UNIVERSITÀ DEGLI STUDI DI CATANIA
FACOLTÀ DI INGEGNERIA
CORSO DI LAUREA IN INGEGNERIA INFORMATICA
DIPARTIMENTO DI INGEGNERIA ELETTRICA, ELETTRONICA E INFORMATICA

CARMELA GRECO

SYSTEM RECOVERY E AFFIDABILITÀ DI DATI NELLA PRODUZIONE E USO DELL'ENERGIA NUCLEARE

TESI DI LAUREA

Relatore:
Prof.ssa Ing. Vincenza Carchiolo

Correlatori:
Prof. Paolo Finocchiaro
Dott. Ing. Gianfranco Vecchio

ANNO ACCADEMICO 2010/2011

INDICE

RINGRAZIAMENTI

Di una cosa sono certa: l'impegno nello studio, la costanza e la pazienza che ho messo nell'affrontare il mio percorso universitario, non sarebbero stati sufficienti se in questi anni non avessi avuto al mio fianco delle persone speciali.

Ringrazio la mia famiglia: papà, mamma e Graziella, per aver condiviso con me non solo i momenti di gioia, ma soprattutto quelli di difficoltà, senza mai stancarsi di incoraggiarmi.

Ringrazio di cuore Salvo per essere stato sempre con me e per aver creduto in me, molto più di quanto non ci credessi io.

Ringrazio i miei amici più cari e i miei "colleghi", compagni di viaggio la cui presenza ha reso migliore il mio cammino fino ad oggi.

Ringrazio infine il Prof. Paolo Finocchiaro e l'Ing. Gianfranco Vecchio del gruppo di ricerca dell'INFN e la prof. Vincenza Carchiolo per l'aiuto datomi per lo svolgimento del lavoro di tesi.

CAPITOLO 1: Introduzione

La presente tesi di laurea, svolta a Catania presso i Laboratori Nazionali del Sud dell'Istituto Nazionale di Fisica Nucleare, ha come oggetto la progettazione e la realizzazione di un High Availability cluster a due nodi per l'affidabilità dei dati e il System Recovery di una applicazione php web-based realizzata nell'ambito del progetto DMNR (Detector Mesh for Nuclear Repository). In seguito all'esigenza di tutelare la continuità dell'applicazione web sviluppata dal team di lavoro si è pensato di realizzare un clustering ad alta affidabilità. L'applicazione ha come scopo principale quello del monitoraggio in tempo reale dei depositi di scorie radioattive. Questa è la missione critica la cui sopravvivenza a possibili interruzioni deve essere garantita, salvaguardando i dati e i servizi di cui fa uso. Infatti è proprio attraverso l'elaborazione continua dei dati provenienti dal deposito sotto forma di file di log elaborati dall'elettronica di front-end che l'applicazione riesce a evidenziare particolari situazioni di allarme.

Gli altri servizi, non critici, forniti dall'applicazione permettono di interagire con il database che ne contiene lo storico; permettono di inserire, rimuovere o sostituire dati relativi allo stato attuale del deposito. Inoltre l'applicazione contiene delle sezioni informative che mostrano l'organizzazione gerarchica dell'impianto monitorato e altre informazioni divulgative sul progetto DMNR.

L'applicazione è stata pensata per l'utilizzo da parte di utenti autorizzati a consultarne i contenuti, come ad esempio il personale di un sito di gestione di scorie radioattive, cui viene garantito un servizio in tempo reale in grado di dare una visione continua dello stato dell'impianto.

Segue una panoramica del lavoro svolto nella tesi:

CAPITOLO 2: In questo capitolo viene presentato e approfondito il concetto di "Business Continuity", il quale costituisce il punto di partenza per i progetti di System Recovery di qualunque entità. Dopo aver dato una serie di definizioni per chiarirne il significato vengono presentati gli obiettivi che la Business Continuity si prefigge e i vari step che permettono di giungervi.

CAPITOLO 3: Il terzo capitolo affronta il problema del Disaster Recovery, visto come una delle possibili soluzioni alla Business Continuity. Viene sottolineata la differenza che intercorre tra i due concetti e vengono discussi nei dettagli: le alternative, la tecnologia, le fasi di progetto e i costi di un piano di Disaster Recovery.

CAPITOLO 4: Il capitolo quattro introduce la soluzione di clustering realizzata in questo lavoro di tesi. Dopo una breve descrizione delle caratteristiche principali di una struttura cluster generica, viene presentato in dettaglio la soluzione scelta nel caso di studio: un High Availability cluster.

CAPITOLO 5: In questo capitolo il concetto di Alta Affidabilità, viene correlato ai concetti di Business Continuity e Disaster Recovery affrontati ai capitoli 2 e 3, dandone una breve definizione. In seguito vengono discussi i requisiti necessari per la sua realizzazione.
Tale capitolo è utile per una comprensione approfondita del caso di studio presentato al capitolo successivo.

CAPITOLO 6: E' il capitolo in cui viene esposto il caso di studio. Comprende una parte di introduzione in cui oltre alla presentazione del progetto DMNR si spiegano le motivazioni che hanno portato all'adozione della soluzione scelta. Segue una presentazione del materiale software e hardware utilizzato per la realizzazione fisica del progetto, con le relative configurazioni affrontate in dettaglio.

CAPITOLO 7: La conclusione di questo lavoro di tesi comprende una valutazione del lavoro svolto, i possibili miglioramenti applicabili alla soluzione adottata e i possibili sviluppi futuri.

CAPITOLO 2: Business Continuity

2.1 DEFINIZIONE

I termini "Business Continuity" indicano quell'insieme di processi e procedure che permettono ad un'organizzazione di assicurare la continuità della propria attività anche durante e dopo un disastro.
Una definizione formale di Business Continuity è la seguente:

"Il Business Continuity Planning promuove la capacità di un'organizzazione di salvaguardare sistemi critici di business, in caso di eventi disastrosi, che impattano sulle normali operazioni di processamento dati. La pianificazione include la preparazione, il testing e la manutenzione di azioni specifiche, al fine di proteggere i processi critici di business dagli effetti di guasti estesi sui servizi di processamento dei dati."[1]

Un ulteriore chiarimento su questo concetto si può avere illustrando con un'altra breve definizione la pianificazione della Business Continuity, che verrà approfondita in seguito:

"Il Business Continuity Planning indica quanto propriamente una organizzazione si prepara a sopravvivere a disastri, interruzioni o cambiamenti inattesi, assicurandosi che i processi critici del business continuino a funzionare in circostanze avverse con limitazioni accettabili."[2]

Il NIST - National Institute of Standards and Technology, l'ISC - International Information Systems Security Certification Consortium e il Business Continuity Institute sono alcune delle organizzazioni cui le aziende di tutte il mondo intenzionate ad attuare la continuità operativa possono rivolgersi. In Italia si possono affidare ad IMQ - Istituto Italiano per il Marchio di Qualità, il più importante ente di certificazione italiano, che definisce la Business Continuity come segue:

"Per gestione della continuità operativa o continuità aziendale (business continuity) si intende la capacità dell'azienda di continuare ad esercitare il proprio business a fronte di eventi avversi che possono colpirla."[3]

Si tratta di organizzazioni riconosciute, a livello mondiale, da oltre 120 nazioni, le quali hanno stabilito una serie di standard per la protezione dei siti di lavoro.
Sempre in merito alla tematica della Business Continuity, lo standard ISO/IEC 27001 definisce le specifiche per il sistema di gestione della sicurezza delle informazioni (ISMS) nella gestione della continuità operativa.
Negli ultimi anni molte aziende o università investono nella ricerca in questo campo, dal momento che se l'azienda in questione è molto grande le perdite che si possono avere per una interruzione dei servizi anche limitata a poche ore può causare danni economici gravissimi, che in alcuni casi potrebbero portare l'azienda anche al fallimento.
La Business Continuity presenta due aspetti principali: un aspetto preventivo e uno reattivo. L'obiettivo primario è la prevenzione dell'interruzione dei servizi. Qualora nonostante quest'ultima si verifichi una interruzione, si procede alla risposta

reattiva che consiste nel cercare di ristabilire nel minor tempo possibile il pieno funzionamento del sistema.
È chiaro che la Business Continuity cura per buona parte gli aspetti tecnici, poiché se è necessario si deve procedere al recovery delle infrastrutture, dei dati e dell'intero sistema fisico. Inoltre in quest'ultimo caso si devono considerare fattori non trascurabili quali il tempo e i costi.
Poiché il costo è uno dei fattori determinanti, dal momento che la realizzazione di un Business Continuity Plan comporta dei notevoli investimenti sul piano economico, è importante che il punto di partenza dello sviluppo di un piano di continuità del business sia la cosiddetta "Analisi dei rischi". L'analisi dei rischi prevede che siano valutate le minacce che effettivamente possono intaccare il sistema e che siano esaminati tutti i "point of failure" del sistema. L'analisi dei rischi è solitamente seguita dalla cosiddetta "Analisi degli impatti", che analizza in che modo le minacce individuate potrebbero danneggiare il sistema.
Le due componenti di analisi presentate costituiscono il "Risk Management", operazione preliminare fondamentale per pianificare una soluzione concreta per la Business Continuity.
Quando si elencano tutte le possibili conseguenze che si possono avere in seguito ad un disastro, si deve considerare anche il "worst case", quello in cui tutti i dati vengono colpiti con la possibilità di avere delle perdite totali permanenti. In questo scenario viene adottata la soluzione del "Disaster Recovery" (Capitolo 3). Si ricorre al Disaster Recovery nel momento in cui non si è più in grado di garantire la continuità, poiché l'intero sistema è *in down*. L'esigenza primaria diventa quella di riattivare tutte le attività del business tramite le procedure descritte nel Business Continuity Plan e le infrastrutture tipiche dei sistemi di Disaster Recovery. Questa fase si chiama “incident

handling" o "crisis management", quando avviene una interruzione l'organizzazione deve sapere immediatamente come gestire la situazione.
Le possibili cause di un disastro sono molteplici e vengono discusse nel prossimo capitolo.

2.2 GLI OBIETTIVI

Dopo aver introdotto in linea generale il concetto di Business Continuity vengono ora presentati gli obbiettivi specifici che un'azienda vuole perseguire nel momento in cui decide di attuare un piano di continuità:

- rispondere in maniera adeguata, immediata e mirata ad una emergenza
- fornire un metodo collaudato per il ripristino della capacità di erogare servizi critici ad un livello predefinito ed entro un lasso di tempo prestabilito a seguito di un'interruzione
- minimizzare le perdite in caso di disastro
- fornire le procedure dettagliate e l'elenco di tutte le risorse utili al ripristino delle funzioni di processamento dei dati che stanno alla base del business delle applicazioni IT
- raccogliere le procedure in documenti che siano chiari e definiti, resi disponibili per il personale che può così avere una guida sulle azioni da svolgere
- documentare le procedure per la memorizzazione, la salvaguardia e il recupero dei dati
- fornire le linee guida di comportamento da portare avanti durante il guasto, al fine di evitare una risposta confusa e controproducente all'evento
- descrivere le azioni da compiere, le risorse da impiegare e i materiali indispensabili per riportare le operazioni critiche

su un sito alternativo, nel caso che il sito primario abbia subito un guasto per un tempo prolungato;

- riparare o rimpiazzare le facility danneggiate entro un arco di tempo estremamente breve.
- consentire una chiara comprensione di come funziona l'intera organizzazione e permettere d'identificare opportunità di miglioramento.

2.3 STEP DEL PROGETTO DELLA BUSINESS CONTINUITY

Il progetto di Business Continuity è basato su quattro step fondamentali:

1) knowledge del business
2) analisi dei rischi
3) formulazione del piano ed attuazione del piano
4) testing del piano

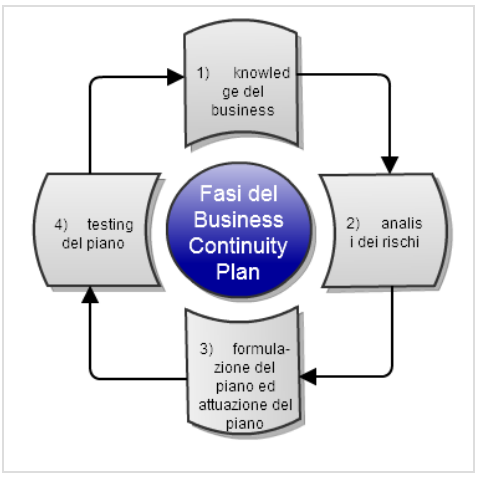

Figura 1: Fasi del Business Continuity Plan

I primi due step servono a definire una pianificazione duplice: preventiva e reattiva. L'aspetto della prevenzione ha l'obbiettivo di individuare le attività da svolgere prima che il disastro colpisca il sistema che si vuole proteggere. Supponendo che il sistema da

proteggere sia un sito web, l'attività di prevenzione potrebbe consistere nel predisporre un server secondario che sostituisca il primario quando questo cade.

L'aspetto reattivo invece specifica le attività che devono essere messe in atto dopo che il disastro è avvenuto. A differenza dell'attività di prevenzione, nel caso ipotetico precedente del sito internet permanentemente oppure per lungo tempo non disponibile, nell'attività di reazione si pianificano le azioni concrete da svolgere per trasferire il servizio dal server d'origine a quello secondario.

La formulazione del piano è sempre presente, poiché la prevenzione è una azione continua, mentre l'attuazione vera a propria è messa in atto solo a seguito del disastro.

Il testing viene effettuato non solo sull'attuazione delle attività ma anche sul piano stesso. La revisione continua del piano è infatti essenziale per garantire l'efficacia della Business Continuity che si deve adeguare di volte in volta alle esigenze dell'azienda e deve essere modificata se dai test si evince che su certi punti è possibile un miglioramento.

Bisogna pianificare e coordinare delle routine di test periodiche. Diversi livelli di test come falsi allarmi, simulazioni, walk-through test dovrebbero essere condotti per assicurarsi che il personale coinvolto sappia cosa fare in modo da mantenere la Business Continuity.

Tutto il processo di pianificazione appena illustrato produce in output un documento di estrema importanza: il Business Continuity Plan, fulcro di qualunque sistema di IT-continuity che consente la salvaguardia degli aspetti finanziari di una azienda, della sua immagine e del rispetto delle normative.

Affinché un documento di BCP si possa ritenere utile e totalmente comprensibile deve seguire delle linee guida e degli standard prestabiliti. Alcuni di questi standard sono:

- ISO-22399: Incident Management & Business Continuity
- MS 1970: Business Continuity standard inMalaysia
- HB 221: Business Continuity standard in Australia
- TR 19: Business Continuity Reference Singapore
- NFPA 1600: Disaster Recovery & BC standard (National Fire Protection Association USA)

La presentazione della Business Continuity trattata in questo capitolo è molto generale, essendo un argomento molto vasto la sua trattazione richiederebbe ulteriori approfondimenti che andrebbero oltre la tematica principale oggetto di questa tesi che è il System Recovery. Il System Recovery è solo una delle possibili soluzioni per la continuità del business e verrà affrontato nel dettaglio nel prossimo capitolo.

CAPITOLO 3: Disaster Recovery

3.1 Disaster Recovery: una soluzione per la Business Continuity

3.1.1 Definizione

Viene definito disastro un evento che spegne un sistema informatico per più di qualche minuto. Talvolta questa situazione può permanere per molte ore o addirittura per settimane. Il Disaster Recovery è quel processo messo in atto da una compagnia per riportare in vita i suoi computer e le sue applicazioni dopo una interruzione dei servizi in larga scala.[4]

Il Disaster Recovery è implementato da un'organizzazione quando la sua attività poggia prevalentemente su dati, applicazioni e infrastrutture la cui perdita può significare la perdita dell'intero business. È necessario che queste informazioni vengano salvaguardate tramite l'impiego di strutture alternative che ne garantiscono la sopravvivenza ad un disastro. Segue un grafico che illustra le principali cause di disastro, la maggior parte delle volte esso è dovuto ad errori umani e a malfunzionamenti hardware e software.

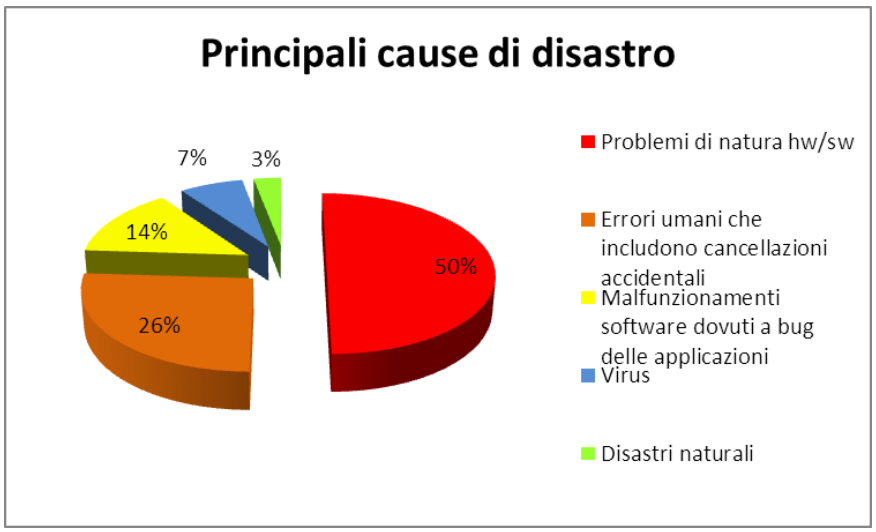

Figura 2: Principali cause di un disastro

3.1.2 Differenza tra Business Continuity e Disaster Recovery

Sia la Business Contnuity sia che il Disaster Recovery fanno parte del "dominio dell'emergenza" di una organizzazione, ma la Business Continuty ha come obbiettivo principale quello di evitare l'interruzione del business in generale e prende quindi in considerazione tutti gli eventi indesiderati e tutte le risorse a sostegno dei processi aziendali che potrebbero essere compromesse. Il Disaster Recovery Planning invece riguarda solo l'eventualità di un disastro, che se da un lato è raro che si verifichi, dall'altro se dovesse accadere ha un altissimo impatto sul business. In poche parole il Disaster Recovery Plan si colloca all'interno della Business Continuity e considera solo l'IT dell'azienda, cioè il suo sistema informatico e tutte le risorse ad esso relative (le persone che lo gestiscono, l'organizzazione, le politiche e le procedure correlate, i siti ove è collocato).

Si potrebbe quasi dire che l'obiettivo ultimo di un progetto di Business Continuity sia la definizione di un Disaster Recovery Plan, che dunque rappresenta l'insieme di quegli adempimenti di tipo fisico, logico, organizzativo, amministrativo, logistico e legale atti a fronteggiare un evento a carattere catastrofico, che renda indisponibili le risorse deputate alle operazioni di elaborazione dei dati. Il Disaster Recovery Planning è in sostanza un "processo" che consente di ripristinare il normale o indispensabile funzionamento dell'operatività aziendale e il trattamento dei dati precedentemente interrotti da un evento indesiderato di natura eccezionale e cioè da quello che può definirsi quale vero e proprio "disastro" in un sistema informatico automatizzato.[5]

3.1.3 ALTERNATIVE DI RECOVERY

Alcune delle possibili alternative di recovery possono includere:[6]

- ***Hot Site:*** Una facility alternativa totalmente equipaggiata per recuperare le funzioni di business attaccate da un disastro

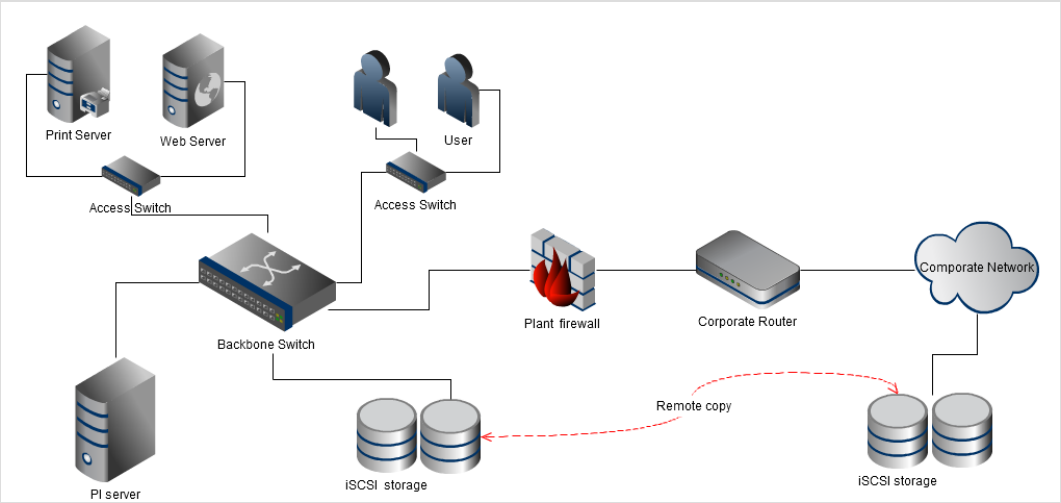

Figura 3: Un esempio di Hot Site

- ***Warm Site:*** Una facility alternativa solo parzialmente equipaggiata per il recupero del business
- ***Off-site:*** Una facility off-site che ospita un completo backup dotato di un sistema integrato usato per automatizzare, monitorare e controllare i servizi
- ***Fault Tolerance:*** Vengono considerate diverse strategie di Fault Tolerance come il mirroring e il clustering.
- ***Una combinazione delle alternative appena presentate***.

3.1.4 DISASTER RECOVERY TECHNOLOGY

Nella pratica le possibili soluzioni usate per realizzare il Disaster Recovery, inteso come trasferimento dati al sito secondario e la conseguente attivazione dei servizi, si possono suddividere in due grandi categorie:[7]

- ***technology based:*** vari tipi di cluster che automaticamente trasferiscono dati e fanno lo switch tra sito secondario e primario
- ***organizational***: procedure manuali che consistono nel recuperare i dati da backup esistenti e attivare con l'intervento umano il sito secondario

3.1.5 RPO ED RTO: PARAMETRI DI TEMPO

Tra tutti gli aspetti del Disaster Recovery Planning, quello più legato alla tecnologia utilizzata è il tempo. Vengono definiti in particolare due parametri di tempo:

- ***Recovery Point Objective (RPO):*** Il momento dal quale i dati devono essere ripristinati in modo da riportare in vita le funzioni del business
- ***Recovery Time Objective (RTO):*** Il periodo di tempo che occorre per il recovery, ad esempio potrebbe essere il periodo che intercorre tra il disastro e l'attivazione del sito secondario.

Il significato dei due tempi viene illustrato in *figura 4.*

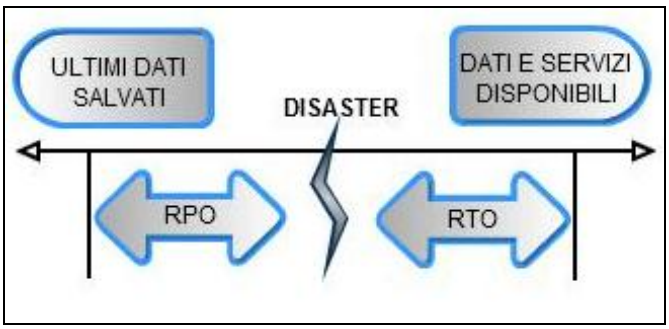

Figura 4: Parametri di tempo RPO ed RTO

Tuttavia il valore di RPO e RTO non può essere stabilito in modo esatto, per determinarli con un margine di errore trascurabile bisogna fare un accurato studio che consideri oltre al tempo anche la grandezza dell'area interessata dal disastro.

La vulnerabilità del business e dei processi tecnologici ad una interruzione viene descritta correlando l'impatto di tale interruzione alla lunghezza del periodo, riflettendo così sia la lunghezza del periodo dell'interruzione (RTO) che il tempo in cui si ha perdita di dati (RPO). Non è facile conoscere l'andamento esatto di queste funzioni che approssimativamente per molti sistemi è quello mostrato in *figura 5*.

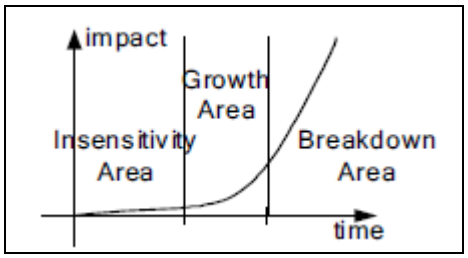

Figura 5: Grafico TIME-IMPACT

L'"Insensitivity Area" è la zona in cui le interruzioni che sono appena percepibili agli user poiché non causano problemi ai processi del business.

La "Growth Area" riflette le situazioni in cui i processi critici sono tenuti in vita senza particolari difficoltà, ma cominciano ad apparire effetti negativi sia dal punto di vista delle applicazioni che dal punto di vista delle perdite finanziarie.

Nella "Breakdown Area" le conseguenze di una interruzione sono significative e intaccano la continuità dei processi. I valori di RTO e RPO richiesti rimangono definiti nella Growth Area, la cui larghezza determina la precisione con cui i due parametri possono essere definiti, poiché prima e dopo di essa i valori di tempo non sono suscettibili alla misura perché troppo piccoli o troppo grandi.[7]

3.2 FASI DEL DISASTER RECOVERY PLAN

Segue un modello suggerito da IBM per la pianificazione del Disaster Recovery:

1. Project Initiation and Team Selection: La fase di Project Initiation coinvolge la formalizzazione e l'approvazione del progetto, la selezione di un coordinatore e la selezione del team, la standardizzazione dei metodi di raccolta delle informazioni e la formulazione di una lista delle risorse utili per il progetto.

2. Data Collection and Critical Needs: La fase di collezione dei dati comporta la raccolta delle informazioni sui processi del business, sui supporti tecnologici, sui potenziali costi dovuti alle interruzioni, sulle esposizioni ai disastri e sulle tecniche e strategie applicate dalla compagnia per mitigare i rischi.

3. Risk Analysis: L'analisi dei rischi convoglia i dati raccolti alla fase 3 agli obbiettivi del Recovery Plan. In pratica questo tipo di analisi porta alla comprensione di cosa è a rischio e quali risorse sono richieste per il ripristino in tempi accettabili. Uno dei risultati di questa analisi potrebbe essere l'implementazione di tecnologie di disaster avoidance, che potrebbero aiutare nella prevenzione dei disastri, come ad esempio: impianti di rilevazione di fumo o acqua, alimentatori supplementari di corrente, controlli di accesso ai servizi erogati e così via.

4. Data Protection: La protezione dei dati è di vitale importanza per il disaster recovery, deve essere formalizzato ogni scelta riguardante i supporti su cui si decide di salvare i dati.

5. Recovery Plans: La fase di Recovery Plans coinvolge la formulazione di strategie di sostituzione dei sistemi e delle relative reti nel caso di una interruzione inaspettata.

6. Training e Plan Testing: Il Training e il Plan Testing valida le strategie finora discusse e ne valuta i punti deboli che possono essere migliorati.

7. Change Management: Il Change Management fornisce un meccanismo per aggiornare il piano a seconda delle esigenze attuali del business.

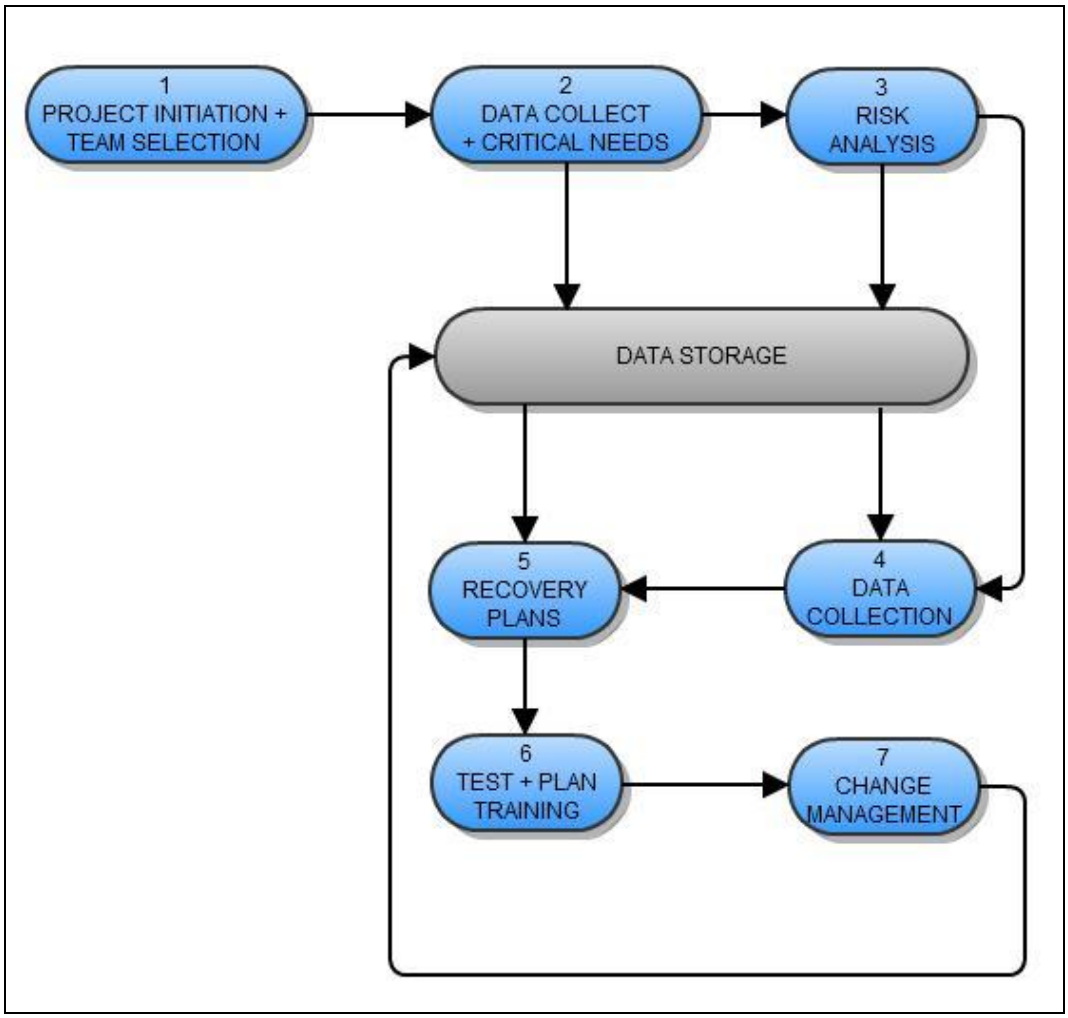

Figura 6: Fasi del Disaster Recovery Plan

3.3 Livelli di Disaster Recovery

Il Disaster Recovery può essere affrontato a diversi livelli, dal più basso al più alto, ad un aumento di prestazioni coincide un aumento dei costi e delle risorse da impiegare.

Vengono definiti 7 livelli:[8]

LIVELLO 0: Nessun Off-site Data

In poche parole non esiste alcun piano di Disaster Recovery, il tempo di ripristino diventa dunque imprevedibile e non si ha la certezza che il ripristino stesso sia possibile. Non esiste una locazione secondaria off-site cui delegare i servizi in caso di down della primaria. In questo modo è chiaro che l'organizzazione ha già preso una decisione in merito a ciò che potrebbe accadere in seguito ad un disastro e in tale decisione è contemplato il fatto che la perdita economica che si avrebbe con una interruzione del business è meno importante di quella che si avrebbe investendo in un ripristino dello stesso.[10]

LIVELLO 1: Data Backup senza Hot Site

Le organizzazioni che usano le soluzioni al Livello 1 fanno un backup dei loro dati su una facility off-site. Il ripristino quindi dipenderà da quanto spesso vengono fatti i backup, infatti per backup con periodicità ridotta basteranno probabilmente pochi giorni per riattivare i servizi. In ogni caso tutti i dati o buona parte di essi sono al sicuro su un sito secondario, nel quale però non sono predisposti i sistemi e le facilities utili al ripristino. Quindi in questo livello vi è sempre un ritardo nei tempi dovuto a quanto si impiegherà affinché i sistemi collaterali al sistema di data storage siano acquistati e resi disponibili.

LIVELLO 2: Data Backup con Hot Site

Le compagnie che adottano il Livello 2 fanno anch'esse regolari backup, ma a questi sono combinate delle infrastrutture off-site pronte all'uso nel caso di un evento avverso. A questo livello, nonostante siano richieste ancora molte ore per il ripristino, è già possibile fare delle previsioni abbastanza precise sui tempi di riattivazione del business.

LIVELLO 3: Electronic Vaulting

Il Livello 3 implementa il 2 e vi aggiunge l'Electronic Vaulting. Si tratta del più veloce processo di recovery attualmente disponibile negli ambienti IT. Il Vaulting è la duplicazione del supporto di backup. Tale tecnica non raddoppia i tempi del processo, ma raddoppia solo il numero dei supporti. Con questo tipo di tecnologia è molto conveniente mantenere una copia dei supporti off-site rispetto alla sede che li genera.[11]

LIVELLO 4: Copie Point-in-time

Le soluzioni al Livello 4 comportano un più ampio utilizzo di soluzioni disk-based permettendo di fare delle copie PIT con più frequenza, data la maggiore capienza dei dispositivi di storage.

Livello 5: Transaction Integrity

Nel Livello 5 c'è una perdita di dati quasi pari a zero, poiché si cerca di mantenere sempre la coerenza dei dati tra sito primario e siti secondari.

Livello 6: Zero or Little Data Loss

Il Livello 6 è adottato solitamente dalle organizzazioni che hanno una tolleranza minima o zero alle perdite dei dati e che per questo motivo richiedono che i tempi di recovery siano minimi.
In esso sono riassunti i livelli 4 e 5, ma a differenza di questi ultimi aumentano i costi e la complessità, poiché è richiesta una capacità di switching automatico tra i due nodi.

Livello 7: Highly Automated, Business Integrated Solution

Il livello 7 è il più alto della scala e ovviamente include tutti gli altri livelli, in più cerca di rendere i processi di ripristino automatici diminuendo ancora di più i tempi di recovery. [9]

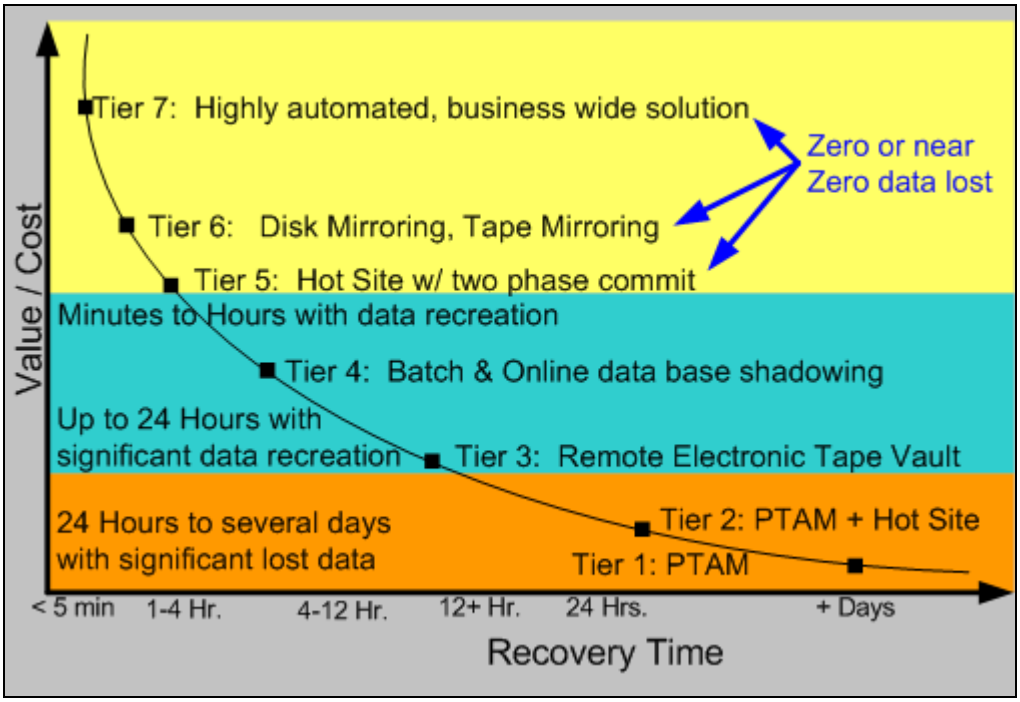

Figura 7: I 7 Livelli del Disaster Recovery

3.4 Costi del Disaster Recovery

È importante capire in quali casi convenga effettivamente ad una azienda tutelare il suo business investendo in una soluzione di Disaster Recovery. Infatti, non perché questa soluzione esiste è detto che essa debba essere applicata a qualsiasi tipo di organizzazione. Il costo di tale investimento non è a sé stante, ma è collegato al valore del proprio business: non si dovrebbe spendere più denaro nella pianificazione del Disaster Recovery di quanto se ne perderebbe se l'intero sistema fosse colpito da un disastro.[9]

Quindi non bisogna trascurare fattori come i costi di implementazione, di mantenimento e le perdite finanziarie che si avrebbero in seguito ad un disastro. Facendo un bilancio di tali fattori si riesce a capire se è conveniente o meno avere un System Recovery e a che livello lo si vuole applicare.
Nel nostro caso di studio è stato determinante il fatto che si tratta di un progetto ancora in via sperimentale, quindi essendo una applicazione di piccole dimensioni e ancora agli albori si è preferito realizzare un System Recovery che non eccedesse nei costi.

3.5 REPLICA DEI DATI

3.5.1 DEFINIZIONE DI REPLICA

Una definizione formale di replica dei dati è fornita di seguito:

"La replica è la copia dei dati da un sistema con i suoi dischi ad un altro sistema con i suoi dischi totalmente indipendenti e ridondanti. La replica non coincide con il mirroring del disco, perché il mirroring tratta entrambi i dischi come un singolo disco, mentre nella replica i due dischi sono visti come due entità totalmente indipendenti. Il mirroring è confinato ad un singolo computer system, mentre la replica sposta i dati da un sistema ad un altro. Il risultato finale è quello di avere due set di dati consistenti e ugualmente utilizzabili in due locazioni fisiche differenti."[10]

L'uso più comune delle tecniche di replica si ha nella realizzazione del Disaster Recovery.

3.5.2 TIPI DI REPLICA

Le principali categorie di replica sono le seguenti:

- ***sincrona***
- ***asincrona***
- ***semi-sincrona***
- ***periodica***

3.5.3 REPLICA SINCRONA

Mi soffermerò ad approfondire la replica sincrona, dal momento che è stato questo il protocollo utilizzato nel caso di studio che verrà trattato al capitolo 6. La replica sincrona è l'unica tra le categorie proposte a potersi considerare real-time. Nel momento in cui i dati vengono scritti sul nodo locale, il nodo remoto riceve gli stessi dati e avverte il nodo locale di averli ricevuti. A causa di questo avviso di ricezione da parte del nodo remoto, le scritture sul nodo locale possono subire un notevole ritardo, detto "latenza". La latenza è l'unico aspetto negativo della replica sincrona, poiché essa è l'unica tra le quattro tipologie a garantire l'integrità dei dati, dal momento che questi vengono scritti sul nodo remoto rispettando l'ordine di scrittura. La consistenza e l'usabilità dei dati è così garantita su entrambi i nodi.

3.5.4 REPLICA HARDWARE O SOFTWARE?

I dati possono essere replicati in due modi:

- ***hardware-based replication***
- ***software-based replication***

Nelle repliche hw-based la replica è avviata da meccanismi insiti al disco stesso. Solitamente i dischi coinvolti devono provenire

dallo stesso produttore e avere come supporto per lo scambio dei dati delle reti dedicate.

Nel caso di repliche sw-based il tutto è gestito da un software, il che crea indipendenza dall'hardware sottostante e di conseguenza una maggiore flessibilità, spesso associata a costi minori. La maggior parte delle repliche sw-based non richiedono particolari reti, ma poggiano su protocolli comuni come TCP/IP oppure UDP/IP. L'unica pecca della replica basata sul software è che può essere fatta solo da sistemi supportati dal software.

Solitamente in questo caso vengono scambiati dei blocchi di dati di dimensione minore rispetto alla replica hw, il che comporta un minore traffico sulla rete e una velocità maggiore.[10]

Due esempi delle tipologie di repliche possono essere: per la replica hw-based, l'interfaccia standard SCSI (Small Computer System Interface) e per la replica sw-based, il software DRBD (Distributed Replicated Block Device, utilizzato nel caso di studio).

SCSI è una interfaccia ampiamente impiegata per collegare direttamente i device di memorizzazione dati. Tramite il protocollo internet iSCSI vengono inviati i comandi ai dispositivi di memoria SCSI fisicamente collegati a server o ad altri dispositivi remoti. I comandi permettono la scrittura e la lettura sui vari device. I comandi possono essere inviati utilizzando il protocollo TCP/IP, rendendo così possibile l'utilizzo di reti esistenti così da estenderne l'utilizzo a distanze maggiori.

L'alternativa sw-based è DRBD. Il termine si riferisce sia al software impiegato (modulo kernel e tools applicativi associati) che al device logico a blocchi da esso gestito. DRBD verrà trattato in dettaglio al capitolo 6, viene qui introdotto solo per un veloce confronto con il caso di replica hw-based: entrambi iSCSI ed DRBD sono basati sul trasporto remoto su TCP/IP, durante il

quale possono perdersi pacchetti, ci può essere l'esigenza di ritrasmetterne alcuni e può essere introdotta una latenza consistente, ma in genere la replica software-based essendo quasi del tutto slegata dall'hardware sottostante tende ad essere più veloce e a sfruttare meglio le risorse.[12]

3.5.5 Replica dei dati e metodi di switching

Le componenti di un moderno sistema di calcolo possono essere suddivise in attive (device e processi) e passive (strutture dati). Dopo un disastro per poter continuare a fornire agli utenti i servizi attivi prima che questo accadesse, bisogna:

1) rendere disponibili i dati accumulati sul nodo primario sul nodo secondario
2) attivare i processi sul nodo secondario

Ognuna delle due funzioni, riferite rispettivamente una alla replica dei dati e una alla gestione dei processi possono essere assegnate ad una delle componenti attive di un moderno calcolatore oppure attuate "manualmente".[7]

3.6 Cluster Management

Quando i dati vengono replicati su un sito remoto, questo non è sufficiente a fornire una soluzione completa di System Recovery, non è detto infatti che i processi che servono a riportare in vita l'intera attività ripartano automaticamente. Bisogna che il software di replica si occupi di promuovere il nodo secondario a primario e che su questo riattivi le applicazioni critiche.
È importante che si riesca almeno a semiautomatizzare questo processo, riducendo al minimo l'intervento umano.

Quindi la replica dei dati da sola non include che esista un cluser management software a gestire le azioni da compiere.
È importante che questo processo sia automatizzato, poiché lo start manuale dei servizi in certi casi potrebbe non essere disponibile a causa del disastro.
Quando di sceglie di realizzare un clustering locale con almeno due nodi si potrebbe ricorrere al cluster manager Heartbeat, il quale supporta l'integrazione con la replica dei dati eseguita tramite DRBD e del quale verranno approfonditi gli aspetti tecnici nel capitolo 6.

CAPITOLO 4: Cluster

4.1 CLUSTER: GENERALITÀ

4.1.1 DEFINIZIONE

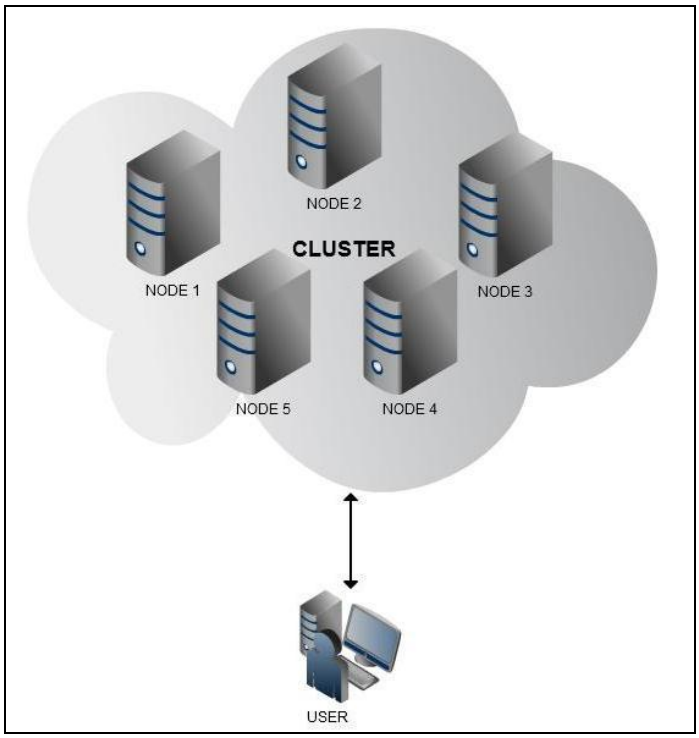

Figura 8: Raffigurazione di un generico cluster

Un cluster consiste di due o più nodi identici e indipendenti, ma interconnessi. Alcuni cluster sono predisposti unicamente per permettere al lavoro di essere trasferito su un nodo secondario se il primario cade, altri sono progettati per fornire scalabilità, permettendo agli user di distribuire il carico di lavoro sui diversi nodi.

Una prerogativa importante dei cluster è che i nodi possono apparire ad una applicazione come se si trattasse di un unico nodo, quindi la loro gestione può essere vista come la gestione di un unico nodo. È il software di gestione del cluster che offre

questa "trasparenza" all'utente. Affinché i nodi possano agire come un unico nodo, i file da essi richiesti devono essere resi disponibili ad ogni nodo che ne richieda l'uso. I nodi sono interconnessi fra loro da una rete di comunicazione privata usata e per la sincronizzazione, lo scambio e per la condivisione delle risorse. [13]

Ci sono diverse specifiche tipologie di cluster, la cui progettazione è legata agli intenti per cui vengono impiegati.[14]

4.1.2 SISTEMA OPERATIVO PER UN CLUSTER

Il sistema operativo associato ad un cluster può essere essenzialmente di due tipi:

1) ***Network operating system***
2) ***Distribuited operating system***

I due sistemi operativi differiscono principalmente nel modo in cui presentano il cluster allo user, ma entrambi garantiscono le operazioni di scambio tra i nodi tipiche di ogni cluster.

Nel primo caso viene mantenuta l'immagine del singolo nodo, poiché i singoli sistemi operativi presenti su ogni nodo vengono coordinati per gestire le risorse. Poiché non si ha una visione di insieme, l'utente è libero di scegliere su quale nodo eseguire le operazioni.

Nel secondo caso, si sfrutta la caratteristica peculiare del clustering, ossia il sistema viene visto come un unico nodo, in cui un sistema operativo globale si occupa di distribuire il lavoro sui vari nodi. Ogni nodo non è più visibile all'utente come singolo. Un distribuited operating system ha una maggiore complessità di un network operating system, oltre che una maggiore scalabilità, affidabilità e sicurezza.

4.1.3 FILE SYSTEM PER UN CLUSTER

Nel caso di un distribuited operating system, poiché tutti i nodi appaiono come un singolo nodo, deve essere garantita la stessa disponibilità dei dati per ogni nodo, cosicché possano operare come un'unica entità di calcolo. La scelta del file system è cruciale. I file system tradizionali non supportano il mounting simultaneo su più di un sistema, ciò vuol dire che se viene montato su un nodo non sarà accessibile dagli altri. Si dovrebbe quindi scegliere un file system che supporti gli accessi concorrenti da parte di più sistemi oppure utilizzare più che una condivisione delle risorse un metodo detto "Federated Cluster", utilizzato da alcuni produttori, in cui i dati sono sparsi sulle varie macchine piuttosto che essere condivisi da tutti, così da delegare allo specifico nodo che possiede la risorsa da usare la computazione.[14]

4.2 DIFFERENZA TRA HIGH PERFORMANCE CLUSTER E HIGH AVAILABILITY CLUSTER

Come accennato nell'introduzione esistono diverse tipologie di cluster a seconda dello scopo per cui vengono impiegati.

Un High Performance Cluster viene impiegato quando è richiesta una grande capacità di calcolo, come potrebbe accadere in campo scientifico nel caso delle simulazioni, per le quali vengono prodotte quantità massicce di elaborazioni in tempi molto ridotti.

Mentre un High Availability Cluster ha come scopo quello di garantire l'affidabilità di un sistema distribuito, garantendo la sopravvivenza dei servizi grazie alla prevenzione e alla gestione delle interruzioni.

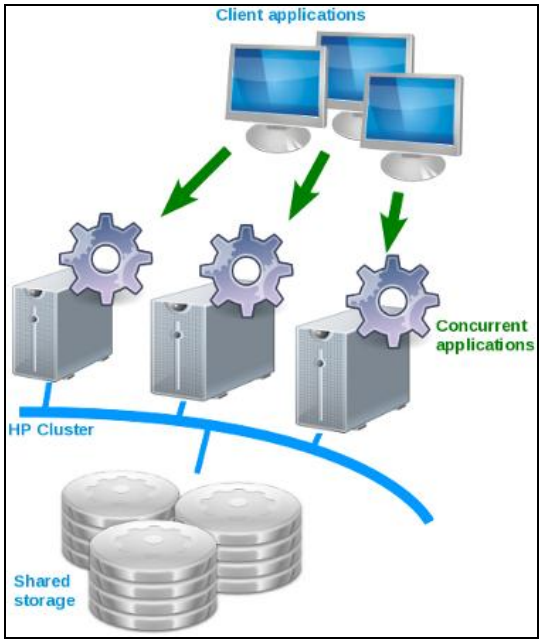

Figura 9: High Performance Cluster

Un cluster HA vuole minimizzare l'impatto delle interruzioni sul sistema spostando le attività dal nodo che stava erogando i servizi, che ha subito l'interruzione, ad un altro nodo disponibile e funzionante.

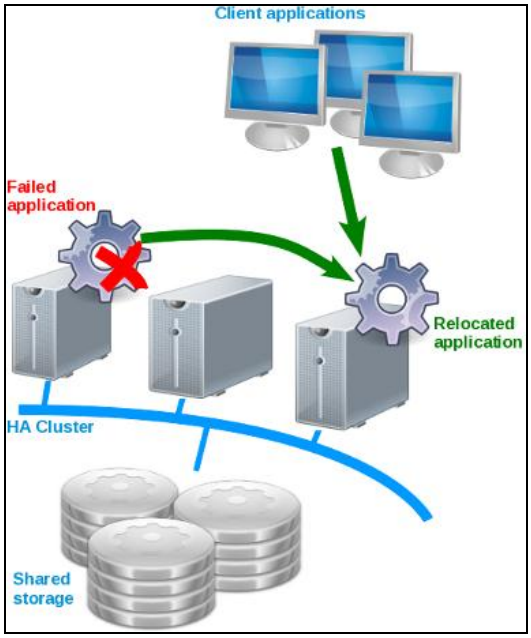

Figura 10: High Availability Cluster

Lo scopo che ci si prefigge nel progetto di un cluster per l'alta disponibilità è quello di eliminare i punti deboli del sistema

definiti “single point of failure”, attraverso la ridondanza degli stessi.

Dato che l’interesse primario di un cluster HA non sono le performance computazionali, solitamente non sono presenti molti nodi e spesso il loro numero è limitato a due. In questo modo diventa anche più semplice la gestione delle risorse condivise e la prevenzione di errori come il “brain-splitting”.

Il brain-splitting è legato ai problemi di comunicazione tra i due nodi, infatti qualora la linea di comunicazione che li unisce fosse unica, se dovesse cadere la connessione, anche se entrambi sono ancora funzionanti, ogni nodo crederà di essere rimasto l’unico in vita, quindi entrambi inizieranno a fornire gli stessi servizi duplicati, ma ancor peggio inizieranno a scrivere sulle risorse condivise contemporaneamente, creando inconsistenza di dati.

Lo switching fra i nodi è trasparente all’utente al quale sembrerà che i servizi sono forniti da un singolo sistema.

Riassumendo, le principali differenze tra le due tipologie di cluster sono:

- ***High performance:*** *priorità alle performance. Si lavora per rendere l'elaborazione il più veloce possibile*
- ***High availability (HA):*** *priorità ai servizi. Si lavora per garantire continuità a quanto erogato. Un disservizio deve essere, nei limiti del possibile, trasparente all'utilizzatore finale.*[15]

4.3 HIGH AVAILABILITY CLUSTER

4.3.1 GENERALITÀ DI UN CLUSTER HA

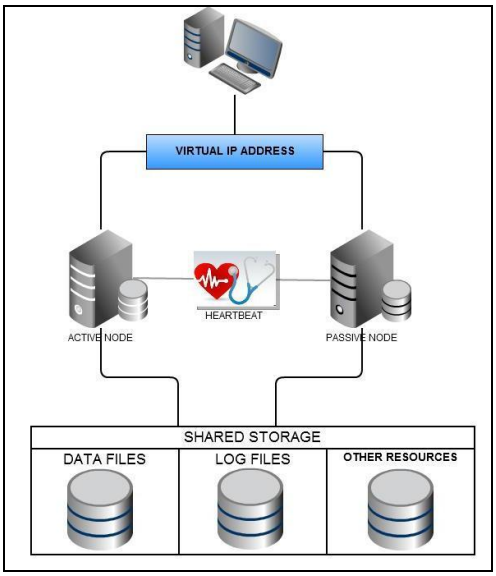

Figura 11: Cluser HA in dettaglio

Dopo aver definito gli obbiettivi primari di un cluster HA viene ora fornita una descrizione più dettagliata di quelle che sono le caratteristiche peculiari di questi sistemi per l'alta affidabilità.

In primo esame viene fatta una suddivisione dei cluster HA in due macro categorie:[15]

- ***shared-everything:*** cluster che per operare si basano su un'area comune (Storage, SAN, etc.). Il software di gestione del cluster si occupa di regolare gli accessi concorrenti ai dati nelle zone condivise
- ***shared-nothing:*** cluster in cui ciascun nodo è totalmente indipendente dagli altri (DRBD, replica MySQL). Il gestore del cluster si occupa di mantenere aggiornati i dati uniformemente e limitare i danni provocati da situazioni di split-brain.

Un sistema in HA di qualsiasi categoria deve essere in grado di ridurre al minimo il tempo di disservizio in seguito a: guasti hardware, malfunzionamenti del Sistema Operativo, malfun-

zionamenti delle applicazioni, interventi di manutenzione, errori umani o altri problemi. Il fine ultimo dell'HA è quello di ottenere la cosiddetta "Affidabilità 24x7" (24 ore su 24, 7 giorni su 7) dei servizi offerti da un sistema informatico. Tale fine ovviamente è estremamente difficile da raggiungere e nella maggior parte dei casi ci si accontenta di una sua buona approssimazione, dal momento che i costi per raggiungere un tale grado di affidabilità sono molto elevati e spesso possono essere affrontati solo da chi gestisce sistemi con "critical missions", per i quali è vitale evitare interruzioni.

Un'altra misura spesso utilizzata per esprimere il livello di affidabilità di un cluster è il cosiddetto "requisito dei cinque 9", presentato in dettaglio nel prossimo capitolo.

L'alta affidabilità in ogni caso non è soltanto un problema prettamente tecnico, ma coinvolge anche un aspetto delle risorse umane. Infatti oltre all'utilizzo di componenti hw/sw di alta qualità, all'applicazione della ridondanza dei possibili "points of failure" e alla pianificazione di tecniche di switchover bisognerebbe occuparsi anche della formazione del personale addetto alla gestione dei sistemi, dei servizi e delle infrastrutture.

Tramite un cluster HA è possibile gestire in modo sicuro risorse come: indirizzi IP, dispositivi di storage, file system e servizi offerti all'utente (mediante appositi server: web, mail, FTP).

Tali risorse per poter essere tutelate devono essere gestite unicamente dal cluster. Tutti i nodi del cluster HA durante il funzionamento si scambiano costantemente informazioni sul loro stato di salute, tramite quello che in gergo tecnico viene chiamato "heartbeat" (battito cardiaco).

L'heartbeat può essere scambiato mediante vari canali di comunicazione: cavo seriale o cavo cross Ethernet. Tali canali sono generalmente ridondati per aumentare la disponibilità.

Quando si verifica un malfunzionamento si ha una interruzione dell'heartbeat, che segnala che il nodo primario non è più attivo, quindi uno dei nodi in standby prenderà il suo posto acquisendone le risorse.
Lo switchover dal punto di vista del nodo che acquisisce le risorse prende il nome di "takeover", mentre lo switchover che consente al nodo primario di riacquisire le risorse quando torna disponibile, invece, prende il nome di "failback", che spesso è automatico su molti cluster HA.
Dopo aver descritto nel dettaglio il funzionamento tipico di un cluster HA, è chiaro perché esso spesso è composto da due soli nodi. La semplicità di questa scelta, non è un fattore che contribuisce tanto all'economicità della soluzione, quanto all'affidabilità della stessa. Infatti la complessità non incentiva l'alta affidabilità, bensì la ostacola.[16]

4.3.2 PROGETTI OPEN-SOURCE PER LA REALIZZAZIONE DI CLUSTER HA

Oltre alla possibilità di acquistare un sistema ad alta disponibilità da un produttore, esiste quella di utilizzare soluzioni open-source che oltre all'economicità permettono di intervenire sul codice sorgente modificandolo a seconda delle proprie esigenze, senza le limitazioni che si avrebbero utilizzando tecnologie proprietarie.
Infatti un sistema in alta disponibilità è caratterizzato da diversi componenti software che comunicano tra loro per fornire servizi in alta disponibilità. Queste componenti nelle soluzioni proprietarie sono spesso nascoste e l'utente del sistema, che spesso comprende anche l'hardware, può non avere nessuna conoscenza di tali componenti, né di come essi interagiscono, né delle funzioni che essi svolgono.
È possibile risparmiando e incentivando anche la diffusione della filosofia dell'open-source, oggi più che mai promossa con

l'avvento e la diffusione massiva di internet, realizzare lo stesso prodotto usando componenti software di tipo open-source che oltre ad essere gratuiti il più delle volte, hanno un codice sorgente modificabile, e talvolta anche ridistribuibile, a seconda della licenza in questione. Risulta quindi evidente come ogni soggetto che voglia implementare una soluzione di cluster in alta disponibilità possa sfruttare le proprie conoscenze per progettare il sistema in ogni sua parte accedendo a risorse open-source, riuscendo ad acquisire le competenze necessarie per costruire il proprio sistema personalizzato.

Alcune delle soluzioni cluster disponibili per Linux sono:[15]

- *Veritas Cluster:* licenza chiusa, prodotto e mantenuto da Symantec
- *Oracle RAC:* licenza chiusa, prodotto e mantenuto da Oracle
- *Red Hat Cluster:* licenza GPL, mantenuto da Red Hat
- *Linux HA:* licenza GPL/LGPL, prodotto e mantenuto dalla comunità Linux-HA

Il caso di studio descritto al capitolo 6 di questa tesi descrive il cluster realizzato utilizzando totalmente strumenti open-source.

4.3.3 MODELLI DI CLUSTER HA

Si possono realizzare due tipologie di cluster che rispecchiano i requisiti di semplicità e affidabilità presentati finora:

1. Cluster Active-Active (AA)

2. Active-StandBy (AS)

Nel modello A-S, i servizi offerti dal cluster sono residenti tutti su un nodo, vengono spostati totalmente sull'altro nodo precedentemente in standby, nel momento in cui il nodo attivo cade.

Nel modello A-A i servizi sono distribuiti su entrambi i nodi che si compensano a vicenda quando uno dei due cade.
Entrambi minimizzano, con due strategie diverse, il tempo di interruzione. Ci deve essere un software che monitora lo stato dei nodi e arbitra l'attivazione a nodo primario di uno dei due.

CAPITOLO5: High Availability, l’Alta Affidabilità

5.1 Differenza tra High Availability e Disaster Recovery

Finora il concetto di Alta Affidabilità è stato discusso solo a margine della sua applicazione pratica ai cluster HA. In questo capitolo sarà approfondita tutta la teoria dell’Alta Affidabilità. È bene distinguere l'Alta Affidabilità dal Disaster Recovery, trattato ampiamente nel capitolo 3. Il Disaster Recovery si colloca all'interno di eventi disastrosi, che hanno una bassa probabilità di verificarsi, ma che possono causare conseguenze gravi al business. Nel Business Continuity Plan si prevedono tempi di ripristino rapidi e prestabiliti, che possano arginare i danni causati dal disastro. Spesso si tratta di recuperare l'operatività dopo un grave incidente (es: un terremoto, un incendio, una alluvione...) normalmente in sede diversa da quella originale.

L'Alta Affidabilità riguarda, invece, quella categoria di interruzioni che hanno una maggiore probabilità di verificarsi dovute principalmente a malfunzionamenti di componenti hardware o software o a incidenti locali di esigua entità. Serve a coprire la continuità dei servizi e garantire l’integrità dei dati.

Alta affidabilità, in realtà è una traduzione "errata" dei termini inglesi: High Availability. "Availability" significa letteralmente "disponibilità", quindi l'HA sta ad indicare il concetto di "disponibilità continua nel tempo", che può essere interpretato come vera e propria affidabilità.

5.2 LIVELLI DI AFFIDABILITÀ

L'affidabilità può essere resa su diversi livelli, che in ordine crescente presentano una complessità e una efficienza sempre maggiori, dalle garanzie minime offerte dal livello uno a quelle ad alto livello offerte all'ultimo livello, che coincide con il Disaster Recovery.

LIVELLO 1: Affidabilità normale

Si tratta del livello più basso, in cui per proteggere i servizi offerti si procede ad un semplice backup dei dati periodico. L'efficienza di questo sistema, tanto semplice quanto rudimentale, sta nella periodicità con cui i dati vengono salvati. Poco male se l'interruzione avviene in un istante di tempo relativamente vicino all'ultimo salvataggio, ma se ciò non dovesse accadere, con un periodo di backup lungo, si perderà una notevole quantità di dati.
I servizi torneranno attivi solo dopo aver recuperato totalmente i dati se la qualità del ripristino lo permetterà. Le tempistiche in questo livello sono imprevedibili.

LIVELLO 2: Affidabilità maggiore

Su questo livello si realizza quanto detto per il livello sottostante e in più si cerca di migliorare l'aspetto dell'attualità dei dati, cercando di mantenere i backup quanto più aggiornati possibile. La tecnica impiegata per fare ciò è quella del mirroring fisico tra dischi, attraverso la tradizionale tecnologia RAID (Redundant Array of Independent Disks)[17], che sfrutta un insieme di dischi rigidi per condividere o replicare i dati, migliorandone l'integrità e la tolleranza ai guasti rispetto all'impiego di un unico disco. Anche

in questo caso, pur avendo dati più aggiornati, i tempi sono imprevedibili, dal momento che la riattivazione dei servizi dipende totalmente dall'intervento umano degli amministratori del sistema.

LIVELLO 3: Alta affidabilità

Al terzo livello si tutelano, non solo i dati, ma anche i servizi, il tutto indipendentemente dall'azione umana. Un esempio di alta affidabilità di livello tre è un cluster a due nodi in configurazione AS. Il funzionamento di quest'ultimo è già stato affrontato nel capitolo precedente.

LIVELLO 4: Disaster Recovery

L'evoluzione all'ultimo livello dell'alta disponibilità è il Disaster Recovery, inteso come replica totale dell'intero sito che fornisce i servizi su un sito secondario lontano dal primo. A seconda dei livelli di fault tolerance che si vuole garantire ci si occuperà di realizzare uno dei sette livelli di recovery presentati al capitolo 2.

5.3 REQUISITO DEI CINQUE 9

Solitamente un sistema ad alta affidabilità viene classificato in base al costo, alle prestazioni e al tempo di disponibilità dei servizi. Il tempo è il più importante tra questi parametri. Si vuole che il tempo si erogazione effettiva dei servizi soddisfi il cosiddetto "requisito dei cinque 9", ovvero che raggiunga una percentuale del 99,999% del tempo di osservazione del sistema. Il sistema deve risultare attivo nel 99,999% del tempo di monitoraggio. Se si considera ad esempio l'utilizzo di un sistema per un anno intero, si ottiene che il tempo massimo in cui il

sistema può andare off-line è di 5 minuti e 15 secondi, come mostrato in figura 12.

Percentuale Uptime	Downtime per anno	Downtime per settimana
98,00%	7,3 giorni	3 ore, 22 minuti
99,00%	3,65 giorni	1 ora, 41 minuti
99,80%	17 ore, 30 minuti	20 minuti, 10 secondi
99,90%	8 ore, 45 minuti	10 minuti, 5 secondi
99,99%	52 minuti, 30 secondi	1 minuto
99,999%	5 minuti, 15 secondi	6 secondi
99,9999%	30,5 secondi	0,6 secondi

Figura 12: Tabella tempi downtime per anno/settimana

Per ogni riga vediamo il tempo di downtime concesso in un anno e in una settimana, che un sistema deve rispettare per soddisfare il requisito relativo a quel determinato numero di "9". L'obbiettivo dei cinque 9 è evidente che sia molto difficile da raggiungere; esso comporta l'eliminazione totale dei "single point of failure" del sistema o ridondando le componenti critiche oppure con la costruzione di un cluster, come accade nel caso di studio esposto nella tesi.

Un esempio per la prima categoria di soluzioni potrebbe essere quello della ridondanza dei dischi, i quali se supportano anche la sostituzione a caldo diminuiscono ancor di più i problemi nel caso di guasto. Questa soluzione si definisce monolitica e richiede l'utilizzo di componenti molto particolari e costosi, che realizzino i collegamenti.

Per quanto riguarda l'altra soluzione è certamente più economica, perché il sistema che non funziona viene soppiantato totalmente dal secondario, sul quale si trova una copia esatta dei dati e sul quale vengono avviati gli stessi servizi. Il vantaggio più evidente è quello che non viene utilizzato hardware dedicato, ma si utilizzano dei generici calcolatori che collegati opportunamente riescono a costituire un cluster. Lo svantaggio è che la configu-

razione del software che sta dietro a una simile installazione, essa è articolata e richiede una certa attenzione da parte dell'utilizzatore.

È opinione abbastanza diffusa che l'alta affidabilità richieda un numero di 9 non inferiore a 3. Spesso il requisito dei cinque 9 risulta un target troppo ambizioso, che spreca risorse economiche e non, dal momento che per tutte le applicazioni non critiche i tempi di recupero possono anche essere maggiori senza compromettere il business sia dal punto di vista delle funzionalità che da quello finanziario.

Alcuni servizi potrebbero richiedere di soddisfare il requisito dei cinque 9, in modo da proteggersi contro effetti dannosi, ma ignorando questi casi critici, secondo alcuni il requisito può anche non essere soddisfatto (ad esempio per i servizi internet). In questa visione soddisfarlo per minimizzare la percezione degli effetti avversi da parte degli utenti, vuol dire sottrarre risorse del sistema che potrebbero essere utilizzare per altri scopi, ad esempio la tutela dell'integrità dati.[18]

Secondo altri invece soddisfare il requisito dei cinque nove per servizi non critici, come quelli internet, è importante. La mancanza di disponibilità di un servizio internet per la fetta di organizzazioni che basano il loro introito finanziario su questi ultimi, comporta la perdita di molti clienti.[19]

Per concludere quindi, il requisito dei cinque 9 dovrebbe essere soddisfatto nei casi di sistemi con missioni critiche (come nel nostro caso, in cui è importante mantenere la continuità del sistema di monitoraggio di scorie radioattive) e nei sistemi di importanti organizzazioni commerciali.

5.4 LE PRESTAZIONI

Negli ultimi anni il problema della Business Continuity ha interessato non solo le grandi aziende produttrici di prodotti per l'IT, ma anche grandi gruppi di ricerca come la Garter INC.[20] La Gartner INC. oltre a svolgere analisi di mercato sull'andamento delle imprese, si occupa anche di analisi delle tendenze comuni e di comportamento.

Il tema della Business Continuity, così come quello del Disaster Recovery, è stato ampliamente trattato dalla Gartner INC., la quale nelle sue linee guida evidenzia che il tempo di attesa tollerabile dagli utenti dopo la richiesta di un servizio va dai 4 ai 20 secondi relativamente al tipo di applicazione.

Risulta chiaro che affinché i servizi forniti non subiscano una interruzione che superi le soglie di tempo fissate, in caso di malfunzionamenti dei server, si deve fare in modo che il tempo di recovery sia molto rapido.

L'unico tempo su cui si possono fare delle previsioni è quello di risposta del sistema, ma a questo vanno aggiunti dei tempi non prevedibili come quelli di latenza dovuti alla larghezza di banda del sistema di comunicazione tra client e server. Questo vuol dire che durante un progetto di alta affidabilità si devono considerare anche questi aspetti "esterni", ma indispensabili, al sistema.

Un altro aspetto importante è quello di lavorare nell'ipotesi che molti utenti siano connessi al servizio. L'erogazione infatti deve essere indipendente dal numero di utenti connessi, è auspicabile che il sistema riesca a reggere un carico di utenti massimo previsto senza avere cali notevoli nelle prestazioni.

Si deve quindi procedere al dimensionamento del sistema a partire dal numero ipotetico di utenti che richiedono contemporaneamente il servizio.

Il processo di dimensionamento consiste nell'individuazione dei parametri che caratterizzano il sistema in oggetto e che ne garantiscono al tempo stesso la piena funzionalità/prestazioni secondo le specifiche tecniche del progetto nonché la sua affidabilità e sicurezza durante l'intero tempo di vita operativa o di esercizio dell'opera stessa.[21]

Nel caso in cui il sistema informatico sia di tipo monolitico, l'unico intervento da fare è quello di potenziare l'hardware sottostante, sostituendo o aggiornando le componenti che lo rallentano. Ottenere miglioramenti nell'erogazione dei servizi è costoso e complesso.

Nel caso in cui invece la scelta per l'alta affidabilità sia stata quella del clustering, un miglioramento è subito ottenibile in maniera semplice ed economica, aumentando il numero di nodi a disposizione sui quali viene distribuito il carico del lavoro, realizzando quello che viene definito cluster con bilanciamento dinamico delle risorse.

CAPITOLO 6: Il caso di studio

6.1 PRESENTAZIONE DEL CASO DI STUDIO

6.1.1 IL PROGETTO DMNR

INFN-Energy, di cui il progetto DMNR (Detector Mesh for Nuclear Repository) fa parte, è una linea di ricerca che concerne la produzione e l'uso di energia nucleare, nonché lo studio delle problematiche e tecnologie connesse.

Tra le tante questioni legate allo smaltimento dei rifiuti radio-attivi a breve e medio termine, vi è la conservazione all'interno di siti idonei in cui deve essere garantito un elevato livello di sicurezza. Infatti la possibilità che all'interno di un sito di stoccaggio vi siano delle perdite è reale. Al fine di individuare eventuali fughe di materiale radioattivo e di ridurre efficace-mente i rischi di contaminazione per gli operatori e per l'ambiente circostante è preferibile l'adozione di sistemi di monitoraggio in tempo reale. Il DMNR è un progetto innovativo volto a garantire il monitoraggio real-time dei depositi di scorie radioattive. Al momento sembra che un tale sistema non sia in funzione in nessun sito di stoccaggio in tutto il mondo. L'idea alla base del progetto è del tutto innovativa, infatti fino al giorno d'oggi non esiste un sistema che riesca a garantire un monitoraggio sicuro, tempestivo e continuo. Fino ad ora la soluzione più comune è stata quella di utilizzare dei controlli diretti da parte di operatori specializzati, con i conseguenti rischi legati alla salute di questi ultimi nel caso di fuoriuscite radioattive. Un'altra soluzione possibile, benché molto costosa è quella dei contatori Geiger-Muller. DMNR è invece una soluzione economica e per questo innovativa che si basa sull'utilizzo di

sensori realizzati con materiale a basso costo collegati ad una elettronica di front-end anch'essa dai prezzi contenuti. Il sistema sfrutta le grandi potenzialità delle fibre ottiche scintillanti per inviare a dei microcontrollori FPGA le informazioni rilevate dai bidoni cui i sensori ottici sono applicati. I fotosensori ad altissima sensibilità solitamente sono disposti a griglia attorno ad ogni singolo bidone in un numero che varia dalle cinque alle dieci unità. I rivelatori fibra + fotosensori sono in grado di contare le radiazioni gamma da cui è possibile valutare l'attività radioattiva attuale. Lo scopo primario del sistema è misurare il livello di attività per rappresentare l'andamento della radioattività di ogni contenitore.

L'elettronica collegata ai sensori è costituita da un sistema FPGA-based che comunica con una console locale su cui vengono visualizzati a schermo e registrati i dati provenienti dalle fibre.

Ricordando che la pericolosità delle scorie radioattive decresce in un tempo che può essere dell'ordine di migliaia di anni, tale sistema si presta a osservazioni a breve e a medio termine. Il sistema inoltre mira a creare uno storico degli stabilimenti di stoccaggio in modo da rendere rintracciabili i bidoni anche in seguito a spostamento degli stessi all'interno dell'impianto e tenere un monitoraggio continuo nel tempo di quest'ultimo.

I primi test eseguiti sulle sorgenti radioattive hanno dato dei risultati molto incoraggianti.

Il progetto DMNR (Detector Mesh for Nuclear Repositories) ha da poco iniziato a studiare la possibilità di un sistema integrato per il monitoraggio on line sul campo, fornendo una mappa tridimensionale della radioattività prodotta dai rifiuti. In questo modo anche una piccola fuga di materiale radioattivo potrebbe essere riconosciuta a causa dell'aumento dell'attività locale.[22]

6.1.2 LE MOTIVAZIONI DEL CASO DI STUDIO

Dalla breve presentazione del progetto al paragrafo 6.1 si evince chiaramente che l'obiettivo primario è quello di riuscire a realizzare un controllo sugli impianti di stoccaggio non più periodico, come avviene con le tecniche tradizionali accennate, ma in tempo reale. Da questo deriva la necessità della garanzia della continuità del servizio. In figura 13 viene rappresentata l'architettura hardware dell'intero sistema.

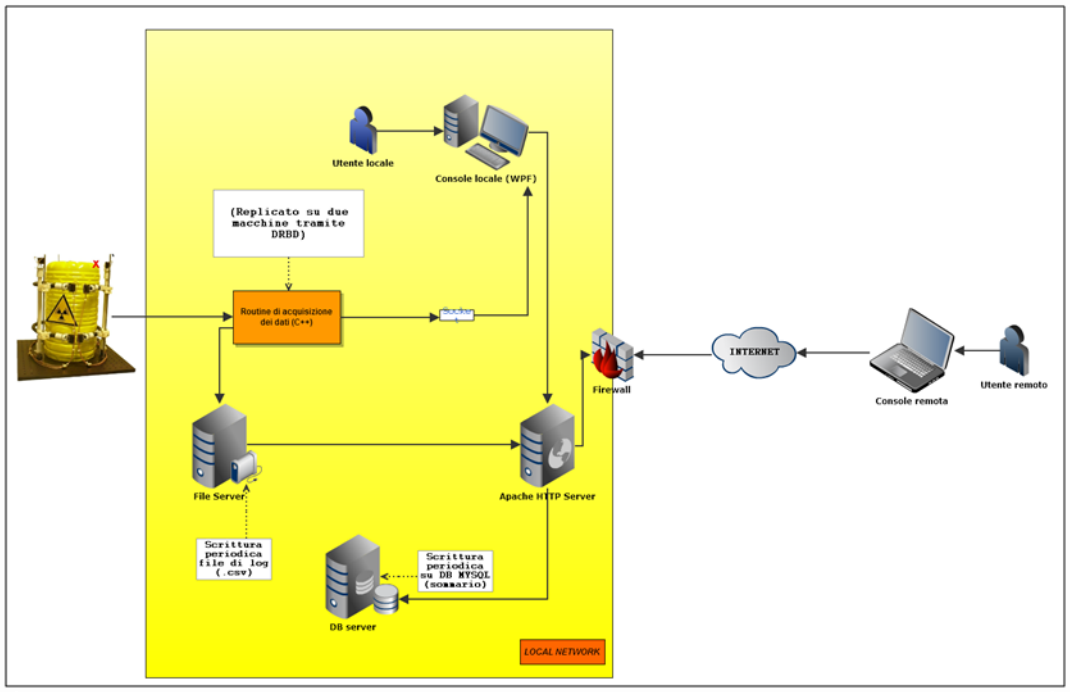

Figura 13: Architettura hardware del sistema

A partire da sinistra viene rappresentato tutto il processo di raccolta, immagazzinamento e presentazione dei dati. Viene evidenziata la duplicazione dei dati provenienti dalla routine di acquisizione. In caso di interruzione per malfunzionamenti hardware o software si deve garantire che questi dati non vadano persi.

Il servizio che si vuol garantire su questo sistema può essere considerato critico, dal momento che sono proprio i dati raccolti

istante per istante a rappresentare la storia dell'impianto e a evidenziare situazioni pericolose. La perdita di una parte di questi dati potrebbe nascondere un allarme e tale situazione di emergenza, dal momento che non viene riconosciuta tale, potrebbe essere trascurata.

Bisogna quindi garantire la continuità del servizio attraverso la realizzazione del concetto di High Availability trattato ampiamente al capitolo 5.

In particolare il cluster HA in configurazione Active-Standby realizzato per questo sistema serve non solo a porre in duplicato su due macchine differenti i file di log provenienti dai microcontrollori FPGA, ma anche a duplicare i servizi offerti da una applicazione web-based (figura 14) che costituisce il front-end del sistema.

Figura 14: Applicazione web in php

Tale applicazione è in grado non solo di presentare i dati all'utente, che in questo caso si suppone sia un membro dell'INFN interessato a monitorare lo stato dei bidoni radioattivi, ma si occupa anche della parte di back-end che consiste nel

prelevare i dati dai file di log, immagazzinarli su un grande database e da qui eseguire le elaborazioni atte a fornire dei dati utili per comprendere e rappresentare l'andamento globale dell'attività radioattiva.
Quindi nella prospettiva di possibili interruzioni indesiderate sul cluster HA si avrà uno switch tra nodo primario e secondario che permetterà di non perdere nessuno dei dati forniti dall'elettronica di back-end.

6.2 STRUMENTI SOFTWARE UTILIZZATI

6.2.1 DRBD

DRBD (Distributed Replicated Block Device) è un device a blocchi replicato sulla rete sviluppato per la piattaforma GNU/Linux. È stato sviluppato da Philipp Reisner, il quale iniziò il progetto come tesi di laurea nel 2000 e che tuttora ne continua lo sviluppo. Prodotto e mantenuto da Linbit[23] e distribuito con licenza GPL; consiste essenzialmente di un modulo kernel e alcuni tools di gestione dedicati. In figura 15 viene rappresentato il funzionamento generale di DRBD [24].

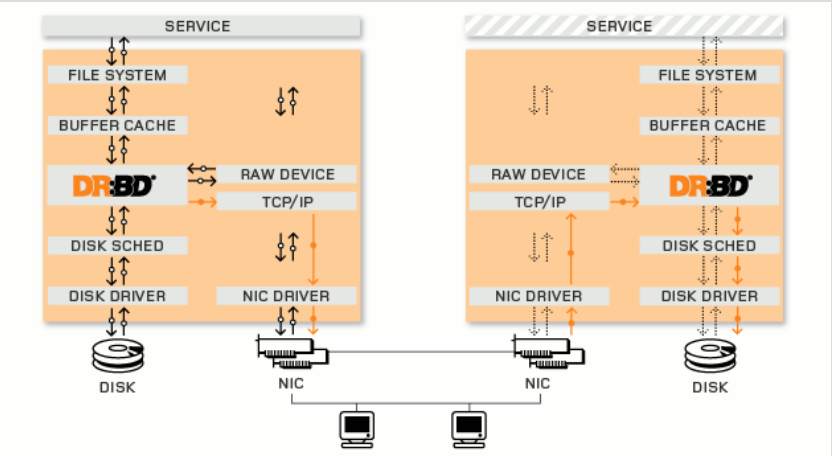

Figura 15: Due server di un cluster HA basato su DRBD

DRBD si riferisce sia alla parte software appena citata (modulo kernel e userspace tool associati) che ai dispositivi logici a blocchi gestiti dal software.
DRBD ha molte somiglianze con RAID, la differenza principale è che opera su network. Si può tranquillamente assimilare ad un RAID via ethernet: il device a blocchi viene replicato infatti su un canale ethernet dedicato.
DRBD è stato ideato per i cluster HA, per svolgere la funzione di replica dei dati tra i nodi del cluster. DRBD non fa una chiara differenza tra replica sincrona e asincrona, ma offre tre protocolli di replica:

- protocollo A
- protocollo B
- protocollo C

Protocollo A:

Quando si usa il protocollo A, quando l'operazione di scrittura è stata completata sul nodo locale DRBD invia i dati al nodo remoto. DRBD in questo caso non aspetta nessun tipo di ack da parte del nodo remoto, ciò vale a dire che non appena termina la scrittura sul proprio dispositivo locale, il nodo locale invia il segnale di aver terminato l'operazione senza preoccuparsi di ciò che è avvenuto sul secondario. In questo protocollo vi è un problema importante, che è quello che se il nodo locale fallisce durante l'operazione di scrittura, prima che il secondario abbia ricevuto tutti i dati, la scrittura dei dati non sarà speculare, ma ci sarà una perdita di dati. Quindi può accadere che alcune operazioni di scrittura falliscano. Dopo un certo numero di tentativi falliti da parte del nodo primario il nodo remoto viene abilitato a "primario" e si scambiano i ruoli. Questo tipo di

protocollo può essere considerato asincrono, dove si rischia la perdita di alcune operazioni di scrittura per quanto visto. Questo protocollo può essere utile nel caso di linee di connessione molto lente, in cui è impensabile di occupare banda per i segnali di acknowledgment.

PROTOCOLLO B:

DRBD si occupa di segnalare che una operazione di scrittura è completa quando è stata conclusa sul nodo locale ed è stato ricevuto un ack da parte del nodo remoto. L'ack viene spedito dal secondario al primario appena riceve il comando da quest'ultimo di avviare la scrittura, che eseguirà successivamente. Questo comporta un miglioramento rispetto al protocollo A, perché si favorisce una maggiore sicurezza nella replica dei dati. Se il nodo primario fallisce il nodo secondario può continuare ad aggiornare i dati e non si ha perdita di dati. Il protocollo B è un mix tra il protocollo A e il protocollo C, quindi non è del tutto soddisfacente.

PROTOCOLLO C:

Il protocollo che assicura la massima sicurezza nella replica dei dati è il protocollo C. Esso può essere considerato a tutti gli effetti un tipo di replica sincrona. DRBD vedrà il completamento della scrittura dei dati solo non appena entrambi i nodi hanno finito di scrivere. Dopo ogni operazione i due nodi saranno perfettamente allineati. Nel caso si utilizzi il protocollo C si devono considerare i tempi dovuti non solo alla scrittura su ogni nodo ma anche a quelli dovuti alla trasmissione dei segnali di sincronizzazione tra i due nodi, quindi è bene che la rete di trasmissione su cui questi vengono mandati non presenti particolari problemi di latenza.

Da quanto detto finora i due nodi lavorano in configurazione master-slave. In particolare su un sistema DRBD vengono denominati "nodo primario" il master e "nodo secondario" lo slave, e in ogni istante solo uno dei due può essere primario, mai tutti e due.
Il nodo primario può eseguire sia operazioni di lettura che di scrittura, mentre il secondario generalmente può solo leggere. I ruoli si possono scambiare senza problemi e lo scambio non richiede la risincronizzazione dei dati. Al momento DRBD è in grado di gestire cluster a soli 2 nodi.
La sincronizzazione dei nodi su DRBD può avvenire in due modi:

- *full synchronization (FullSync)*
- *quick synchronization (QuickSync)*

La Full Synchronization viene utilizzata solo quando non è possibile utilizzare la Quick Synchronization, che si può applicare quasi sempre tramite un espediente spicciolo. Si interrompe la comunicazione tra i nodi, ad esempio staccando fisicamente il cavo ethernet che li collega, e successivamente la si ripristina. DRBD si rende conto automaticamente che la replica ha subito una interruzione e provvede a memorizzare i cambiamenti effettuati al block-device in una locazione del disco che si definisce “bitmap”, per poter eseguire, quando il collegamento fra i nodi è ripristinato, l'aggiornamento della replica sull'altro nodo, dei soli dati specificati dalla bitmap.
La configurazione di DRBD è abbastanza semplice e lineare.
I file interessati nella configurazione di DRBD sono:

- **/etc/drbd.conf:** può contenere totalmente la configurazione di DRBD o includere altri file;

- **/etc/drbd.d/global_common.conf:** generalmente contiene le opzioni globali (sezione global) e comuni (sezione common) della configurazione;
- **/etc/drbd.d/rX.res:** definizione della risorsa rX (dove X indica la generica risorsa);

I file devono esistere su entrambi i nodi e la loro configurazione verrà presentata in riferimento al caso di studio al paragrafo 6.3.
In particolare nel file /etc/drbd.conf viene definita la cosiddetta: risorsa. Il concetto di risorsa è importante, perché rappresenta il cluster. Contiene i nodi che lo compongono e i relativi indirizzi IP dediti alla comunicazione e i dispositivi a blocchi impiegati per creare l'area di storage condiviso.
DRBD usa i "meta-data", sono dei dati speciali che contengono informazioni sulla replica dei dati, dei tipici valori presenti sono quelli delle dimensioni dei device oppure i log di attività. Solitamente stanno su un'area dedicata. Di default nella guida ufficiale di DRBD si fa riferimento agli "Internal meta-data", i quali sono posti dentro al device stesso sugli ultimi blocchi di memoria disponibili, per evitare possibili sovrascrizioni con i dati utili, cosa che accade quando la partizione usata è quasi totalmente piena. Per evitare problemi di questo tipo si può ricorrere agli "External meta-data" per i quali viene allocata una partizione specifica separata da quelle contenente i dati utili.
DRBD garantisce le operazioni di scrittura e consistenza su tutte le macchine impostate nella configurazione del cluster. Questo strumento si occupa di controllare i nodi della nostra rete per mantenere sempre allineate le macchine e quindi poterle sfruttare in caso di fallimento di uno o più nodi, garantendo in questo modo sempre i dati disponibili.

Come detto in precedenza, DRBD effettua un RAID 1 su rete e per questa motivazione non è possibile avere lo stesso volume montato su due o più server contemporaneamente, per questa motivazione la configurazione tipica che si realizza è di tipo "ACTIVE-STANDBY" ovvero una macchina attiva e l'altra in standby pronta ad essere utilizzate in caso di fallimento della macchina attiva.
Il pregio migliore del cluster DRBD è sicuramente il costo e la semplicità di configurazione e attivazione del sistema.
I costi sono estremamente ridotti, in quanto non si ha la necessità di uno storage condiviso, spesso estremamente costoso e non alla portata di chiunque, questo perché per costituzione del sistema è proprio il gestore DRBD che allinea i dischi in tempo reale, come se fossero un unico sistema fisico.
Il difetto principale di un cluster HA basato su DRBD è quello già introdotto al capitolo 4, lo split-brain. In questa evenienza i nodi del cluster non riescono a scambiarsi messaggi, vi sono molti metodi per ridurre le probabilità di split-brain, spesso sono complessi quindi rimane un problema che deve essere considerato.
Purtroppo DRBD è solo un gestore RAID basato su rete e quindi non ha conoscenza dei servizi che si hanno su un server, come i servizi web apache, ftp e riguardanti la gestione dei database. Per questa motivazione la completa configurazione si conclude con Heartbeat, tradotto battito cardiaco, che invece si occupa di automatizzare il processo di switching da una macchina in fallimento verso una disponibile e viceversa quanto la principale ritorna "on-road", considerando oltre ai dati anche l'aspetto dei servizi.

6.2.2 HEARTBEAT E PACEMAKER

Heartbeat è un demone che fornisce una infrastruttura per la comunicazione all'interno di un cluster. L'impiego di Heartbeat permette ad ogni nodo di conoscere quali altri nodi fanno parte del cluster e di scambiare con essi messaggi.

Heartbeat per risultare utile all'utilizzatore deve essere combinato con un cosiddetto CRM (Cluster Resource Manager) che ha il compito di attivare e disattivare i servizi (web server, indirizzi IP, etc.) che il cluster vuole rendere disponibili. In passato veniva usato anche senza CRM, ma questa pratica è ormai considerata obsoleta. Solitamente viene impiegato il CRM: Pacemaker, il quale è ricco di funzionalità pensate per supportare totalmente Heartbeat.

Esistono varie interfacce per Pacemaker, ma la più usata è certamente la "crm shell" sviluppata da Dejan Muhamedagic, che per molti resta la prima e unica scelta per il cluster management in alta affidabilità. Si tratta di un'interfaccia a linea di comando capace di fornire una serie di comandi crm interattivi che permettono non solo di gestire il cluster, ma anche di agire sulla sua configurazione.[25]

Come accade il più delle volte per le suite di programmi sviluppate su piattaforma Linux, si tratta di architetture software modulari. Ogni modulo svolge un compito specifico e può essere aggiornato indipendentemente dagli altri, qualora vengano resi disponibili dei miglioramenti o in caso di malfunzionamenti si può agire localmente sul modulo interessato dal problema, senza intaccare le altre componenti della suite.[26]

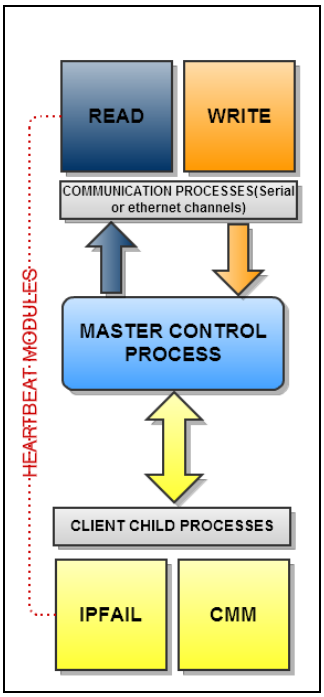

Figura 16: Architettura modulare di Heartbeat

In figura 16 troviamo una rappresentazione globale dell'organizzazione modulare di Heartbeat.
Il modulo di comunicazione rappresenta la parte superiore dello schema dei moduli e serve a gestire le autenticazioni e lo scambio dei messaggi. Le comunicazioni che Heartbeat è in grado di supportare sono di 4 tipi:

- unicast UDP su IPv4
- broadcast UDP su IPv4
- multicast UDP su IPv4
- comunicazione su linea seriale

In basso troviamo due processi di importanza fondamentale, che possono essere considerati processi “client”, poiché eseguono operazione esterne al modulo "Master Control Process" basate sulla richiesta di servizi a quest'ultimo. Essi sono “ipfail” e “CCM” (Consensus Cluster Membership).

Il processo "ipfail" esegue un monitoraggio della connessione fra i nodi e stabilisce quando è necessario eseguire il failover del gruppo di servizi che verrà delegato materialmente al Master Control Process.
Il processo CCM si occupa di monitorare lo stato globale del cluster che comunica istantaneamente al Master svolge la funzione di monitorare lo stato del cluster per fornire in ogni momento al Master Control Process. Tra le informazioni che fornisce ci sono le identità dei nodi che compongono il cluster e lo stato di funzionamento. Il CCM serve essenzialmente a controllare che la connessione tra i nodi funzioni e a identificare possibili problemi legati a incidenti esterni o malfunzionamenti software interni.
Le tre componenti modulari presentate costituiscono il nucleo di Heartbeat.
Come anticipato Heartbeat è una suite molto ricca di funzioni, molte delle quali sono svolte non dai moduli centrali appena presentati, ma da una serie di plugin che fanno da contorno al nucleo di Heartbeat. Le funzioni svolte dai plugin sono più ad alto livello e costituiscono una interfaccia di accesso diretto al contesto operativo del cluster. I plugin principalmente utilizzati, sono tre:

- plugin di STONITH
- plugin di Comunicazione
- plugin di autenticazione

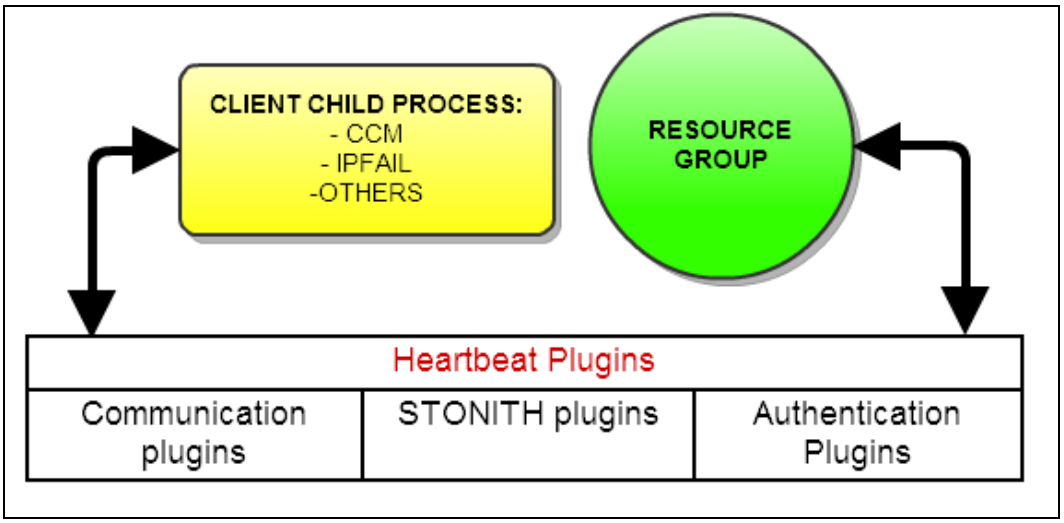

Figura 17: Principali plugin di Heartbeat

I ***plugin di STONITH*** (Shoot The Other Node In The Head) servono a risolvere il problema del brain-splitting. STONITH è un vero e proprio di interruttore che permette di spegnere istantaneamente un nodo del cluster quando serve. Ad esempio se entrambi i nodi di un Cluser HA si pongono come "primary" a causa di interruzione della comunicazione, uno dei due verrà spento per riportare la situazione alla normalità.

Per quanto riguarda i ***plugin di comunicazione*** essi sono essenziali per gestire i canali di comunicazione. Tutti i segnali sia in ingresso che in uscita in ogni caso passeranno per Master Control Process di Heartbeat che si occupa di smistare il traffico.

Questi plugin gestiscono i moduli di comunicazione visti in figura 16. In cui con "read processes" si intendono i processi dedicati lettura di messaggi da sottoporre al "Master Control Process", quindi riguardano le comunicazioni in ingresso attraverso i canali di comunicazione fra i nodi.

Mentre i "write processes", sono strutturalmente identici ai "read processes", ma sono dedicati all'invio di messaggi.

I ***plugin di autenticazione*** servono a garantire una certa sicurezza negli scambi tra i nodi. Tali plugin garantiscono che il

messaggio ricevuto da un nodo sia autentico ed effettivamente proveniente da un nodo appartenente allo stesso cluster. Tali meccanismi fanno uso solitamente di chiavi condivise. Esistono 3 diverse modalità di autenticazione: crc, sha1 e md5.
Il Resource Group che interagisce con i plugin in maniera diretta rappresenta l'insieme delle risorse gestite da Heartbeat. Una risorsa può essere ad esempio un indirizzo IP, un servizio web come Apache, un componente hardware come un disco o anche un intero file system. Per ogni risorsa Heartbeat imposta degli appositi script di start e stop. Il Resource Group è un concetto chiave, perché racchiude in sé il concetto di takeover da un nodo all'altro. Poiché Heartbeat è un gestore di servizi, per ogni nodo si avrà un certo gruppo di servizi da gestire, come un servizio web o anche una partizione di disco affidata a DRBD. Se si ha una interruzione il gruppo di servizi viene interamente migrato dal nodo primario a quello secondario e le risorse vengono riattivate su quest'ultimo.
Scendendo nei dettagli dell'esempio appena accennato che fa uso di una risorsa disco costituita da un device gestito da DRBD. Tale risorsa risulta essere identica a quella presente sul nodo caduto, in sé racchiude i dati utili per il funzionamento del sistema. Con l'operazione di takeover semplicemente viene cambiato il nodo sul quale gira l'applicazione che usa tali dati. Tale passaggio è reso così lineare da uno script di Heartbeat dedicato ai device DRBD, detto "datadisk".
I file di configurazione principali per Heartbeat sono:[27]

- ***ha.cf:*** questo file definisce tutti i settaggi del cluster, comprende la definizione dei nodi del cluster, dei metodi usati da Heartbeat per la comunicazione tra i nodi, i time-out da usare per i possibili failover.

- ***haresources:*** questo file serve a configuare le risposte che un nodo darà in seguito ad un failover. Questi provvedimenti potrebbero includere il mounting/ unmounting del filesystem o lo start/stop dei servizi.
- ***authkeys:*** questo file contiene una chiave segreta che tutti i nodi condividono. Questa chiave è usata come metodo di autenticazione tra i nodi, in modo che essi sappiano con quale protocollo di autenticazione stanno comunicando con gli altri membri del nodo.

6.3 DESCRIZIONE DELL'ESPERIENZA

6.3.1 OBIETTIVO: CREAZIONE DI UN CLUSTER HA

Il cluster Ha realizzato serve a garantire la continuità del servizio per l'applicazione web-based presentata al paragrafo 6.1.2. Per tale applicazione bisogna garantire il requisito di alta affidabilità per permettere il monitoraggio real-time di un impianto di stoccaggio di scorie radioattive. L'applicazione messa a punto dal team di sviluppo gira su HTTP Server Apache che espone i servizi all'esterno. L'applicazione è accessibile da qualunque web browser. Affinché essa sia sempre disponibile bisogna salva-guardare i dati, il database su cui essi vengono immagazzinati e il servizio server web per cui è sviluppata. Attraverso questo cluster HA in configurazione AS si vuole garantire l'affidabilità dei dati (tramite DRBD) e la continuità dei servizi (Heartbeat + Pacemaker). In modo che se il nodo primario cade l'applicazione viene resa del tutto disponibile sul secondario.

6.3.2 STRUMENTAZIONE HARDWARE UTILIZZATA

Segue un elenco delle componenti impiegate per la realizzazione del cluster HA oggetto di questa tesi e uno schema dell'architettura del caso di studio:

- due macchine identiche aventi le seguenti caratteristiche tecniche: processore Intel Core i5-2310 CPU 2,90 GHz x 4, Memoria 7,8 GiB, Disco fisso 976,0 GB (dev/sda7), Sistema operativo Ubuntu 11.10 (64 bit) dotate di due schede di rete (eth0 e eth1, in figura 18)

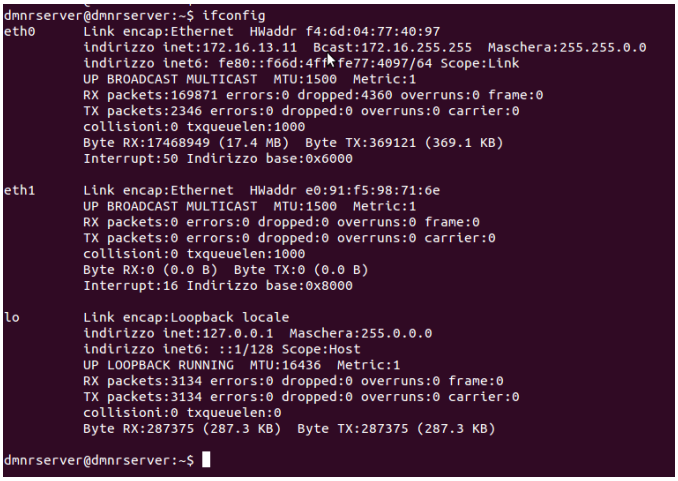

Figura 18: Schede di rete presenti su un nodo

- un cavo Ethernet cross
- due cavi Ethernet dritti
- uno switch

Ogni macchina ha due interfacce di rete che abbiamo denominato eth0, linea di connessione alla rete locale tramite switch, e eth1, linea di connessione tramite cavo cross che collega direttamente le due macchine costituenti i nodi del cluster. Sulla prima interfaccia è stato considerato l'indirizzo che identifica univocamente il nodo del cluster e tramite il quale i nodi sono collegati, mentre sulla seconda si considerano gli indirizzi privati

usati per la sincronizzazione di DRBD e per il controllo da parte di Heartbeat.

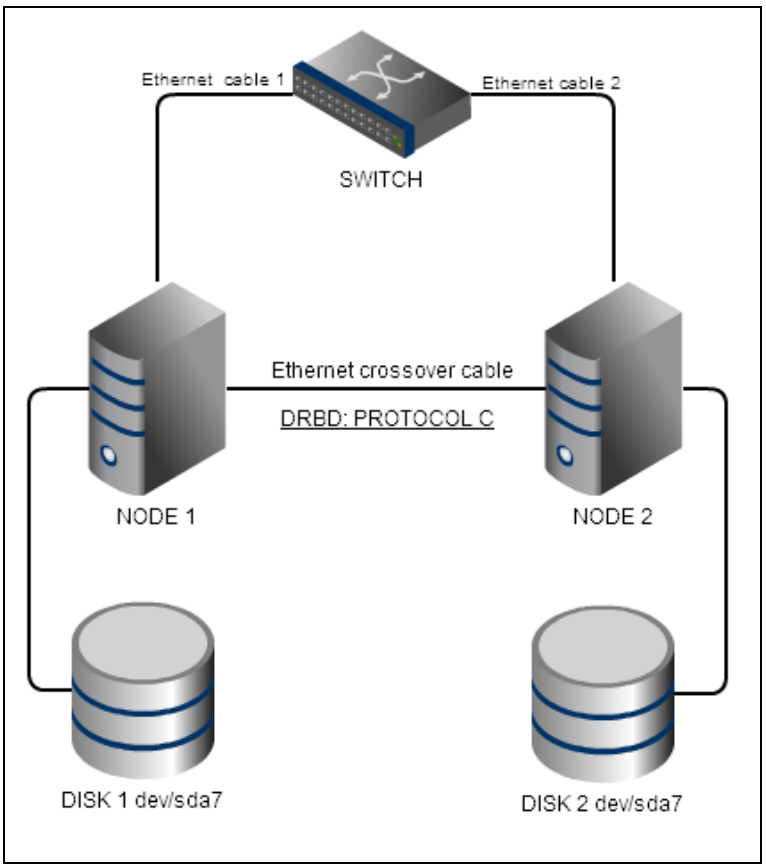

Figura 19: Cluster HA realizzato nel caso di studio

6.3.3 CONFIGURAZIONE DI DRBD

Il primo passo per la realizzazione del cluster è stata la configurazione di DRBD su entrambi i nodi. Dopo aver installato DRBD sui due nodi dai repository ufficiali di Ubuntu tramite il comando da terminale:

```
sudo apt-get install drbd8-utils
```

Si è proceduto al montaggio dello stesso come modulo kernel tramite il comando:

```
sudo modprobe drbd
```

Fatto ciò, si è passato a scegliere le aree di memoria che DRBD doveva gestire. Poiché le due macchine sono state impiegate appositamente per lo scopo non sono state create apposite partizioni, ma sono stati usati i due hard-disk nella loro interezza, pur avendo un'applicazione web dalle dimensioni esigue si è scelto di operare in questo modo per semplicità. Inoltre è stato a priori evitato il problema della sovrapposizione dei dati utili con gli internal meta-data (una possibile soluzione più semplice dell'impiego di partizioni dedicate agli external meta-data, suggerita sulla documentazione ufficiale di DRBD, è infatti quella di aumentare le dimensioni della partizione che ospita i dati da replicare).

Un requisito esplicitamente richiesto è che le due aree di memoria abbiano la stessa dimensione approssimativamente e se possibile anche gli stessi nomi, nel nostro caso come illustrato in figura 19 entrambe le aree di memoria sono denominate /dev/sda7. Su ogni nodo del cluster le due partizioni vengono destinate alla creazione di due DRBD block-device. La partizione sul nodo 1 sarà quella su cui vengono mantenuti i dati, che verranno poi replicati sulla partizione del nodo 2.

Il file che contiene la configurazione di DRBD è il file */etc/drbd.conf*. In tale file sono specificate le risorse da gestire con la relativa configurazione e le modalità con cui Drbd dovrà utilizzarle. Questo file deve essere identico sui due nodi, per cui una volta impostato è stato copiato senza apportare alcuna modifica sull'altro. All'interno del file */etc/drbd.conf* ci sono diverse sezioni specifiche per ogni block-device che si vuole creare sul cluster.

La configurazione di un block-device inizia con una riga che contiene il termine "resource", il nome arbitrario che lo segue identifica il device DRBD creato. In appendice si trova il file di

configurazione */etc/drbd.conf* così come è stato settato durante l'esperienza. Tale file si occupa solo di includere i file che contengono la vera e propria configurazione, cioè quei file del tipo "drbd.d/global_common.conf" e "drbd.d/*.res". In appendice si trova la configurazione utilizzata.
Mi soffermerò brevemente a chiarire il significato dei parametri settati per la configurazione in entrambi i file:

FILE global_common.conf:

usage-count no: si disabilita l'invio via http al sito ufficiale di DRBD dei dati per il conteggio delle installazioni.

syncer {rate 5M}: riguarda la configurazione della sincroniz-zazione tra i due device, il numero indica i megabyte trasferiti al secondo.

common {protocol: C} esegue la replica sincrona introdotta e approfondita al paragrafo 6.2.

FILE r0.res:

resource r0: nome arbitrario della risorsa

device /dev/drbd0: device virtuale DRBD associato al device fisico reale /dev/sda7, insieme formano la risorsa r0.

address 172.16.29.61:7789: Indirizzo IP del nodo e porta TCP di ascolto

Dopo aver configurato il file drbd.conf come visto sono state eseguite in ordine le seguenti operazioni. Le operazioni sono state eseguite sfruttando il tool di amministrazione a riga di comando "drbdadm".

- ***CREAZIONE META-DATA DEVICE:*** questa operazione è stata fatta solo una volta su entrambi i nodi, serve per inizializzare i meta-data riservando loro uno spazio in fondo al dispositivo di memoria a blocchi.

```
drbdadm create-md r0
```

- ***ABILITAZIONE DELLA RISORSA:*** in questa fase si la partizione fisica /dev/sda7 è stata associata al device astratto /dev/drbd0, sono stati settati i parametri discussi precedentemente ed è stato eseguito un primo collega-mento tra i due nodi.

```
drbdadm up r0
```

A questo punto DRBD risultava essere configurato e funzionante. Per portare DRBD alla completa operatività si è proceduto a selezionare una fonte di sincronizzazione tra i due nodi, da cui far partire la sincronizzazione completa, dal momento che i tutti i dati riguardanti l'applicazione web per cui è progettato il cluster si trovavano originariamente su questa partizione. L'altra partizione era vuota e pronta per essere scritta con i suddetti dati. A questo punto l'operazione che segue è stata eseguita solo sul nodo scelto come primario.

```
drbdadm -- --overwrite-data-of-peer primary r0
```

Prima di questa operazione, dal momento che non si erano ancora scelti i ruoli tra i due nodi la situazione era la seguente: i nodi si vedevano entrambi nel cluster ma risultavano entrambi secondari e i dati risultavano in uno stato di "inconsistenza"

riferito alla totale assenza di operazioni valide su di essi. Il contenuto del file /proc/drbd, che contiene info riguardanti lo status di DRBD, a questo punto era quello rappresentato in figura 20.

Figura 20: Connessione nodi DRBD. Status: Secondary/Secondary

Dopo aver eseguito l'ultimo comando da terminale ha avuto inizio la sincronizzazione, i dati presenti sul nodo primario vengono replicati sul secondario.

Alla fine della configurazione e attivazione di DRBD il nodo primario può essere usato senza restrizioni, su di esso il device astratto /dev/drbd0 può essere usato come un normale device a blocchi, quindi può essere formattato e su di esso si può montare un file system. Tutte le modifiche apportate su di esso verranno replicate sul nodo secondario. Il mounting di questa partizione, come detto più volte in questo tipo di configurazione, è possibile solo su un unico nodo. Di volta in volta sarà solo il nodo che si trova nella condizione di primario a poter montare il file system su /dev/drbd0/.

Il mounting di un file system ext3 è stato eseguito sul nodo primario con i seguenti comandi da terminale:

```
mkfs.ext3 /dev/drbd0
mount /dev/drbd0
```

I ruoli dei nodi possono comunque essere invertiti, in modo da dare la possibilità di leggere e scrivere su /dev/drbd0 anche al nodo secondario, anche se non si è in una situazione di takeover. Il procedimento è inverso a quello appena visto, vale a dire che si procede con l'unmount del device DRBD sul primario, per poi dichiarare questo secondario e l'altro primario. I comandi per il nodo primario attuale sono:

```
unmount /dev/drbd0
drbdadm secondary r0
```

Per il nodo secondario, promosso a primario:

```
drbdadm primary r0
mount /dev/drbd0
```

6.3.4 CONFIGURAZIONE DI HEARTBEAT E PACEMAKER

Heartbeat e Pacemaker vengono forniti anch'essi nei repository ufficiali di Ubuntu.

```
sudo apt-get build-dep heartbeat-2
```

La configurazione del demone Heartbeat e del relativo CRM Pacemaker è presente in appendice. La differenza principale rispetto alla configurazione senza CRM è che il file authkeys non serve dal momento che viene abilitato Pacemaker, che come visto fornisce dei plugin per l'autenticazione. Pacemaker è abilitato tramite l'inserimento della riga: "crm respawn" alla fine del file di configurazione ha.cf. I parametri principali settati in ha.cf sono stati:

autojoin none: disabilita la funzione automatica di scoperta ed aggiunta di nodi sulla rete

autofailback on: serve a far sì che dopo un takeover, quando il nodo primario precedentemente caduto torna attivo si abbia un nuovo switch che riporti i ruoli come impostati in origine.

ping: utilizza un IP che verifica se il nodo fa parte della rete ed è connessa ad esso. Deve essere un IP affidabile, come quello del gateway di rete per esempio.

warntime: è il tempo dopo il quale Heartbeat decide che un nodo potrebbe essere morto.

deadtime: il tempo dopo cui Heartbeat può affermare che un nodo è morto.

keepalive: il tempo durante il quale i pacchetti inviati fra i nodi restano in vita.

Negli ultimi tre parametri i numeri associati esprimono il tempo in secondi.
Prima di avviare il demone è stato controllato che il servizio apache fosse disattivato, dal momento che solitamente è avviato al boot del sistema. La sua gestione deve essere eseguita da Heartbeat.
Il file haresources contiene la lista di risorse che vengono spostate da un nodo all'altro. La prima risorsa consiste nel blocco DRBD numero 0 montato in /dev/drbd0 con filesystem ext3. Su tale risorsa viene attivato il servizio apache.
La configurazione di Heartbeat su DRBD permette che il device virtuale /dev/drbd0/ sia gestito dal cluster:

```
crm configure primitive drbd0 ocf:linbit:drbd params drbd_resource="r0" op monitor interval="20s" timeout="40s" op start interval="0" timeout="240s" op stop interval="0" timeout="100s"
```

Tramite il seguente comando r0 viene definita sul nodo primario come "risorsa multi-state", in modo tale che questo possa comunicare agli altri nodi il ruolo assunto, ed evitare che la risorsa risulti contesa tra i nodi.

```
# crm configure ms ms-drbd0 drbd0 meta master-max="1" notify="true"
```

Dopo aver configurato Heartbeat su una delle due macchine, la configurazione è stata propagata sulla rimanente utilizzando il comando:

```
/usr/lib/heartbeat/ha_propagate
```

Fatto ciò non solo i file di configurazione sono stati duplicati sull'altro nodo, ma i nodi risultano connessi.
Con il seguente comando è stato avviato per la prima volta il demone:

```
/etc/init.d/heartbeat start
```

La riga che abilita Pacemaker sul file ha.cf è servita ad avviarlo. Dopo pochi secondi era presente tra i processi attivi sul monitor di sistema e funzionava totalmente. Il file di log ha-log, in appendice, fa vedere l'effettivo funzionamento del cluster realizzato. Inoltre da terminale, tramite il seguente comando crm si è avuto il seguente risultato:

```
# crm status
============
Last updated: Tue Sep 14 12:17:29 2010
Stack: Heartbeat
Current DC: gianfri-laptop (1627f3ec-ec7e-4f0c-b69e-e9729933a2cc) -
partition with quorum
Version: 1.0.8-042548a451fce8400660f6031f4da6f0223dd5dd
2 Nodes configured, unknown expected votes
1 Resources configured.
============
```

```
Online: [ gianfri-laptop gianfranco-client ]

 Clone Set: ping_clone
     Started: [gianfri-laptop gianfranco-client]
```

6.3.5 TEST DI FUNZIONAMENTO

Primo test: disconnessione cavo LAN nodo master

Alla disconnessione del cavo la risorsa ping avverte immediatamente l'assenza di connettività e forza la migrazione delle risorse all'altro nodo, in questo caso da gianfri-laptop a gianfranco-client. Il tutto è osservabile dal file /var/log/syslog, presente in appendice.

La scrittura sul device condiviso si ferma per qualche secondo e riparte appena l'ultima operazione di start è eseguita da Pacemaker.

Secondo test: migrazione manuale delle risorse

Lanciando il comando migrate viene forzato manualmente lo switch della risorsa dal nodo attuale al nodo indicato:

```
crm resource migrate share-a gianfri-laptop
```

Nel log il comportamento è simile al precedente test e la scrittura sulla risorsa condivisa si ferma il tempo che la risorsa migra.

Terzo test: spegnimento improvviso nodo master

Lo spegnimento improvviso del nodo master rende il comportamento del cluster del tutto simile a quello illustrato primo test. Il cluster ha dovuto rimpiazzare il primario poiché spento ed ovviamente il dispositivo drbd è in stato primary sul nodo sopravvissuto. Anche in questo caso, tutto si è svolto in maniera trasparente, la copia si è fermata per qualche secondo per poi ripartire.

CAPITOLO 7: Conclusioni

7.1 POSSIBILI MIGLIORAMENTI

I risultati ottenuti dall'esperienza svolta presso i Laboratori Nazionali del Sud possono essere considerati soddisfacenti. L'obiettivo di garantire, in seguito a malfunzionamenti, errori o piccoli incidenti locali, cioè la continuità dei servizi offerti dall'applicazione web dedicata al progetto DMNR si può considerare raggiunto. Ovviamente come in ogni esperienza è sempre possibile un reale miglioramento. Il tocco finale di questa installazione prevede una linea seriale su entrambe le macchine, collegata ad incrocio ad un gruppo di continuità. Mediante l'aggiunta dell'utility STONITH (shoot the other node in the head!) presente nel pacchetto di Heartbeat, è possibile, in caso di failover, far sì che il server rimasto tolga letteralmente la corrente all'altro, dopo averne preso in carico i servizi. Questo evita un eventuale ed inatteso 'ritorno in vita' del server defunto con conseguente conflitto di IP e di porte (almeno, fino a che un amministratore non provvede a ripristinare la situazione iniziale), risolvendo il problema dello split-brain.

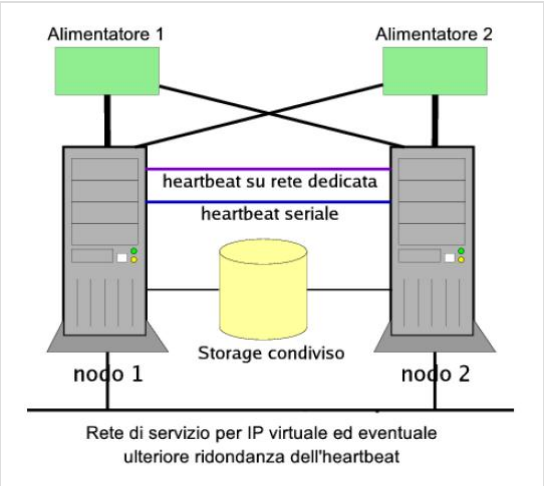

Figura 21: Eventuali miglioramenti per la soluzione adottata

7.2 SVILUPPI FUTURI

Gli sviluppi futuri per una simile problematica sembrano mirare sempre più al Cloud Computing. Il Cloud computing è un modello emergente per il business, dati e i programmi non devono necessariamente risiedere su una macchina fisica; possono infatti essere "hosted in the cloud". Il lavoro viene distribuito su un grande numero di computer. Le applicazioni possono sfruttare la capacità di calcolo, lo spazio di storage e i servizi della "nuvola". Anche in questo ambito vengono valutati i fattori di rischio derivanti da possibili interruzioni. In modo da soddisfare il requisito di continuità delle applicazioni che usano la nuvola e la sicurezza dei dati, viene definito un livello di "fault tolerance" per evitare che vi siano single point of failure che facciano perdere dati.

La struttura tipica di un disaster recovery system basato sul cloud computing è detta "distributed computing centralized storage". Questo tipo di organizzazione ha la caratteristica di avere un'alta riusabilità, scalabilità e di avere alte performance. Per le organizzazioni che basano il loro business su risorse molto ampie sembra essere una buona soluzione.

Il termine Cloud Storage non si riferisce ad uno specifico device, ma a un aggregato di numerosi device di storage, questo rappresenta il punto di forza rispetto alle tecniche tradizionali di clustering.

I sistemi tradizionali di Disaster Recovery, illustrati in dettaglio al capitolo 3, usano essenzialmente la replica del sistema primario in diverse aree geografiche. Questo approccio come visto viene affrontato a diversi livelli, con un investimento più o meno considerevole di risorse. Negli ultimi anni queste soluzioni stanno cadendo in disuso, poiché vengono reputate costose. Nel caso

infatti di grandi organizzazioni, le sole infrastrutture atte a sostenere un probabile Disaster Recovery in seguito ad un incidente, consumano risorse economiche, energetiche ed umane. Molte delle organizzazioni che reputano di avere dei Disaster Recovery Plan all'avanguardia utilizzano dei sistemi distribuiti di questo tipo. Un report di IBM afferma che nei sistemi distribuiti nell'85% del tempo il sistema sta in idle, in attesa di poter essere utilizzato in qualche missione di recovery, consumando solo energia elettrica. Gli svantaggi delle soluzioni tradizionali sono: costi alti, spreco di energia, sottoutilizzo delle infrastrutture e scarsa riusabilità delle stesse. Per questi motivi, molte organizzazioni stanno puntando a un nuovo modo di progettare il Disaster Recovery tramite il cloud computing. Negli ultimi anni sono stati proposti dei piani di recovery cloud-based, i quali però sono al momento alla portata solo delle grandi organizzazioni, mentre le piccole organizzazioni preferiscono affidarsi ai metodi tradizionali. Il disaster recovery cloud-based implica l'utilizzo di facilities di recovery che non si basano su costosi server dedicati, grandi risorse umane o altri tipi di infrastrutture tipiche dei metodi tradizionali (spazio, energia elettrica, larghezza di banda per le trasmissioni, impianti di cooling).Il cloud-based disaster recovery plan è una alternativa molto ambiziosa e innovativa per l'alta affidabilità a bassi costi. Il Cloud Computing infatti non solo promuove l'alta affidabilità, ma anche la qualità dei servizi e un impiego minore di risorse per interventi di manutenzione.[28][29]

BIBLIOGRAFIA E SITOGRAFIA:

[1] Official ISC, International Information Systems Security Certification Consortium, http://www.isc2.org/

[2] "Business Continuity Planning (BCP) Methodology-Essential for every business", Dr. Manik Dey PhD, 2011 IEEE GCC Conference and Exibition(GCC), February 2011, Dubai (United Arab Emirates)

[3] "Certificazione Sistemi di Gestione per la Continuità Operativa Norma BS 25999-2", IMQ - Istituto Italiano per il Marchio di Qualità, http://www.imq.it/

[4] "IBM Geographic Remote Mirror for AIX - Concepts and Facilities", IBM, fourth edition, 2002, Poughkeepsie, NY (USA)

[5] "Disaster Recovery Plan", Telesio Perfetti, 2005 http://www.computerlaw.it/

[6] "Disaster Recovery Planning & Methodology for Process Automation Systems", Fouad Al-Khabbaz, Hussain Al-Zahir, Salem Elwi, Hassan Al-Yousef, EUROCON - International Conference on Computer as a Tool (EUROCON), 2011 IEEE

[7] "Selecting Technology for Disaster Recovery", R.Cegiela, Institute of Control & Computer Engineering, Warsaw (POLAND), DepCos-RELCOMEX '06, International Conference on, May 2006

[8] "IBM TotalStorage Solutions for Disaster Recovery", Cathy Warrick, Rainer Eisele, Ray Pratts, John Sing, Ulrich Walter, and Ian R. Wright. - Redbooks IBM, second edition, San Jose, CA (USA), January 2004

[9] "Disaster Recovery Strategies with Tivoli Storage Management", Charlotte Brooks, Matthew Bedernjak, Igor Juran, and John Merryman - Redbooks. IBM, second edition, San Jose, CA, (USA), November 2002

[10] "Blueprints for High Availability", Marcus Evan and Hal Stern, Wiley Publishing, Indianapolis, IN, USA, second edition, 2003

[11] "Electronic Vaulting: Facilitating the New Era of Rapid Recovery", John Lindeman, http://www.disaster-resource.com/articles/electric_vault_rapid_lindeman.shtml

[12] "Network Planning for Disaster Recovery" ,Andrea Bianco, Jorge Finochietto, Luca Giraudo, Marco Modesti, Fabio Neri, Dip. di Elettronica, Politecnico di Torino, Italy, Universidad Nacional de Cordoba - CONICET, Argentina, "Local and Metropolitan Area Networks-LANMAN 2008", 2008 - 16th IEEE Workshop on

[13] "Distributed Operating Systems: concept and design", Pradeep K. Sinha, IEEE Press New York, NY, 1997

[14] "A High Availability and Disaster Recovery System", Qin Zhang, Hong Xu, 2008 IEEE Conference, on date:21-24 Sept. 2008

[15] "Evoluzione dell'alta affidabilità su Linux", Seminario di MMUL, 24 giugno 2011, http://www.miamammausalinux.org/

[16] Massimiliano Filacchioni, CASPUR (Consorzio interuniversitario per le Applicazioni di Supercalcolo Per Università e Ricerca), 29 Maggio 2004,"Settimana del Software Libero"

[17] RAID, http://it.wikipedia.org/wiki/RAID

[18] "Is Five 9's Availability Required for Internet Services?", S. Smaldone, http://www.cs.rutgers.edu

[19] "Internet Services need high availability", R.Martin, http://www.cs.rutgers.edu

[20] GarterItalia,http://www.gartner.com/regionalization/content/emea/it/home.jsp

[21] Dimensionamento, http://it.wikipedia.org/wiki/Dimensionamento

[22] "DMNR: A new concept for real-time online monitoring of short and medium term radioactive waste", P.Finocchiaro, 1022 Nova Science Publishers, Inc. (in print)

[23] Linbit, http://www.linbit.com/

[24] "What is DRBD", http://www.drbd.org/

[25] CRM CLI, http://www.clusterlabs.org/

[26] Linux-HA Project, http://www.linux-ha.org/

[27] "The official Ubuntu server book", Kyle Rankin, Benjamin Mako Hill — 2nd edition, 2010

[28] "Disaster Recovery for System Architecture using Cloud Computing", Manish Pokharel, Seulki Lee, Jong, Sou Park 2010 10th Annual International Symposium on Applications and the Internet

[29] "Cloud Computing-based Data Storage and Disaster Recovery", Zhang Jian-hua and Zhang Nan, School of Computer Science and Technology South-west University for Nationalities Chengdu, China, International Conference on Future Computer Science and Education, 2011

APPENDICE

A.1 FILE CONFIGURAZIONE DRBD

FILE drbd.conf:

```
# You can find an example in
/usr/share/doc/drbd.../drbd.conf.example
include "drbd.d/global_common.conf";
include "drbd.d/*.res";
```

FILE global common.conf:

```
global {
     usage-count no;
     # minor-count dialog-refresh disable-ip-
verification
}

common {
     protocol C;

     handlers {
          pri-on-incon-degr "/usr/lib/drbd/notify-pri-
on-incon-degr.sh; /usr/lib/drbd/notify-emergency-
reboot.sh; echo b > /proc/sysrq-trigger ; reboot -f";
          pri-lost-after-sb "/usr/lib/drbd/notify-pri-
lost-after-sb.sh; /usr/lib/drbd/notify-emergency-
reboot.sh; echo b > /proc/sysrq-trigger ; reboot -f";
          local-io-error "/usr/lib/drbd/notify-io-
error.sh; /usr/lib/drbd/notify-emergency-shutdown.sh;
echo o > /proc/sysrq-trigger ; halt -f";
          # fence-peer "/usr/lib/drbd/crm-fence-
peer.sh";
          # split-brain "/usr/lib/drbd/notify-split-
brain.sh root";
          # out-of-sync "/usr/lib/drbd/notify-out-of-
sync.sh root";
          # before-resync-target
"/usr/lib/drbd/snapshot-resync-target-lvm.sh -p 15 -- -
c 16k";
          # after-resync-target
/usr/lib/drbd/unsnapshot-resync-target-lvm.sh;
     }

     startup {
```

```
          # wfc-timeout degr-wfc-timeout outdated-wfc-
timeout wait-after-sb
     }

     disk {
          # on-io-error fencing use-bmbv no-disk-
barrier no-disk-flushes
          # no-disk-drain no-md-flushes max-bio-bvecs
     }

     net {
          # sndbuf-size rcvbuf-size timeout connect-int
ping-int ping-timeout max-buffers
          # max-epoch-size ko-count allow-two-primaries
cram-hmac-alg shared-secret
          # after-sb-0pri after-sb-1pri after-sb-2pri
data-integrity-alg no-tcp-cork
     }

     syncer {
          rate M5
     }
}
```

FILE r0.res:

```
resource r0{

on gianfranco-client {
     device /dev/drbd0;
     disk /dev/sda7;
     address 172.16.29.61:7789;
     meta-disk internal;
     }


on gianfri-laptop {
     device /dev/drbd0;
     disk /dev/sda7;
     address 172.16.29.17:7789;
     meta-disk internal;
     }
}
```

A.2 FILE CONFIGURAZIONE HEARTBEAT

FILE ha.cf:

```
#
#    There are lots of options in this file. All you
have to have is a set
#    of nodes listed {"node ...} one of {serial, bcast,
mcast, or ucast},
#    and a value for "auto_failback".
#
#    ATTENTION: As the configuration file is read line
by line,
#            THE ORDER OF DIRECTIVE MATTERS!
#
#    In particular, make sure that the udpport, serial
baud rate
#    etc. are set before the heartbeat media are
defined!
#    debug and log file directives go into effect when
they
#    are encountered.
#
autojoin none
autofailback on
keepalive 1
deadtime 10
warntime 5
initdead 20
mcast eth0 239.0.0.43 694 1 0
bcast eth1
node    gianfri-laptop
node    gianfranco-client
crm respawn
```

FILE authkeys:

```
#    Authentication file.  Must be mode 600
#
#
#    Must have exactly one auth directive at the front.
#    auth  send authentication using this method-id
#
#    Then, list the method and key that go with that method-id
#
#    Available methods: crc sha1, md5.  Crc doesn't need/want
a key.
#
```

```
#     You normally only have one authentication method-id
listed in this file
#
#     Put more than one to make a smooth transition when
changing auth
#     methods and/or keys.
#
#
#     sha1 is believed to be the "best", md5 next best.
#
#     crc adds no security, except from packet corruption.
#           Use only on physically secure networks.
#
auth 1
1 crc
#2 sha1
#3 md5
```

FILE haresources:

```
# /etc/ha.d/haresources
left 193.206.185.10 \
  drbddisk::drbd0 \
  Filesystem::/dev/drbd/0::ext3\
  drbdlinks-http apache
```

FILE /var/log/syslog:

```
Nov 04 17 18:05:35 gianfranco-client heartbeat: [1079]: info: Link gianfri-
laptop:eth0 dead.
Nov 04 18:05:46 gianfranco-client kernel: [20384.996634] block drbd0: peer(
Primary -> Secondary )
Nov 04 18:05:50 gianfranco-client Filesystem[27581]: [27636]: INFO: Running start
for /dev/drbd0 on /share-a
Nov 04 18:05:50 gianfranco-client kernel: [20389.636665] EXT3-fs: mounted
filesystem with ordered data mode.
Nov 04 18:05:51 gianfranco-client exportfs[27676]: [27739]: INFO: File system
exported
Nov 04 18:05:52 gianfranco-client IPaddr2[27764]: [27817]: INFO: ip link set eth0
up
```

FILE ha-log:

```
Nov 04 15:27:19 gianfri-laptop heartbeat: [1480]: info:
*************************
Nov 04 15:27:19 gianfri-laptop heartbeat: [1480]: info:
Configuration validated. Starting heartbeat 3.0.5
Nov 04 15:27:19 gianfri-laptop heartbeat: [1482]: info: heartbeat:
version 3.0.5
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: info: Heartbeat
generation: 1320403818
```

```
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: info: glib: UDP
Broadcast heartbeat started on port 694 (694) interface eth0
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: info: glib: UDP
Broadcast heartbeat closed on port 694 interface eth0 - Status: 1
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: info: Local
status now set to: 'up'
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: info: Link
gianfri-laptop:eth0 up.
Nov 04 15:27:22 gianfri-laptop heartbeat: [1482]: WARN:
G_CH_dispatch_int: Dispatch function for read child took too long
to execute: 180 ms (> 50 ms) (GSource: 0x209c850)
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: WARN: node
gianfranco-client: is dead
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: info:
Comm_now_up(): updating status to active
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: info: Local
status now set to: 'active'
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: WARN: No STONITH
device configured.
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: WARN: Shared
disks are not protected.
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: info: Resources
being acquired from gianfranco-client.
harc[2306]: 2011/11/04_15:29:23 info: Running
/etc/ha.d//rc.d/status status
mach_down[2336]:  2011/11/04_15:29:23 info:
/usr/share/heartbeat/mach_down: nice_failback: foreign resources
acquired
mach_down[2336]:  2011/11/04_15:29:23 info: mach_down takeover
complete for node gianfranco-client.
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: info: mach_down
takeover complete.
Nov 04 15:29:23 gianfri-laptop heartbeat: [1482]: info: Initial
resource acquisition complete (mach_down)
IPaddr[2375]:     2011/11/04_15:29:23 INFO:  Resource is stopped
Nov 04 15:29:23 gianfri-laptop heartbeat: [2307]: info: Local
Resource acquisition completed.
harc[2418]: 2011/11/04_15:29:23 info: Running /etc/ha.d//rc.d/ip-
request-resp ip-request-resp
ip-request-resp[2418]:  2011/11/04_15:29:23 received ip-request-
resp 127.0.1.1 OK yes
ResourceManager[2439]:  2011/11/04_15:29:23 info: Acquiring
resource group: gianfri-laptop 127.0.1.1 httpd
IPaddr[2466]:     2011/11/04_15:29:23 INFO:  Resource is stopped
ResourceManager[2439]:  2011/11/04_15:29:23 info: Running
/etc/ha.d/resource.d/IPaddr 127.0.1.1 start
IPaddr[2524]:     2011/11/04_15:29:23 INFO: Using calculated nic
for 127.0.1.1: lo
IPaddr[2524]:     2011/11/04_15:29:23 INFO: Using calculated
netmask for 127.0.1.1: 255.0.0.0
IPaddr[2524]:     2011/11/04_15:29:23 INFO: eval ifconfig lo:0
127.0.1.1 netmask 255.0.0.0 broadcast 127.255.255.255
IPaddr[2512]:     2011/11/04_15:29:23 INFO:  Success
```