente i dati nella memoria RAM, MySQL Cluster è così definito un DB in Memory. Ovviamente una copia dei dati è scritta su disco fisso nel caso si abbia la necessità di riavviare la macchina.

Per configurare correttamente i nodi del cluster si utilizza uno script in Perl, che si trova nel pacchetto di MySQL, al quale si da in ingresso il dump di tutto il DB che noi andremo a caricare nel cluster e ci da come risultato la memoria necessaria su ogni nodo. Nel mio caso con un db di circa 2 Gbyte il risultato è il seguente:

```
Parameter minimun requirements
* indicates greater than default
```

Parameter	Default	4.1	5.0	5.1
DataMemory (KB)	81920	6856000*	6856000*	6223232*
NoOfOrderedIndexes	128	21544*	21544*	21544*
NoOfTables	128	24417*	24417*	24417*
IndexMemory (KB)	18432	1405208*	468832*	468832*
NoOfUniqueHashIndexes	64	13740*	13740*	13740*
NoOfAttributes	1000	100437*	100437*	100437*
NoOfTriggers	768	160432*	160432*	160432*

Come si vede dalla tabella la memoria necessaria per i dati è di almeno 6.856.000 KB, mentre per gli indici è di almeno 1.405.208 KB. Sommando entrabe risultano necessari per ogni nodo almeno 8.5 GB. Discutendo con il responsabile, siamo giunti alla conclusione che il parametro minimo obbligatorio è troppo elevato per il funzionamento dei nodi, prevedendo un carico di dati sempre maggiore nel futuro Inoltre la memoria RAM la si può utilizzare per far funzionare al meglio altri servizi e non relegarla ad un contenitore di dati.

Abbandonando questa soluzione sono state implementate le funzioni di replica di MySQL, le quali ci consentono:

- letture e scritture concorrenti.
- possibilità di scalare la soluzione nel caso di aumento del traffico.
- continuità del servizio in caso di errore.

Per riversare i dati vecchi all'interno della nuova struttura è stato necessario fare alcuni passaggi tramite varie versioni di MySQL, in modo da renderli compatibili con la nuova versione implementata. La versione precedentemente usata era 5.0, per passare alla versione 5.5, sono stati necessari due passaggi intermedi di aggiornamento tramite il comando `mysql_upgrade`. Il primo per trasportare i dati dalla versione 5.0 alla 5.1 e poi dalla 5.1 alla 5.5. MySQL non dà la possibilità di passare direttamente all'ultima versione per motivi di incompatibilità. Si è dovuto installare quindi la versione 5.1 e poi da questa ricavare il dump per trasferire i dati definitivamente.

Per ogni passaggio si è inserito i dati nel DB e successivamente eseguito il comando `mysql_upgrade`, a questo punto si è riavviato il server per rendere effettive le modifiche. Il comando `mysql_upgrade` è necessario in quanto esamina tutte le tabelle del DB per verificare l'incompatibilità, nel caso ci fosse bisogno le corregge e consente di aggiornare il system table per aggiungere nuovi privilegi e funzionalità. La soluzione finale perciò è ricaduta sulle funzionalità di replica date da MySQL.

2.5 Considerazioni finali

L'obiettivo del progetto è stato quello di creare un sistema che gestisca la ridondanza dei dati integrandolo con la nuova infrastruttura di rete del provider.

Si sono utilizzati principalmente software Open Source dimostrando così la possibilità di realizzare soluzioni enterprise pur mantenendo un spesa contenuta. Al di là della riduzione dei costi questi tipi di software ormai non hanno niente da invidiare con i prodotti commerciali, anzi consentono di rendere ancora più personalizzate le soluzioni adottate.

Lo svantaggio derivato dall'utilizzo di questi tipi di software è dovuto soprattutto dal non avere un supporto ufficiale in caso di problemi. Svantaggio che però aiuta a plasmare una mentalità dedita al problem solving. Questa capacità diventa utile soprattutto in caso di criticità improvvise permettendoci di affrotare i problemi in maniera efficace. Inoltre si può sfruttare la notevole quantità di documentazione messa a disposizione e l'aiuto dato dalle community di rete come i forum, all'interno dei quali sono presenti esperti del settore.

L'applicazione è stata realizzata sempre a diretto contatto con i tecnici IT, con loro si è discusso di come implementare il nuovo servizio con le prassi aziendali. In questo ambito l'attività di stage è stata molto formativa, in quanto ho potuto verificare l'importanza di un'attenta pianificazione, da svolgersi nelle fasi iniziali del progetto, ancora prima di iniziare la realizzazione vera e propria, mi ha permesso di testare ampiamente le varire soluzioni considerate nel svolgimento del tircocinio così da avere un ampia panoramica dei problemi o errori che si possono verificare, a margine di ciò si è lasciata all'interno dell'azienda un documento contentente le linee guida per la risoluzione di eventuali problemi.

L'esperienza provata mi ha permesso di conoscere in modo abbastanza approfondito il database MySQL, il software MySQL Cluster e il software di virtualizzazione XenServer. Questi ultimi due prodotti sono relativamente recenti, ma stanno raccogliendo sempre più consensi, grazie alla continua evoluzione e al prezzo contenuto. Un numero sempre maggiore di aziende sta introducendo all'interno della propria azienda queste tecnologie e di consegnuenza più figure professionali con esperienza in questo campo sono richieste.

La soluzione adottata anche se funzionante e messa in produzione è in grado di implementare sviluppi futuri. Oltre alla neccessità degli aggiornamenti software del caso che vengono via via rilasciati dalle aziende produttici, si potrà in futuro implementare funzioni migliorate per il back-up ad esempio inserendo una macchina slave dedita a questo compito, nel caso il traffico aumentasse notevolmente, dedicare delle istanze di mysql solo alla lettura dei dati scalando così il lavoro dei master.

Dando un giudizio a questo esperienza la posso sicuramente definire positiva ma soprattutto utile. Sono venuto a conoscenze di molti aspetti tecnici e mi ha posto a confronto con problemi concreti. Il tirocinio inoltre, mi ha consentito di approfondire diverse tematiche, per la maggior parte viste in forma teorica durante il corso degli studi.

Bibliografia

[1] Storia aziendale tratta da
http://www.terralink.it/
[2] Informazioni sulla virtualizzazione tratte da
http://it.wikipedia.org/wiki/Virtualizzazione
http://www.vmware.com/
[3] Informazioni e documentazione XenServer tratte da
http://www.citrix.com/
[4] Informazioni MySQL tratte da
http://www.mysql.com/
http://it.wikipedia.org/wiki/MySQL
[5] Informazioni e documentazione MySQL Cluster tratte da
http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster.html
[6] Documentazione MySQL 5.5 tratte da
http://dev.mysql.com/doc/refman/5.5/en/replication.html