ISTITUTO NAZIONALE DI FISICA NUCLEARE

CNAF

INFN-13-15/CNAF
11st September 2013

Disaster Recovery INFN: Configurazione di una soluzione High Availability con Oracle Data Guard 11g

C. Galli

CNAF-Centro Nazionale per la ricerca e lo sviluppo nelle tecnologie informatiche e telematiche dell'INFN, Viale Berti Pichat 6/2, 40127 Bologna (BO), Italia

Abstract

Questo documento ha lo scopo di presentare in forma sintetica gli step necessari alla configurzione e realizzazione di un ambiente di High Availability con particolare riferimento ai contesti Oracle Database 11g così come verrà implementato nell'ambito del progetto di Disaster Recovery dell'INFN.
Dopo una breve introduzione sulla soluzione tecnica adottata per la configurazione dell'ambiente H.A., ovvero Oracle Data Guard 11g, vengono presentati gli step necessari alle fasi di Setup iniziale, Avvio della replica (locale o geografica), Verifica, gestione di Faiver o Switchover e Monitoraggio.
Nei rispettivi capitoli verranno inoltre presentate le eventuali differenze, rispetto al caso standard, che possono avere luogo in base alle diverse conformazioni degli ambienti di lavoro Primary e Standby.
In ultimo nelle appendici vengono presentate anche alcuni consigli come ad esempio la configurazione di un canale sicuto TCPS/SSL per le connessioni tra Primary e Slave nel caso di un collegaemnto geografico su reti aperte.

Published by ***SIDS–Pubblicazioni***
Laboratori Nazionali di Frascati

1 Introduzione

In questa sezione viene riportata brevemente una panoramica dei concetti fondamentali su Oracle Data Guard e su alcune delle possibili soluzioni architetturali realizzabili tramite questo tool. Per prima cosa definiamo cosa si intende per *Disaster*, per *Disaster Recovery* (DR.) e *High Availability* (HA.), ed infine per *Business Continuity* (BC.).

Disaster: genericamente si può definire come un qualsiasi evento inaspettato che colpisce un sistema distruggendone alcune parti e ne impedendone il regolare funzionamento. La definizione di tale evento non si limita al settore hardware e software, come ad esempio la rottura di un disco di storage, bensì va estesa anche a calamità naturali ed ai danni a strutture e persone. Questi eventi infatti possono incidere in maniera decisiva sulla capacità di un sistema, o più in generale di un processo, di continuare a erogare correttamente i servizi di cui è composto. L'allagamento di una sala calcolo, l'improvvisa mancanza di energia elettrica, la rottura dei generatori di emergenza, rientrano indubbiamente tutte nella definizione data.

Disaster Recovery: si intende l'insieme di tutte le infrastrutture, le tecnologie e le procedure che, a fronte di un evento disastroso, permettono di ripristinare i servizi sul sito originale o su un nuovo sito.

High Availability: letteralmente indica 'alta disponibilità di servizio'. Si intende la capacità di un sistema di far fronte a guasti, imprevisti o anche semplici necessità di manutenzione, massimizzando il tempo di servizio erogato. Si può definire in maniera duale anche come la capacità di minimizzare il tempo di intervento necessario al ripristino dei servizi a fronte di guasti, eventi imprevisti o attività di manutenzione.

Business Continuity: è fortemente correlato con i due precedenti concetti dei quali si avvale per offrire architetture in grado di continuare a erogare servizi ininterrottamente (o quasi) anche a fronte di eventi disastrosi.

Oracle Data Guard 11g è la soluzione proposta da Oracle per far fronte ad esigenze di DR. Ovviamente ci sono una serie di parametri da tenere in considerazione [6], che influenzano fortemente la qualità delle architetture Data Guard implementabili e dei risultati ottenibili. Questa tecnologia infatti offre molteplici configurazioni per far fronte alle diverse esigenze di replica e protezione dati.

Ci limitiamo a considerare nel dettaglio una implementazione di Data Guard che prevede l'utilizzo di un database *slave* (*Physical Standby, single instance* [1.2 pag. 7]) che ospiterà la replica geografica di un database *master* (*Primary database, single instance*). Per completezza daremo anche una breve descrizione di altri due parametri fondamentali delle architetture DR; l'RTO (*Recovery Time Objective*) e l'RPO (*Recovery Point Objective*) [sez. 1.4 pag. 13].

In seguito vedremo sia le diverse modalità di protezione dei dati, sia le diverse tipologie di Standby realizzabili. Inoltre dalla prossima sezione [sez.2 pag.15] verrà presentata una configurazione che è stata testata su un ambiente distribuito tra le due sedi di **LNF** [1] e **CNAF** [2]; questo costituisce anche un case study per le future operazioni di replica al CNAF del sistema **GODiVA** [3] presso LNF.

Ovviamente nel seguito verranno omessi tutti i riferimetni espliciti a dati sensibili (come nomi di macchine, indirizzi IP, username/passwd ecc...) sostituendoli con elementi puramente descrittivi.

bg ty

1.1 Introduzione a Oracle Data Guard

Oracle Data Guard è una tecnologia Oracle disponibile nativamente sotto licenza Enterprise dalla versione 9i in poi. Consiste in una serie di particolari configurazioni tra varie istanze di database che permettono di realizzare architetture di replica dati. In particolare una configurazione Data Guard implica almeno un sito

[1] *Laboratori Nazionali di Frascati*

[2] *Centro Nazionale Analisi Fotogrammi* a Bologna

[3] GODiVA (Gestione Ospiti, DIpendenti, Visitatori ed Associati) è il nuovo software per la gestione delle Identità e dei diritti di Accesso (Identity and Access Management o IAM) sviluppato nell'ambito del progetto INFN-AAI della Commissione Calcolo e Reti INFN

primario, che definiremo *Primary*, ed un sito secondario ove sia allocato un database con lo scopo di recepire dal Primary tutte le modifiche ai dati e riapplicarle ad una istanza locale; chiameremo questa seconda istanza *Standby*.

Fino alla versione 10gR2, sotto certe condizioni di replica, le architetture hardware e software dovevano necessariamente essere le medesime sia sul nodo Primary che su quello di Standby. Dalla versione 11g in poi le restrizioni in tal senso si sono ridotte notevolmente e ad oggi esistono percorsi di migrazione che prevedono repliche intermedie con Data Guard come step di passaggio da una architettura all' altra. La matrice completa delle compatibilità, con anche i requisiti minimi, è disponibile sul sito di Oracle Support [4] e ad essa si deve fare riferimetno per i dettagli. E' comunque sempre buona norma utilizzare la stessa versione del software oracle (la ORACLE_HOME) sia sul Prymary che su ogni Standby ad esso collegato.

Dal punto di vista tecnico la base del funzionamento di Data Guard 11g non è molto dissimile dalla versione 9i originale ed è estremamente semplice nel concetto [Figura 1 pag. 6].

Da un lato un processo LNS *(Log Network Server)* si fa carico sul Primary di raccogliere tutti gli ORL *(Online Redo Log)*[4] e spedirli allo Standby. Dall' altro lato il processo RFS *(Remote File Server)* sullo Standby salva localmente gli OLR ricevuti come SRL *(Standby Redo Log)*. Infine un processo di "Apply" sullo Standby si fa carico di leggere gli SRL e di riapplicarli all'istanza che si trova costantemente in "recovery mode".

Poichè negli ORL del Primary (quindi anche negli SRL) ci sono tutti gli elementi per ricostruire le transazioni committate, questo meccanismo permette di replicare "byte per byte" ogni singola azione completata correttaemnte sul Primary. Ovviamente transazioni non committate al momento del guasto verranno scartate dallo Standby, mentre eventuali transazioni che erano soggette a "rollback" sul Primary, saranno soggette ad equivalente rollback sullo Standby.

Nelle Architetture 9i e 10g viene presentato un'altro processo chiamato ARCH *(Archiver)* che, nel Primary, si occupa di archiviare gli ORL e di spedirli allo

[4]Solo quando il LGWR *(Log Writer)* termina di scrivere sull' ORL current, e passa a scrivere il successivo ORL, comunica all' LNS la disponibilità di un nuovo OLR da spedire.

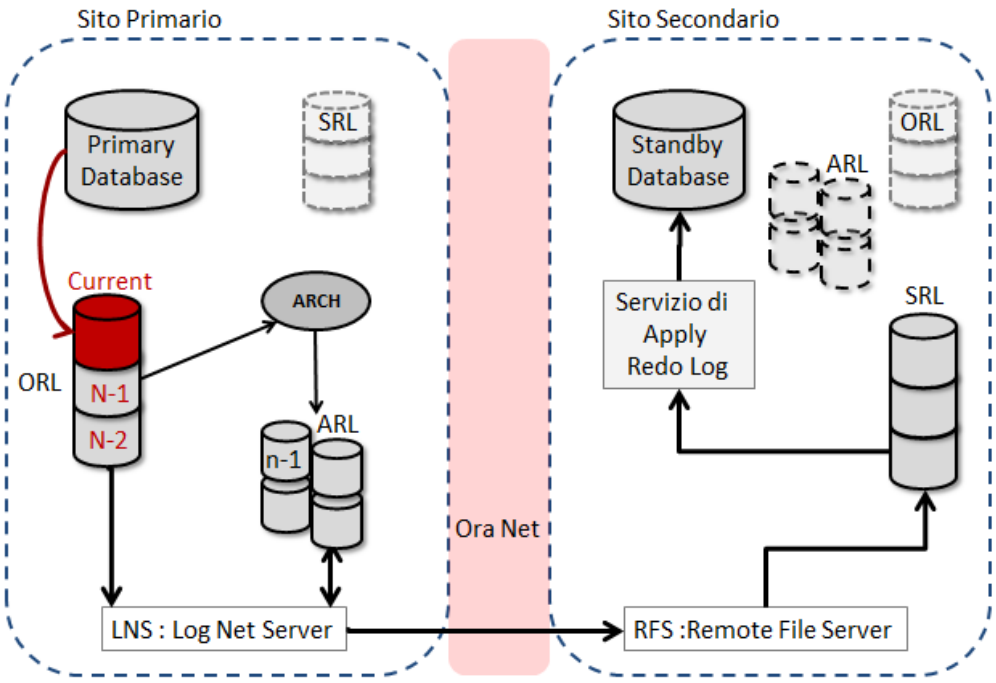

Figura 1: Overview di un ambiente Oracle Data Guard 11g

Standby, mentre nello Standby si occupa di riapplicare gli ARL *(Archive Redo Log)* ai datafile. Nella versione 11g di Data Guard l'utilizzo di ARCH per lo strato di trasporto è deprecato, pur tuttavia rimanendo disponibile la funzionalità di archiviazione degli ARL per retro-compatibilità. Come vedremo in seguito, nelle architetture Max Availability [sez. 1.3.2 pag. 11], quando sopraggiungono delle condizioni di disallineamento (*"gap"*), la presenza dell'ARCH sul Primary consente al processo LGWR di continuare a utilizzare ciclicamente gli ORL, grazie al fatto che l'ARCH continua a generare i relativi ARL che utilizzerà poi per la risoluzione del gap. Dualmente è presente lo stesso processo sullo Standby per la ricezione e applicazione degli ARL necessari nella fase di riallineamento.

Nella *Figura (1)* si può notare come sul Primary siano presenti anche degli SRL inutilizzati. Questi servono qualora l'istanza sia soggetta alle operazioni di riconfigurazione necessarie per una inversione di ruolo tra Primary e Standby [sez.1.5 pag.14]. Ugualmente possono essere presenti degli ARL sullo Stand-

by per esigenze di risoluzione di gap come già accennato, ma anche qualora si volessero gestire i backup sullo Standbuy e non direttamente sul Primary. Il meccanismo presentato si può quindi complicare notevolmente in base a quale tipo di Standby e quale politica di protezione dati vengano implementati.

1.2 Tipi di Standby Database

Con Oracle Data Guard è possibile creare principalmente due diversi tipi di Standby, quello definito "Physical Standby" e quello definito "Logical Standby". Qui di seguito vengono riassunte le principali caratteristiche di entrambe le tipologie di Standby con alcune considerazioni di tipo architetturale.

1.2.1 Physical Standby

Nella *Figura (2)* a seguire viene proposta una rappresentazione dell'architettura Data Guard di una replica tramite Physical Standby. In questa soluzione il pro-

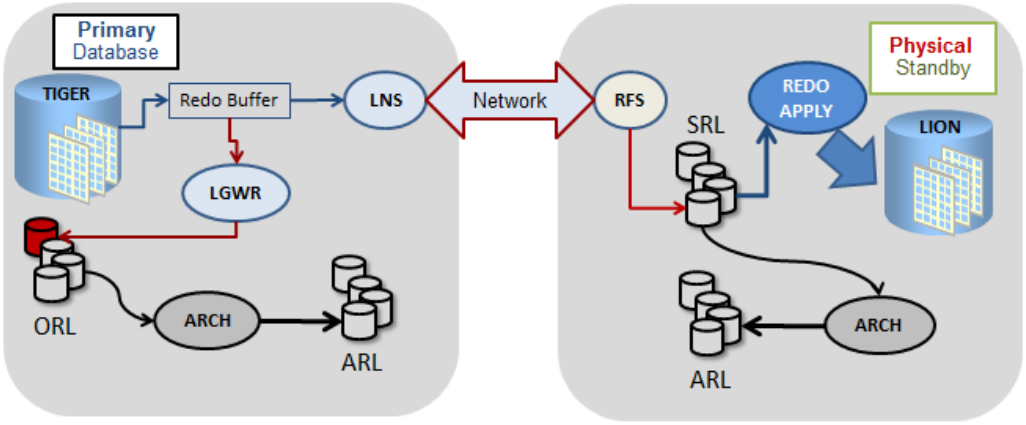

Figura 2: Componenti del Primary e del Physical Standby

cesso di allineamento e replica dei dati prende il nome di ***"Redo Apply"***, ovvero applicazione sull' istanza Standby dei Redo Log del Primary. Questa tecnica garantisce che lo Standby sia una *esatta copia fisica* del Primary. I Redo Log infatti contengono una serie di informazioni a basso livello che indicano quali blocchi dei

datafile vengono manipolati e in che modo. Con la conoscenza di queste informazioni il processo di Redo Apply sullo Standby è in grado di riproporre sui datafile locali le stesse identiche modifiche Avvenute sul Primary. Per questo motivo il tipo di Standby prende il nome di Physical Standby. Questo meccanismo risulta inoltre molto performante dal punto di vista di rapidità di applicazione delle modifiche sullo standby ed anche dal punto di vista della robustezza della protezione dei dati in ottica di DR. Le modalità di funzionamento di un Physical Standby sono ampiamente configurabili e dipendono dalle caratteristiche di Protection Mode scelte. Inoltre questa soluzione supera un problema intrinseco dei meccanismi di mirroring remoto delle SAN (Storage Area Network). Queste infatti, non potendo utilizzare i meccanismi nativi di Oracle per validare lo scambio di pacchetti, non sono in grado di certificare i blocchi dati inviati all'host remoto, con la conseguenza che il db replicato potrebbe anche risultare inconsistente. Data Guard nella configurazione di Standby Fisico è in grado di validare ogni singolo pacchetto ed anche parallelizzare la riapplicazione dei redolog ricostruendo comunque la corretta sequenza dei dati grazie alla scrittura degli SCN *(System Change Number)* all'interno dei redolog.

1.2.2 Logical Standby

Nella *Figura (3)* a seguire viene proposta una rappresentazione dei componenti su un Logical Standby, tralasciando i componenti del Primary che restano invariati rispetto al caso precedente.

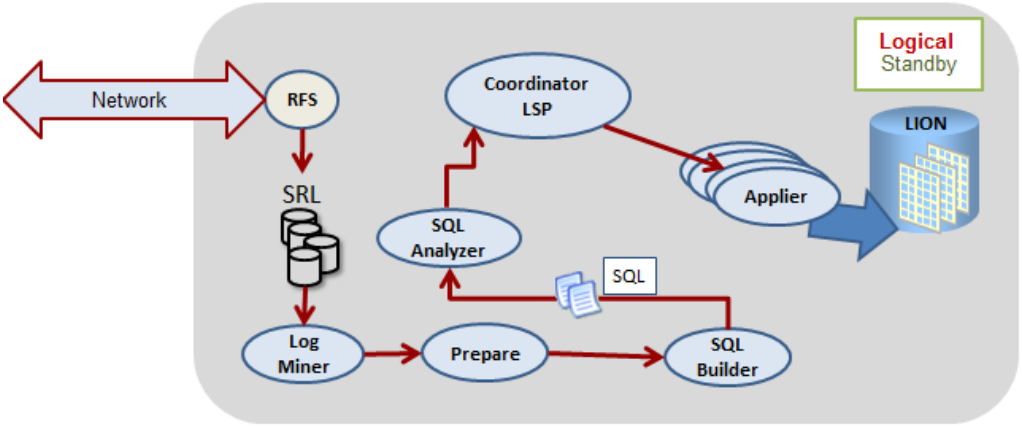

Figura 3: Componenti di un Logical Standby

Il processo di replica dei dati nello Standby Logico coinvolge diverse entità e sottoprocessi e prende complessivamente il nome di ***"SQL Apply"***. Innanzi tutto uno standby Logico permette la replica delle informazioni del Primary svincolando lo Standby dal mantenere la stessa identica struttura fisica del Primary (datafile, tablespace o schema). Questo è possibile perchè viene garantita la replicazione delle informazioni logiche a mezzo della riapplicazione delle stesse operazioni logiche avvenute nel Primary.

Il meccanismo sfrutta la capacità della Log Miner [1] di ricostruire i comandi SQL che hanno originariamente prodotto le entry negli ORL. Quello che viene rappresentato nello schema a eguire indica i vari step concettuali che avvengono sullo Standby a valle dell' analisi della Log miner sugli SRL. Vengono tralasciati gli step riguardanti il Primary in quanto del tutto identici allo schema presentato precedentemente. Come si può facilmente immagianre, l'overhead introdotto dalla Log Miner e dal processo di applicazione degli statement SQL, si traduce in un processo di allineamento dello Standby computazionalmente molto più oneroso, e più lento, rispetto alla tecnica di Redo Apply

A proposito di uno Standby Logico bisogna inoltre precisare che ci sono una serie di limitaizoni sulla tipologia dei dati del Primary che possono essere replicati su Standby di questo tipo. Per contro è invece possibile aggregare su uno

stesso Standby Logico i dati da diverse fonti Primary, cosa che evidentemente uno Standby Fisico non consentirebbe. E quindi importante valutare attentamente quale tipologia di Standby sia la più adatta alle esigenze in essere tenendo presente che è sempre possibile collegare ad un Primary più di uno Standby, anche di tipi differenti costruendo così architetture ibride multi-standby.

1.3 Politiche di Protezione Dati

Nel configurare una soluzione di DR/HA con Data Guard la decisione più importante, dopo la scelta del tipo di Standby, è la politica di protezione dati (*Data Protection Mode*) da implementare. Per meglio chiarire il grado di importanza che questa scelta riveste all'interno di una architettura Data Guard, basta considerare il fatto che da questa dipende la possibilità di avere una replica con o senza perdita di dati. Anche un certo degrado delle performance del Primary o la sua capacità di continuare a lavorare a fronte di guasti allo Standby, perdita di connettività o altro, dipendono dalla Protection Mode implementata. Vengono quindi presentate le tre modalità di protezione dati che Data Guard mette a disposizione.

1.3.1 Maximum Performance Mode

Questa modalità di lavoro predilige sicuramente le performance del Primary rispetto alla protezione dei dati. La trasmissione, dal Primary allo Standby, avviene in modalità *asincrona (ASYNC)*, ovvero il processo LGWR [5] non aspetta nessuna conferma (ACK) da parte dell' RFS sullo standby prima di scrivere i record di redolog sul current redo log. Questo si traduce anche nell' assenza di degrado di prestazioni sul Primary a fronte della possibilità che lo Stnadby possa perdere dei pacchetti di redolog senza che il Primary se ne accorga e possa tentare di rispedire i dati. Per ovviare al problema della perdita di pacchetti di redolog è sempre possibile sincronizzare gli ARL, comunque presenti nel Primary, su una stage area nello Standby e renderli disponibili per operazioni di riallineamento manuale al posto degli SRL persi.

[5] Nella versione 11g di Data Guard la modalità di trasporto dati tramite il processo *ARCH* è deprecata.

1.3.2 Maximum Availability Mode

Questa modalità mette al primo posto per importanza la disponibilità di servizio del Primary tenendo però al secondo posto la protezione dei dati. Detto in altri termini allo strato di trasporto è richiesta una modalità *sincrona (SYNC)*, con un conseguente degrado di prestazioni del Primary dovuto ai tempi di attesa sulla ricezione da parte dell' LGWR degli ACK in arrivo dallo Standby. Questa modalità di trasmissione dati dal Primary allo Standby garantisce l'assenza di perdita di dati, ma allo stesso tempo è suscettibile a fattori che non impattano direttamente sulla disponibilità di servizio del Primary. Ad esempio una loop di rete che generi un ritardo eccessivo nella spedizione o una completa perdita di connettività tra Primary e Standby si traduce in un tempo di attesa degli ACK. Questa modalità prevede la configurazione di un timeout, (*NET_TIMEOUT*), oltre il quale il processo LGWR si disconnette dall LFS dello STandby, continuando ad accettare transazioni dagli utentei e registrandole sui ORL. In questo modo si passa a una comunicazione asincrona con lo Standbay. Data Guard in questa configurazione è comunque in grado di rilevare quando lo Standby torna raggiungibile e riallineare tutte le spedizioni perse. Questo riallineamento viene fatto a partire dagli ARL ad opera del processo ARCH che continua monitorare la connettività verso il processo LFS dello Standby. Ovviamente se un evento potenzialmente disastroso dovesse colpire il Primary durante le operazioni di re-sincronizzazione degli ARL e degli ORL questo si tradurrebbe inevitabilmente in una perdita di dati.

1.3.3 Maximum Protection Mode

Questa modalità, come dice il nome, assegna alla protezione dei dati la maassima priorità, a scapito di prestazioni o disponibilità di servizio. Richiede dunque una comunicazione di tipo *sincrono (SYNC)* nel substrato di trasporto con un ack esplicito da parte dello Standby verso il Primary a fronte di ogni invio/ricezione di SRL. Il processo LGWR sul Primary attende indefinitamente di ricevere *almeno un ACK* dallo Standby e non accetta ulteriori transazioni utente se non ha la certezza che l' ORL spedito non sia stato ricevuto dallo Standby. Dal punto di vista dell' impatto computazionale sul Primary ha dunque lo stesso peso della "Max Availability" con l'eccezione che non viene considerato nessun parametro di timeout sulle trasmissioni degli ORL. Cosa molto importante da tenere presente nella scelta di questa modalità è il fatto che se il Primary non riceve ack dallo Standby va in stallo e può addirittura abortire l' istanza per prevenire commit su transazioni non trasferite sullo Standby. Questo comportamento garantisce l'assenza di perdita di dati anche a fronte di più situazioni di fallimento in sequenza, ad esempio una prima perdita di rete e una successiva rottura sui dischi del primary. Un trucco per evitare che il Primary vada in stallo in caso di mancata ricezione di ack è quello di configurare almeno 2 distinti Stnadby su due siti differenti in modo da garantire una maggiore probabilità che *almeno uno Standby* sia sempre in grado di ritornare un ack al Primary.

1.4 Recovery Time Objective e Recovery Point Objective

I due parametri che svengono presentati qui di seguito da un alto sono considerabili come indici di prestazione per soluzioni di Disaster Recovery e High Availability, dall'altro devono essere considerati come dei requisiti di progetto ed essere costantemente verificati durante il ciclo di vita del sistema. Tanto più piccoli sono questi due indici quindi, tanto migliore risulta essere la soluzione di DR/HA adottata. In alcuni casi è anche possibile implementare architetture che consentano di tenere entrambi a valori molto prossimi a zero; in questi casi si parla di architetture Business Continuity Compliant.

Recovery Time Objective (RTO)

In una analisi architetturale di Disaster Recovery e Business Continuity l'RTO [6] è un indice che rappresenta il tempo necessario per il pieno recupero dell'operatività di un sistema o di un processo organizzativo. In pratica è il massimo tempo di down tollerabile dei servizi. Questo semplice parametro è un aspetto di primaria importanza che deve essere conosciuto e valutato attentamente in fase progettuale. Va inoltre verificato periodicamente sul campo con sessioni in cui vengono simulati i guasti e riapplicate le procedure di ripristino. Non conoscere l'effettiva estensione di questo parametro spesso comporta ingenti perdite e danni per enti e aziende.

Recovery Point Objective (RPO)

Viceversa l'indice RPO [6] indica il tempo massimo che intercorre tra l'istante in cui viene prodotto un dato e quello in cui viene messo in sicurezza, ad esempio tramite la sua replica su un sito remoto o un backup. Anche questo parametro è fondamentale, perchè individua la finestra massima di tolleranza ai guasti di un sistema informatico. In altre parole indica la massima quantità di dati che il sistema potrebbe perdere a fronte di un evento di disastro improvviso.

1.5 Le Transizioni di Ruolo

Una operazione di cambio, o transizione, di ruolo è una delle operazioni fondamentali che possono essere attuate in una configurazione Data Guard. Esistono due differenti tipologie di cambio di ruolo e vengono brevemente introdotte in questa sezione. Per maggiori dettagli si rimanda alla documetazione ufficiale [6]

Switchover: Questa operazione indica una completa inversione dei ruoli tra il Primary e lo Standby database; il Primary diventa Standby e lo Standby diventa Primary. Una caratteristica peculiare di questo tipo di transizione di ruolo è la completa reversibilità, in particolare è sempre possibile tornare alla configurazione originaria applicando una seconda operazione di switchover. Per questa sua caratteristica viene spesso utilizzata qualora il sistema Primary debba essere messo sotto manutenzione; una volta terminate le operazioni di manutenzione può esere pianificato un secondo switchover per il ripristino dei ruoli originari.

Failover: poichè questa operazione non è reversibile, tipicamente si sceglie di attuare una operazione di failover a fronte di gravi problemi sulla sede primaria. E' una operazione che consente di avviare lo Standby in modalità Primary e rendere comunque fruibile il servizio. La non reversibilità è dovuta al fatto che il vecchio Primary viene completamente escluso dalla configurazione Data Guard e non è possibile ricollegarlo con una semplice riconfigurazione. Riportando il tutto all'esempio pratico della presente guida nell'eventualità di un grave problema su TIGER l'operaizone di Failover vedrebbe TIGER escluso dalla configurazione e LION riconfigurato come Primary. A questo punto su domain02.infn.it ci sarebbe l'istanza di produzione. Sarebbe comunque possibile ricreare da capo TIGER come Standby di LION in modo da poter poi effettuare una nuova operazione di switchover e ripristinare la produzione sul dominio originario.

2 Primo Test INFN : la Configurazione Hardware e Software

Viene ora presentata in forma tabellare la configurazione sistemistica degli ambienti *Primary* e *Standby*. A seguire, come già accennato nell' introduzione, tutti i dati sensibili vengono sostituiti con nomi fittizi, da un lato per er motivi di sicurezza informatica, dall' altro per mantenere il carattere generale della nota tecnica. Valgono quindi le seguenti considerazioni:

- I nomi delle macchine e dei domioni di rete INFN vengono sostituiti rispettvamente con *host0X* e *domain0X.infn.it* ove la X vale 1 per l'ambiente Primary e 2 per l'ambiente Standby.
- Qualora necessario indicare numeri di porte logiche negli esempi, vengono considerati quelli "standard" reperibili nella documentazione di riferimento di Oracle [6]. Tipicamente sono 1521 e 2484.
- Su tutte le macchine viene data per scontata l'installazione del software Oracle (ORACLE_HOME) alla versione 11gR2 tramite un utente di Sistema Operativo dedicato ("oracle").
- Ove necessario fare esplicito riferimetno alla password dell'utente "SYS" (o "SYSTEM") questa verrà indicata come **syspasswd** per semplicità [6].

Riportiamo in breve la configurazione dell'istanza Primary (*TIGER*) ospitato su host *host01* e a seguire la configurazione dell' istnaza Standby (*LION*) su host *host02*. Considerando una configurazione di rete che preveda una replica geografica dei dati, i domini di appartenenza delle macchine verranno mappati con *domain01.infn.it* per il Primary database e *domain02.infn.it* per l' istanza Standby.

Daremo inoltre per scontata la configurazione del Sistema operativo e tutti i passi di prerequisito necessari all'installazionee della ORACLE_HOME come da [3] e [5].

[6] E' sempre buona norma seguire le linee guida per la sicurezza [2] pubblicate da Oracle in merito al trattamento delle passwd degli utenti standard e di quelli privilegiati come SYS o SYSTEM.

Configuraizione dell'ambiente Primary:

Oracle Release	**Oracle 11g Release 2 (11.2.0)**
Host Name	**host01.domain01.infn.it**
Operating System	**Red Hat Linux 6.3 a 64bit** (o CentOS equivalente)
Database Name: (db_name)	**TIGER**
Database Domain: (db_domain)	**domain01.infn.it**
ORACLE_SID:	**TIGER**
ORACLE_HOME: (db_name)	**/oracle/db/product/11.2.0/**
Database Unique Name: (db_unique_name)	**TIGER**
TNS Alias:	**TIGER.domain01.infn.it**
Service Names:	**TIGER.domain01.infn.it, TIGER**
Database Files: (db_create_file_dest)	**/oracle/db/oradata/**
Flash Recovery Area: (db_recovery_file_dest)	**/oracle/db/flash_recovery_area/**
Local Online Redo Log: (log_archive_dest_1)	**location=use_db_recovery_file_dest (all_logfiles,all_roles)**
Remote Arc. Dest. (log_archive_dest_2)	**service=LION (online_logfiles, primary_role)**

Tabella 1: Configurazione dell' ambiente Primary

Configuraizione dell'ambiente Standby :

Oracle Release	**Oracle 11g Release 2 - (11.2.0)**
Host Name	**host02.domain02.infn.it**
Operating System	**Red Hat Linux 6.3 a 64bit** (o CentOS equivalente)
Database Name: (db_name)	**TIGER**
Database Domain: (db_domain)	**domain02.infn.it**
ORACLE_SID:	**LION**
ORACLE_HOME: (db_name)	**/oracle/db/product/11.2.0/**
Database Unique Name: (db_unique_name)	**LION**
TNS Alias:	**LION.domain02.infn.it**
Service Names:	**LION.domain02.infn.it, LION**
Database Files: (db_create_file_dest)	**/oracle/db/dataguarddata/**
Flash Recovery Area: (db_recovery_file_dest)	**/oracle/db/flash_recovery_area/**
Local Online Redo Log: (log_archive_dest_1)	**location=use_db_recovery_file_dest (all_logfiles,all_roles)**
Remote Arc. Dest. (log_archive_dest_2)	**service=TIGER (online_logfiles, primary_role)**

Tabella 2: Configurazione dell'ambiente Standby

Un primo elemento da notare rispetto alle due configurazioni è che il parametro *Remote Archive destination* sia settato nel Primary con il riferimento allo Standby, e viceversa. Questo consente ad entrambe le istanze di conoscere la destinazione remota a cui spedire gli ARL (soluzione deprecata) e gli ORL che verranno salvati come SRL. Da notare inoltre che il parametro log_archive_dest_2 sullo Standby può essere configurato sia per puntare al Primary, sia per puntare ad un ulteriore istanza di Standby database. Nel primo caso si stà configurando una soluzione che consente una completa inversione dei ruoli tra Primary e Standby, nel secondo caso si stà ponendo uno Standby database a valle di LION, allungando di un passaggio la catena di replicazione delle istanze.

Il secondo elemento importante è che in entrambe le configurazioni il valore del parametro *Database Name* risulta essere il medesimo (TIGER), mentre le due istanze Primary e Standby differiscono per il valore di *Database Unique Name*.

Questo è necessario perchè le istanze sarano a tutti gli effetti due diverse istanze, che però devono rappresentare lo stesso database al fine della riapplicazione sullo Standby dei pacchetti di redolog in arrivo dal Primary.

Infine si è scelto per semplicità di utilizzare le stesse versioni del software Oracle per entrambe le istanze (11gR2) sebbene con la nuova versione di Data Guard non sia più una scelta obbligata. Se da un lato infatti nella release 10g era necessario avere la stessa medesima architettura [stesso OS, stessa famiglia di processori (32bit o 64 bit), stessa versione e livello di patching della Oracle Home, stesso character set sulla macchina] nella versione 11g questo non è richiesto a meno di considerazioni minori sulle famiglie dei processori (little-endian ws big-endian).

Diventa dunque possibile utilizzare un Physical Stanby come step intermedio di una migrazione oppure nella configurazione di ambienti ibridi. Si rimanda alla [4] per un dettaglio più completo della possibilità introdotte con Data Guard 11g.

3 La Configurazione dei Componenti di Rete

Oracle Data Guard dipende dallo strato di rete per la trasmissione dei pacchetti dati (ORL) dal Primary allo Standby[6]. Vediamo quindi come configurare i vari componenti di rete (Oracle Net) in modo appropriato per garantire la connettività tra i due database Primary e Standby. In particolare verranno configurati tre componenti; il Listener, la Risoluizone degli Alias sul TNS e le Policy di connessione dello strato SQLNet. Configurare questi componenti significa configurare, in maniera assistita o a mano, rispettivamente i seguenti tre file *listener.ora*, *tnsnames.ora* e *sqlnet.ora*

3.1 Configurazione di Oracle Listener *(listener.ora)*

Per permettere ai processi Listener delle due istanze di referenziarsi reciprocamente in maniera statica si consiglia di inserire le seguenti entry :

Configurazione sul Primary

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/db/product/11.0.2/)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = TIGER.domain01.infn.it)
      (SID_NAME = TIGER)
      (ORACLE_HOME = /oralce/db/product/11.0.2/)
    )
)
INBOUND_CONNECT_TIMEOUT_LISTENER = 0
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = host01.domain01.infn.it)(PORT = 1521))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
 )
```

Configurazione sullo Standby

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/db/product/11.0.2/)
      (PROGRAM = extproc)
```

```
    )
    (SID_DESC =
      (GLOBAL_DBNAME = LION.domain02.infn.it)
      (SID_NAME = LION)
      (ORACLE_HOME = /oralce/db/product/11.0.2/)
    )
)
INBOUND_CONNECT_TIMEOUT_LISTENER = 0
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = host02.domain02.infn.it)(PORT = 1521))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
 )
```

Dopo aver effettuato le modifiche riavviare i processi Listener su entrambi i server. E' comunque anche possibile effettuare una semplice operazione di "reload" dei servizi in modo da aggioranre la configurazione del listener senza perdere le connessioni già in essere.

```
[oracle@host01 ~]$ lsnrctl reload TIGER
[oracle@host02 ~]$ lsnrctl reload LION
```

3.2 Risoluzione degli Alias sul TNS *(tnsnames.ora)*

Entrambi i database Primary e Standby devono avere nella configurazione del rispettivo *tnsnames.ora* il riferimento all'altro server in modo tale che il listener riesca a risolvere la risorsa remota.

Ad esempio sul Primary inseriamo un'alias per lo Standby in questo modo :

```
LION.domain02.infn.it =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = host02.domain02.infn.it)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = LION.domain02.infn.it)
    )
  )
```

Contemporaneamente sullo Standby inseriamo le seguenti entry che puntano al Primary:

```
TIGER.domain01.infn.it =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = host01.domain01.infn.it)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = TIGER.domain01.infn.it)
    )
  )
```

Da notare che in entrambi i casi è stato inserito esplicitamente il dominio nel parametro SERVICE_NAME; questo perchè l'ipotesi fondamentale è che le due istanze Primary e Standby siano "up & running" su server differenti sotto due differenti domini.

3.3 Le Policy SQLNet *(sqlnet.ora)*

Le più semplici impostazioni per lo strato di connessione remota tramite SQLNet sono rappresentate dalle seguenti entry sul file sqlnet.ora:

```
NAMES.DIRECTORY_PATH=(TNSNAMES, ONAMES, HOSTNAME)
SQLNET.EXPIRE_TIME= 10
```

La prima riga indica l'ordine di priorità tra le varie modalità di risoluzione dei nomi, mentre la seconda indica un semplice timeout sui tentativi di connessione. Se nel file inoltre venisse impostato il valore per il parametro NAMES.DEFAULT_DOMAIN questo verrebbe automaticamente utilizzato per tutti i SERVICE_NAME presenti nel tnsnames.ora che non fanno riferimetno ad uno specifico domain name. Nel caso di una replica geografica su due distinti domini è sconsigliato quindi utilizare questo parametro.

3.4 Verifica delle Impostazioni di Rete

A questo punto verifichiamo i servizi listener e la possibilità di raggiungere le macchine da entrambi i server; di seguito il tentativo di raggiungere lo Standby (LION) dall' istanza Primary (TIGER).

Innanzi tutto verifico lo stato del Listener sull'istanza LION

```
[oracle@host02 ~]$ lsnrctl services — grep '^Service'

Services Summary...
Service "LION.domain02.infn.it" has 2 instance(s).
Service "PLSExtProc" has 1 instance(s).
Service "LIONXDB.domin02.infn.it" has 1 instance(s).
Service "LION_XPT.domain02.infn.it" has 1 instance(s).
```

Quindi verifico la connettività di rete dal Primary verso lo Standby tramite il comando *tnsping*. Ci sono due modi per farlo; sfruttando una descrizione esplicita (non viene utilizzato il tnsnames.ora per la risoluzione dell'Alias) oppure tramite un Alias TNS. Le vediamo entrambe in sequenza.

```
[oracle@host01 ~]$ tnsping host02.domain02.infn.it:1521/LION  (ping diretto)
[oracle@host01 ~]$ tnsping LION.domain02.infn.it              (ping tramite Alias TNS)

TNS Ping Utility for Linux: Version 11.2.0 on 15-SEPT-2013 14:46:28

Copyright (c) 1997 Oracle Corporation.  All rights reserved.

Used parameter files:
Used TNSNAMES adapter to resolve the alias

Attempting to contact
(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host02.domain02.infn.it)
(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=LION.domain02.infn.it)))
OK (10 msec)
```

Dualmente faremo le stesse verifiche di connettività dallo Standby verso il Primary. L'ultimo punto da verificare ora è la possibilità di effettuare una connessione da un database all'altro tramite sqlplus e RMAN. Per consentire questo tipo di connessioni è necessario che ntrambi i database siano configurati per utilizzare il *password file* ed in particolare che gli utenti condividano la stessa passwd. In particolare, ai fini della creazione dello standby e della corretta riapplicazione su di esso degli SRL, è necessario che i due sistemi condividano in questo modo la passwd dell' utente privilegiato ***SYS***.

Per fare questo il modo più sicuro è creare un passwd file sul Primary, assegnare la passwd di SYS e successivamente trasmetterlo allo Standby tramite una connessione sicura e cifrata . . . tipicamente tramite un comenado *scp* dal Primary allo Standby.

Vediamo come creare il passwd file e spedirlo, correttamente rinominato, allo Standby:

```
[oracle@host01 ~]$   cd $ORACLE_HOME/dbs
[oracle@host01 dbs]$   orapwd file=orapwtiger password=*syspasswd*
[oracle@host01 dbs]$   scp -p orapwtiger host02.domain02.infn.it:/oracle/db/product/11.2.0/dbs/orapwlion
```

Per testare la possibilià di collegarsi con diritti di SYS sulla macchina remota si può procedere sia tramite sqlplus che tramite RMAN. In seguito sarà necessario utilizzare RMAN per le operazioni di BKUP e copia dei dati, quindi è consigliabile testare subito la connettiità tramite questo tool.

Dal Primary allo Standby

```
[oracle@host01 ~]$   rman target sys/*syspasswd*@LION.domain02.infn.it
Recovery Manager: Release 11.2.0.7.0 - Production on Mon Dec 17 15:00:58 2012
Copyright (c) 1982, 2007, Oracle.  All rights reserved.

connected to target database: TIGER (DBID=********)
```

. . . ed anche dallo Standby al Primary

```
[oracle@host02 ~]$   rman target sys/*syspasswd*@TIGER.domain01.infn.it

Recovery Manager: Release 11.2.0.7.0 - Production on Mon Dec 17 15:00:58 2012
Copyright (c) 1982, 2007, Oracle.  All rights reserved.

connected to target database: TIGER (DBID==========)
```

Infine resta da settare ad “EXCLUSIVE” il parametro REMOTE_LOGIN_PASSWORDFILE nell’ spfile (ed anche nel pfile) del Primary.

```
[oracle@host01 ~]$   sqlplus /nolog
SQL>  conn sys as sysdba
SQL>   alter system set remote_login_passwordfile=exclusive scope=both;
System altered.

SQL>   shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.

SQL>   startup open
ORACLE instance started.

Total System Global Area 1241513984 bytes
Fixed Size                  1273420 bytes
Variable Size             318767540 bytes
Database Buffers          905969664 bytes
Redo Buffers               15503360 bytes
Database mounted.
Database opened.
```

A questo punto la configurazione di rete per i due database è completata. Si rimanda alle Appendici [8.1 a pag 39] per la variante della configurazione dei componenti di rete che consente la cifratura dei dati inviati e ricevuti a mezzo di protocollo TCPS su tunnel SSL.

4 La Configurazione del Primary

Normalmente una installazione base di un database Oracle Enterprise non ha riferimenti al ruolo (Primary o Standby) del database e il wizard dell'installazione non prevede la configurazoine di un'architettura con replica remota. Questa va creata configurando a mano le istanze coinvolte oppure agendo tramite le funzionalità avanzate dell'Oracle Enterprise Grid Console.

Di conseguenza quando si vuole creare uno Standby Database, nel caso della presente guida un Physical Standbay, bisogna innanzi tutto istruire il database sorgente in modo tale che l'istanza "sappia" di essere membro di una coppia *Primary-Standby*. Per fare questo si deve agire manualmente sulla configurazione dei una serie di parametri di inizializzaizone del database, assicurarsi che il database sorgente sia in *Archive Mode* ed effettuare una serie di passaggi di riconfigurazione.

Se si vuole inoltre la possibilità di applicare un context switch, invertendo i ruoli di Primary e Standby, si deve inoltre tenere conto del fatto che in un secondo tempo il Primary diventerà uno Standby e vice versa. Diventa quindi importante che entrambe le istanze contengano sia i parametri da utilizzare sotto il ruolo di Primary e quelle da utilizzare sotto il ruolo di Standby. Ovviamente partiamo con TIGER in configurazione Primary e LION in modalità Standby.

In questa sezione ci occupiamo delle configurazioni necessarie sul Primary TIGER, solo successivamente [5 a pag.32]configuriamo lo Standby LION.

Anche se non tutti gli step presentati in questa sezione e nella prossima sono strettamente necessari, sono comunque consigliati per ottenere una configurazione di replica in grado di effettuare operazioni di *switchover* tra le due istanze.

4.1 Abilitare l' Archive Mode

Il primo passo, come già accennato, è porre TIGER in Archive Mode, ed assicurarsi che al restart il processo di archiver venga correttamente avviato. Questa operazione è necessaria per consentire l' invio dei pacchetti di redolog allo standby e consentire il costante allineamento dei dati su di esso. Attuiamo quindi la seguente sequenza di operazioni come sysdba:

```
SQL> shutdown immediate
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
```

Per una verifica immediata dell'attivazione della modalità di archive basta eseguire il comando riportato in seguito e verificare che il parametro "Automatic Archival" risulti "Enabled" :

```
SQL>  achive log list

Database log mode              Archive Mode
Automatic archival             Enabled

Archive destination            USE_DB_RECOVERY_FILE_DEST
Oldest online log sequence     34
Next log sequence to archive   36
Current log sequence           36
```

Nota:
1) Ovviamente il parametro USE_DB_RECOVERY_FILE_DEST *deve essere correttamente definito e identificare la Flash Recovery Area del database.*
2) Allo stesso modo il parametro DB_RECOVERY_FILE_DEST_SIZE indicherà la quota disco assegnata.
3) Dalla versione 10g in poi il database attiva automaticamente l'archiviazione una volta posto il database in archive mode, mentre nelle versioni precedenti era necessario settare a TRUE anche il valore del parametro LOG_ARCHIVE_START .

4.2 Forzare le scritture sui Redo Log (opzionale)

Poichè il meccanismo di replica remota dei dati, dal Primary verso lo Standby, si basa essenzialmente sulla ri-applicazione lato Standby delle operazioni registrate sugli ORL del Pirmary, ne consegue che ogni operazione che non venga scritta nei segmenti di redolog, risulterebbe inevitabilmente **persa** per lo Standby rendendo *inconsistente* la replica.

Una soluzione semplice per prevenire questo genere di problemi è obbligare il Primary a scrivere negli ORL ogni singola operazione effettuata sui dati. Questa tecnica prende il nome di *"force logging mode"* ed implica che anche le operazioni che normalmente non genererebbero scritture sui segmenti di redolog generino delle entry. Anche se in molti manuali è definito come step opzionale, in realtà sottovalutarlo significa dover affrontare, ad esempio nel caso della gestione di un failover, un overhead considervole di operazioni.

Per imporre il logging basta eseguire come *sysdba* la seguente alter :

```
SQL>  alter database force logging;
system altered.
```

Mentre per la verifica del parametro di logging basta interrogare opportunamente la vista di sistema ***v$database***.

```
SQL>  select force_logging from v$database;
FORCE_LOGGING
-------------
YES
```

4.3 Creazione degli Standby Redo Log (opzionale)

Quando si configurano gli SRL per una istanza è raccomandato utilizzare un gruppo di Standby redo log in più rispetto ai gruppi di ORL configurati per l'istanza. Inoltre, cosa molto importante, ogni SRL file deve essere della stessa dimensione del corrispondente ORL file, dovrà contenerne infatti tutte le entry. La seguente formula [6] permette di calcolare il corretto numero di SRL per l'istanza Primary [7]:

Numero SRL = (# of online redo log file groups on primary + 1) * maximum # of threads

In questa nota tecnica viene considerata la configurazione di un database Single Instance (non RAC [8]) con 3 gruppi di ORL composti ogniuno da 2 file di dimensioni pari a 50MB l' uno.

Di seguito riportiamo i dettagli dei dati di un database standard, ad esempio installato con il classico wizard, al fine di esplicitare il meccanismo di computo della formula presentata.

```
SQL>  select group#, thread#, bytes, members from v$log;
    GROUP#    THREAD#      BYTES    MEMBERS
---------- ---------- ---------- ----------
         1          1   52428800          2
         2          1   52428800          2
         3          1   52428800          2
```

Se i dati di partenza sono quelli indicati qui sopra, allora la formula riporterà il seguente conteggio :

```
Numero  di  SRL = (3 + 1) * 1 = 4 da 50MB l'uno.
```

[7] il simbolo di # indica quanti gruppi di redolog sono configurati per un file di redolog.

[8] *Nel caso di una installazione Primary di tipo RAC, diversamente da quanto presentato in questa guida, la formula sopra elencata deve considerare il numero di gruppi di redolog di tutte le istanze del database clusterizzato, quindi se l'installazione prevede due istanze cluster del database, ogniuna con 3 gruppi di online redo log, dovrò conteggiare 6 gruppi totali + 1 gruppo in più per ogni istanza RAC.*

Per creare gli SRL necessari basterà collegarsi come utente 'SYS' del database ed eseguire le seguenti alter aggiungendo i file e gruppi. In particolare, visto che gli ORL occupano i gruppi 1,2,e 3 , quelli che verranno creati per gli SRL saranno i gruppi 4,5, 6 e 7 (un gruppo in più rispetto agli ORL)

```
SQL> alter database add standby logfile thread 1 group 4 size 50m;
SQL> alter database add standby logfile thread 1 group 5 size 50m;
SQL> alter database add standby logfile thread 1 group 6 size 50m;
SQL> alter database add standby logfile thread 1 group 7 size 50m;
```

Ora verifichiamo il risultato :

```
SQL> select group#, type, member from v$logfile order by group#, member;
    GROUP# TYPE    MEMBER
---------- ------- ------------------------------------------------------------
         1 ONLINE  /oracle/db/oradata/TIGER/onlinelog/o1_mf_1_6jjc8hr8_.log
         1 ONLINE  /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_1_6jjc8jxq_.log
         2 ONLINE  /oracle/db/TIGER/onlinelog/o1_mf_2_6jjc8mvj_.log
         2 ONLINE  /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_2_6jjc8nvv_.log
         3 ONLINE  /oracle/db/oradata/TIGER/onlinelog/o1_mf_3_6jjc8qrw_.log
         3 ONLINE  /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_3_6jjc8ry2_.log
         4 STANDBY /oracle/db/oradata/TIGER/onlinelog/o1_mf_4_6hvg3qk9_.log
         4 STANDBY /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_4_6hvg3rnm_.log
         5 STANDBY /oracle/db/oradata/TIGER/onlinelog/o1_mf_5_6hvg48f3_.log
         5 STANDBY /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_5_6hvg49gr_.log
         6 STANDBY /oracle/db/oradata/TIGER/onlinelog/o1_mf_6_6hvg4m9j_.log
         6 STANDBY /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_6_6hvg4nb4_.log
         7 STANDBY /oracle/db/oradata/TIGER/onlinelog/o1_mf_7_6hvg4w3d_.log
         7 STANDBY /oracle/db/flash_recovery_area/TIGER/onlinelog/o1_mf_7_6hvg4xbl_.log
```

4.4 Configurazione Parametri Inizializzazione

A questo punto dobbiamo configurare i parametri di inizializzazione del database Primary per indicare alla sua istanza TIGER che ha uno Standby collegato a valle.

La maggior parte delle impostazioni di Data Guard vengono definite passando attraverso opportune configurazioni dei parametri di inizializzazione del database. Di seguito i settaggi da introdurre nel parameter file del Primary al fine di rendergli nota l'esistenza dello standby e di configurare le modalità di lavoro (Max Protection, Max Availability o Max Performance).

Anche se non è strettaemnte necessario che le macchine fisiche del Primary e dello Standby siano identiche questo consentirebbe di utilizzare lo stesso parameter file del Primary, con opportune modifiche, anche sullo Standby, lasciando inalterati i parametri di memoria, kernel ecc...

Inoltre è importante considerare sia i parametri necessari a TIGER nel ruolo di Primary, che quelli necessari nel ruolo di Standby, qualora si voglia una architettura che consenta l'inversione di ruoli tra le istanze. Questi parametri verranno riportati di seguito divisi per aree di influenza.

```
# ---[ Dump Destination Parameters ] --- #
audit_file_dest='/oracle/db/product/11.2.0/admin/TIGER/adump'
background_dump_dest='/oracle/db/product/11.2.0/admin/TIGER/bdump'
core_dump_dest='/oracle/db/product/11.2.0/admin/TIGER/cdump'
user_dump_dest='/oracle/db/product/11.2.0/admin/TIGER/udump'

# ---[ Role-independent Parameters ] --- #
archive_lag_target=900
compatible='11.2.0'
control_files='/oracle/db/oradata/TIGER/controlfile/o1_mf_6hc6stn9_.ctl',
              '/oracle/db/flash_recovery_area/TIGER/controlfile/o1_mf_6hc6styy_.ctl'
db_name='TIGER'
db_domain='domain01.infn.it'
db_create_file_dest='/oracle/db/oradata'
db_recovery_file_dest='/oracle/db/flash_recovery_area'
dispatchers='(PROTOCOL=TCP) (SERVICE=TIGERXDB)'
instance_name='TIGER'
log_archive_config='dg_config=(TIGER,LION)'
log_archive_max_processes=4
remote_login_passwordfile='exclusive'

# ---[ Primary Role Parameters ] --- #
db_unique_name='TIGER'
log_archive_dest_1=
'location=use_db_recovery_file_dest valid_for=(all_logfiles,all_roles) db_unique_name=TIGER'
log_archive_dest_2=
'service=LION.domain02.infn.it valid_for=(online_logfiles,primary_role) db_unique_name=LION'
log_archive_dest_state_1='enable'
log_archive_dest_state_2='defer'
service_names='TIGER.domain01.infn.it, TIGER'

# ---[ Standby Role Parameters ] --- #
db_file_name_convert='/LION/','/TIGER/'
log_file_name_convert='/LION/','/TIGER/'
fal_server='TIGER','LION'
fal_client='TIGER'
standby_file_management='auto'
```

Poichè alcune di queste modifiche ai parametri di inizializzazione non sono applicabili a runtime, si deve procedere al riavvio del database.

> **Nota:**
> *Nel prospetto appena visto il parametro* ***log_archive_dest_state_2*** *viene lasciato impostato al valore 'deferred' fino a quando non verrà creato e messo in mount il database Standby. Solo allora verrà cambiato sul Primary con una opportuna 'alter database set...' che indicherà al database l'inizio delle attività di spedizione*

degli ORL verso lo Standby.
Inoltre tra Primary e Standby i valori di conversione assegnati al prametro ***log_file_name_convert*** *sono diversi.*

4.5 Creare un Backup del Primary

A questo punto prepariamo un backup completo del database da trasferire sullo Standbay per la creazione dell'istanza di standby. In particolare effettueremo un backup full del database e creeremo da RMAN i controlfile per lo Standby database; infine copieremo il bkup dalla stage area dell' host01 del Primary nel medesimo path sulla macchina dello Standby. Le operazioni da effettuare in sequenza dunque sono :

```
[oracle@host01 ~]$ rman target sys/*syspasswd*@TIGER
RMAN> backup device type disk format '/oracle/db/staging/%U' database plus archivelog;
RMAN> backup device type disk format '/oracle/db/staging/%U' current controlfile for standby;
```

4.6 Preparazione del Parameter File per lo Standby

Non ci rimane che creare il parameter file per lo Standby (initLION.ora) a partire dal parameter file del Primary e trasportarlo sullo Standby. Quindi ripristinare sullo Standby il database a partire dal backupset del Primary, dai controlfile opportunamente creati con la clausola "for standby" e dal nuovo parameter file riconfigurato in questo step. In rosso le parti del parameter file modificate per lo Standby.

```
SQL> create pfile='/u04/oracle/dg˙staging/initlion.ora' from spfile;
# ---[ Memory Parameters ] --- #
LION.__db_cache_size=905969664
LION.__java_pool_size=16777216
LION.__large_pool_size=16777216
LION.__shared_pool_size=285212672
LION.__streams_pool_size=0

# ---[ Dump Destination Parameters ] --- #
audit_file_dest='/oracle/db/product/11.2.0/admin/LION/adump'
background_dump_dest='/oracle/db/product/11.2.0/admin/LION/bdump'
core_dump_dest='/oracle/db/product/11.2.0/admin/LION/cdump'
user_dump_dest='/oracle/db/product/11.2.0/admin/LION/udump'

# ---[ Role-independent Parameters ] --- #
archive_lag_target=900
compatible='11.2.0'
control_files='/oracle/db/oradata/LION/controlfile/o1_mf_6hc6stn9_.ctl',
              '/oracle/db/flash_recovery_area/LION/controlfile/o1_mf_6hc6styy_.ctl'
```

```
db_name='TIGER'
db_domain='domain01.infn.it'
db_create_file_dest='/oracle/db/oradata'
db_recovery_file_dest='/oracle/db/flash_recovery_area'
dispatchers='(PROTOCOL=TCP) (SERVICE=LIONXDB)'
instance_name='LION'
log_archive_config='dg_config=(TIGER,LION)'
log_archive_max_processes=4
remote_login_passwordfile='exclusive'

# ---[ Primary Role Parameters ] --- #
db_unique_name='LION'
log_archive_dest_1=
'location=use_db_recovery_file_dest valid_for=(all_logfiles,all_roles) db_unique_name=LION'
log_archive_dest_2=
'service=TIGER.domain01.infn.it valid_for=(online_logfiles,primary_role) db_unique_name=TIGER'
log_archive_dest_state_1='enable'
log_archive_dest_state_2='enable'
service_names='LION.domain02.infn.it, LION'

# ---[ Standby Role Parameters ] --- #
db_file_name_convert='/TIGER/','/LION/'
log_file_name_convert='/TIGER/','/LION/'
fal_server='TIGER','LION'
fal_client='LION'
standby_file_management='auto'
```

Infine trasferisco con un semplice scp il parameter file sullo Standby.

```
[oracle@host01 ~]$ scp -rp /oracle/db/staging/* host02.domain02.infn.it:/oracle/db/staging/
```

5 La Configurazione dello Standby Database

Qui viene mostrato come creare l'istanza di Standby a partire dal backup del Primary, dai control file e dal parameter file creato e modificato appostitamente per lo Standby. In particolare, dal punto di vista delle credenziali di connessione tra TIGER e LION, ricordiamo che nella sezione 3 [page 23] abbiamo creato il passwd file per l'istanza Primary e lo abbiamo spedito, rinominato, allo Standby.

5.1 Creazione del *pfile* per lo Standby

A questo punto, verificate le connessioni di rete, verificate le credenziali di accesso con RMAN tra un database e l' altro, non resta che creare il pfile a partire dall' initLION.ora che avevamo riconfigurato e spedito allo Standby alla sezione precedente.

```
[oracle@vmlinux2 ~]$    sqlplus / as sysdba
SQL>    create spfile from pfile='/oracle/db/staging/initLION.ora';
SQL>    !ls -l $ORACLE_HOME/dbs

-rw-r----- 1 oracle oinstall 1536 Dec  8 22:20 orapwlion
-rw-r----- 1 oracle oinstall 4608 Dec  8 22:20 spfileLION.ora
```

5.2 Avvio del processo di Recover sullo Standby

Dando per scontata l' installazione corretta della ORACLE_HOME e de di tutti i path necessari per l'istnaza LION ci concentriamo sulla creazione dell' istanza vera e propria a partire dal bkup del Primary. Ricordiamo che è necessario impostare correttamente le variabili d' ambiente sullo Standby ($ORACLE_SID, $ORACLE_HOME,$PATH,$LD_LIBRARY_PATH), e assicurarsi che i path a seguire esistano.

```
/oracle/db/product/11.2.0/admin/LION/adump
/oracle/db/product/11.2.0/admin/LION/bdump
/oracle/db/product/11.2.0/admin/LION/cdump
/oracle/db/product/11.2.0/admin/LION/dpdump
/oracle/db/product/11.2.0/admin/LION/pfile
/oracle/db/product/11.2.0/admin/LION/scripts
/oracle/db/product/11.2.0/admin/LION/udump
/oracle/db/oradata/LION/controlfile
/oracle/db/oradata/LION/datafile
/oracle/db/oradata/LION/onlinelog
/oracle/db/flash_recovery_area/LION/archivelog
/oracle/db/flash_recovery_area/LION/autobackup
/oracle/db/flash_recovery_area/LION/backupset
/oracle/db/flash_recovery_area/LION/controlfile
/oracle/db/flash_recovery_area/LION/onlinelog
```

procedere con una prima verifica dei controlfile e dell' init file :

```
SQL>  startup nomount
```

5.3 Ricostruzione del Physical Standby con RMAN

Passiamo alla rigenerazione del Physical Standby tramite RMAN. Dopo esserci collegati allo Standby tramite RMAN ed aver avviato l' istanza LION in "no-mount" bisogna ricostruire i controlfile a partire da quelli contenuti nel backup del Primary:

```
[oracle@host02 ~]$  rman target sys/*syspasswd*@LION
RMAN>  restore controlfile from '/ora/db/dataguarddata/flash_recovery_area/LION/controlfile/o1_mf_6hc6styy_.ctl';
```

Quindi dalla macchina dello Standby ci si collega al Primary come target e allo Standby come auxiliary e si avvia il ripristino del database LION. In grigio viene mostrato un possibile output del comando RMAN. Al termine si porta in "mount" l' istanza e si avvia il processo di riapplicaizone automatica, in background, degli ORL ricevuti dal Primary.

```
RMAN>  duplicate target database for standby dorecover nofilenamecheck;


Starting Duplicate Db at 13-SEP-2012 12:22:37
using target database control file instead of recovery catalog
allocated channel: ORA_AUX_DISK_1
channel ORA_AUX_DISK_1: sid=156 devtype=DISK
contents of Memory Script:
{
   restore clone standby controlfile;
   sql clone 'alter database mount standby database';
}
executing Memory Script
Starting restore at 13-SEP-2012 12:23:03
using channel ORA_AUX_DISK_1
channel ORA_AUX_DISK_1: starting datafile backupset restore
channel ORA_AUX_DISK_1: restoring control file
channel ORA_AUX_DISK_1: reading from backup piece /oracle/db/staging/0flv2ss4_1_1
channel ORA_AUX_DISK_1: restored backup piece 1
piece handle=/oracle/db/staging/0flv2ss4_1_1 tag=TAG20101208T220036
channel ORA_AUX_DISK_1: restore complete, elapsed time: 00:00:01
output filename=/oracle/db/oradata/LION/controlfile/o1_mf_6j0m05vj_.ctl
output filename=/oracle/db/flash_recovery_area/LION/controlfile/o1_mf_6j0m068b_.ctl
Finished restore at 13-SEP-2012 12:23:04
sql statement: alter database mount standby database
contents of Memory Script:
{
   set newname for tempfile  1 to
 "/oracle/db/oradata/LION/datafile/o1_mf_temp_6hc6v3jd_.tmp";
```

```
   switch clone tempfile all;
   set newname for datafile  1 to
 "/oracle/db/oradata/LION/datafile/o1_mf_system_6hc6t6l0_.dbf";
   set newname for datafile  2 to
 "/oracle/db/oradata/LION/datafile/o1_mf_undotbs1_6hc6trl0_.dbf";
   set newname for datafile  3 to
 "/oracle/db/oradata/LION/datafile/o1_mf_sysaux_6hc6tyvd_.dbf";
   set newname for datafile  4 to
 "/oracle/db/oradata/LION/datafile/o1_mf_example_6hc6vf79_.dbf";
   set newname for datafile  5 to
 "/oracle/db/oradata/LION/datafile/o1_mf_users_6hc6vlf0_.dbf";
   restore
   check readonly
   clone database
   ;
}
executing Memory Script
executing command: SET NEWNAME
renamed temporary file 1 to /oracle/db/oradata/LION/datafile/o1_mf_temp_6hc6v3jd_.tmp in control file
executing command: SET NEWNAME
executing command: SET NEWNAME
executing command: SET NEWNAME
executing command: SET NEWNAME
executing command: SET NEWNAME
Starting restore at 13-SEP-2012 12:24:50
using channel ORA_AUX_DISK_1
channel ORA_AUX_DISK_1: starting datafile backupset restore
channel ORA_AUX_DISK_1: specifying datafile(s) to restore from backup set
restoring datafile 00001 to /oracle/db/oradata/LION/datafile/o1_mf_system_6hc6t6l0_.dbf
restoring datafile 00002 to /oracle/db/oradata/LION/datafile/o1_mf_undotbs1_6hc6trl0_.dbf
restoring datafile 00003 to /oracle/db/oradata/LION/datafile/o1_mf_sysaux_6hc6tyvd_.dbf
restoring datafile 00004 to /oracle/db/oradata/LION/datafile/o1_mf_example_6hc6vf79_.dbf
restoring datafile 00005 to /oracle/db/oradata/LION/datafile/o1_mf_users_6hc6vlf0_.dbf
channel ORA_AUX_DISK_1: reading from backup piece /oracle/db/staging/0clv2spd_1_1
channel ORA_AUX_DISK_1: restored backup piece 1
piece handle=/oracle/db/staging/0clv2spd_1_1 tag=TAG20101208T215908
channel ORA_AUX_DISK_1: restore complete, elapsed time: 00:00:45
Finished restore at 13-SEP-2012 12:25:35

contents of Memory Script:

   switch clone datafile all;

executing Memory Script

datafile 1 switched to datafile copy
input datafile copy recid=6 stamp=737245547 filename=
/oracle/db/oradata/LION/datafile/o1_mf_system_6j0m0gkf_.dbf
datafile 2 switched to datafile copy
input datafile copy recid=7 stamp=737245547 filename=
/oracle/db/oradata/LION/datafile/o1_mf_undotbs1_6j0m0gmf_.dbf
datafile 3 switched to datafile copy
input datafile copy recid=8 stamp=737245547 filename=
/oracle/db/oradata/LION/datafile/o1_mf_sysaux_6j0m0glf_.dbf
datafile 4 switched to datafile copy
input datafile copy recid=9 stamp=737245547 filename=
/oracle/db/oradata/LION/datafile/o1_mf_example_6j0m0gnn_.dbf
```

```
datafile 5 switched to datafile copy
input datafile copy recid=10 stamp=737245547 filename=
/oracle/db/oradata/LION/datafile/o1_mf_users_6j0m0gol_.dbf
Finished Duplicate Db at 13-SEP-2012 12:25:47
Recovery Manager complete.
```

```
SQL> alter database recover managed standby database disconnect from session;
```

Una variante dell'ultima istruzione RMAN che consente di applicare in "Real Time" gli ORL spediti dal Primay utilizza la clausola 'using current logfile' :

```
SQL> alter database recover managed standby database using current logfile disconnect;
```

A questo punto lo Stnadby è ricostruito e pronto a ricevere e riapplicare i redolog.

6 Avvio della Replica Geografica

Dopo esserci assicurati che lo Standby sia in grado di riapplicare i redolog torniamo sul Primary e cambiamo il parametro di inizializzazione log_archive_dest_state_2 da *'DEFER'* a **'ENABLE'** avviando così la trasmissione verso lo Standby degli ORL.

Consideriamo il fato che, in una configurazione Max Performance come quella presentata, il processo LGWR rende disponibile all'LNS un ORL solo dopo che è avvenuto un evento di switch log, ovvero dopo che ha riempito fino in fondo tutto lo spazio disponibile sul current redolog ed ha eletto a current il successivo redolog. Da questa consideraizone ne derivano altre due:

* La massima perdita di dati compatibile con un eventuale crollo dell' istanza Primary è pari alla dimensione di un redlog, quello marcato come 'current'

* Le operazioni di switch dei redolog sono tanto più frequenti quanto più piccoli sono i file di redolog e tanto più frequenti sono le operazioni transazionalmente rilevanti sul Primary.

Per forzare un primo switch dei redolog, e vedere così l'applicaizone sullo standby, basterà dare il comando seguente sul Primary e andare a verificare lo stato di applicazione degli SRL sullo standby.

```
SQL>  alter system archive log current;
```

> ***Attenzione :*** *Se nella fase di configurazione del Primary questo parametro non fosse stato impostato con "DEFER" è possibile che si incorra in errori di ricostruzione del database Standby e in gap di sincronizzazione tra i due db già allo stato iniziale. Ricordarsi inoltre di usare la clausola 'scoper=both' nel comando di alter per applicare l'effetto sia sul pfile che sull'spfile.*

7 Verifica e Monitoraggio

La prossima operazione da effettuare, è quella di verificare lo stato di allineamento tra Primary e Standby. Per fare questo basterà controllare il contenuto di alcune viste di sistema su entrambe le istanze. In particolare la V$ARCHIVED_LOG e la V$ARCHIVE_DEST consentono di verificare che il redolog(n-1) (preso il current come 'redolog(n)') venga correttamente archiviato e spedito allo Standby e da questo venga correttamente riapplicato sui datafile.

Possiamo quindi identificare due principali processi da monitorare; quello di "Trasporto" e quello di "Applay" degli SRL. E' anceh possibile monitorare il contenuto dell' Alert Log del database Standby per verificare che gli eventi di "media recovery" vadano a buon fine.

7.1 Verifichiamo lo stato della *Trasmissione* (Redo Transport) dei pacchetti OLR dal Primary allo Standby.

Questa query ci dice solo se la trasmissione avviene con successo ed in caso contrario con quali tipologie di errori. In particolare se la trasmissione ha successo nella vista V$ARCHIVE_DEST lo stato risulterà 'VALID' e non ci saranno segnalazioni di errori. In caso contrario lo stato sarà 'INVALID' e verrà popolato il campo 'ERROR' con la codifica del messaggio di errore utile per comprenderne le cause.

```
SQL>    alter system switch logfile;
SQL>    select status, error from v$archive_dest where dest_id = 2;

STATUS    ERROR
--------- --------------------------------------------------
VALID
```

7.2 Verifichiamo lo stato di *Applicaizone* (Redo Apply) degli OLR sullo Standby.

Per verificare lo stato di applicazione degli ORL tipicamente si interroga la vista di sistema **V$ARCHIVED_LOG** sullo Standby prima e dopo aver imposto l'archiviazione del "current redolog" sul Primary . Per imporre questa archiviazione si può eseguire uno switch log oppure la seguente *alter system* sul nodo Primary:

```
[oracle@host01 ~]$    sqlplus sys/sysypassword@TIGER as sysdba
SQL>  alter system archive log current;
```

Basterà quindi confrontare l' esito della sequente query subito prima, e subito dopo, l'*alter system* precedente per verificare che il nuovo ARL in arrivo dal Primary venga applicato sullo Standby. La query di verifica è :

```
SQL> select sequence#, first_time, next_time, archived, applied from v$archived_log order by sequence#;
```

7.3 Monitoraggio *alert.log* sullo Standby

In aggiunta ai controlli già presentati possiamo monitorare il contenuto dell'**alert.log** dello Standby per l'identificazione delle eventuali condizioni di errore.

Dall'host dello Standby possiamo verificare il contenuto dell'alert log con un semplice "tail -f" del file oppure con tecniche più sofisticate riportarne il contenuto in una external table e interrogare questa. Pr semplicità ci limitiamo a mostrare, in blu, un esempio dei record che troveremmo in un alert.log di uno Standby che stà riapplicando correttamnete gli SRL sulla propria istanza locale.

```
[oracle@host02 bdump]$ tail -f /oracle/db/admin/LION/bdump/alert_LION.log
....
RFS[2]: Successfully opened standby log 5: '/oracle/db/oradata/LION/onlinelog/o1_mf_5_6j0ml7nw_.log'
Fri Dec 10 10:14:51 EST 2010
Media Recovery Log /oracle/db/flash_recovery_area/LION/archivelog/2010_12_10/o1_mf_1_326_6j4jzbpc_.arc
Media Recovery Log /oracle/db/flash_recovery_area/LION/archivelog/2010_12_10/o1_mf_1_327_6j4jzcbm_.arc
Media Recovery Waiting for thread 1 sequence 328
Fri Dec 10 10:30:24 EST 2010
RFS[1]: Successfully opened standby log 4: '/oracle/db/oradata/LION/onlinelog/o1_mf_4_6j0mkynk_.log'
Fri Dec 10 10:30:26 EST 2010
Media Recovery Log /oracle/db/flash_recovery_area/LION/archivelog/2010_12_10/o1_mf_1_328_6j4kwjb5_.arc
Media Recovery Waiting for thread 1 sequence 329
Fri Dec 10 10:39:57 EST 2010
RFS[2]: Successfully opened standby log 4: '/oracle/db/oradata/LION/onlinelog/o1_mf_4_6j0mkynk_.log'
Fri Dec 10 10:39:58 EST 2010
Media Recovery Log /oracle/db/flash_recovery_area/LION/archivelog/2010_12_10/o1_mf_1_329_6j4lgfsr_.arc
Media Recovery Waiting for thread 1 sequence 330
```

8 Appendici

Nelle appendici è descritta una variante delle impostazini di rete che riguarda gli aspetti avanzati della configurazione dello strato di trasporto dei database Oarcle, in particolare la configurazione necessaria a garantire connessioni cifrate tra client e database o, come nel caso presente, tra diversi database.

8.1 Configurare Connessioni SSL con OraNet e Wallet

I passi di presentati riguardano da un lato il repository Oracle dei certificati, chiamato *Oracle Wallet*, e dall'altro lo strato di connettività dei database Oracle, Listener e TNS. Quest' ultimo sfrutta il wallet per instaurare connessioni su protocollo TCPS con tunnel SSL al posto delle normali connessioni TCP. Nel seguito ci riferiremo con il termine di 'server' a qualsiasi database che stia validando una connessione in ingresso, con il termine di 'client' qualsiasi database o application che stia cercando di instaurare una connessione con un server oracle. Fatta questa distinzione, ne consegue che in una coppia Data Guard entrambe le macchine rivestiranno alternativaente sia il ruolo di 'client', che quello di 'server'; da cui risulta importante che il wallet sia condiviso (o siano condivisi i certificati al suo interno) tra i due database. Per semplicità ci limitiamo all' assunto di configurare una connessione tra un client e un server.

8.1.1 Creare e Configurare un Wallet lato Server

Il 'wallet' in definitiva altro non è che un file binario in cui Oracle registra il certificato server, la CA e gli eventuali altri certificati necessari. Quindi prima di tutto dobbiamo creare il path corretto per il wallet; ad esempio un percorso potrebbe essere **ORACLE_HOME/wallet** .

Una volta creato il percorso ci si posiziona all' interno e si genera il wallet vuoto che poi verrà popolato con i certificati corretti. Per la creazione del wallet basta utilizzare il tool *orapki* a riga di comando come segue :

```
orapki wallet create -wallet /oracle/db/product/11.2.0/wallet
-auto_login -pwd *syspasswd*
```

Da notare anche che è stata utilizzata la passwd dell'utente sys (o system, purchè sia un utente che gode del ruolo sysdba) insieme al parametro *auto_login* che consente di evitare il prompt di richiesta della passwd ad ogni tentativo di connessione.

Un semplice comando **"ls -la"** sotto linux evidenzia che ora nella cartella sono presenti due nuovi file, *cwallet.sso* e *ewallet.p12*.

A questo punto, se non disponessi di un certificato per il client lo potrei creare self-signed insieme alla Root CA, altrimenti potrei utilizzare i seguenti comandi per caricare il DN dei certificati nel wallet. Per poter mettere in comunicazione due database il wallet dovrà infine essere condiviso tra i due.

```
orapki wallet add -wallet /oracle/db/product/11.2.0/wallet
-dn "CN=host01,L=domain01, O=INFN,C=IT" -keysize 512 -self_signed
validity 365 -pwd *syspasswd*

orapki wallet add -wallet /oracle/db/product/11.2.0/wallet
-dn "CN=host02,L=domain02, O=INFN,C=IT" -keysize 512 -self_signed
validity 365 -pwd *syspasswd*
```

L'arco di validità dei certificati all' interno del wallet risulta essere di 1 anno solare (parametro *-validity*), la chiave è codificata a 512 kb self-signed. Ispezioniamo il contenuto del wallet.

Come verifica possiamo ispezionare il contenuto del wallet:

```
[oracle@host01 ~]$ orapki wallet display -wallet .

Requested Certificates:
User Certificates:

Subject: CN=host01.domain01.infn.it,OU=server certificate,L=doamin01,O=INFN,C=IT
Subject: CN=host02.domain02.infn.it,OU=server certificate,L=doamin02,O=INFN,C=IT
Trusted Certificates:

Subject: CN=GTE CyberTrust Root,O=GTE Corporation,C=US

Subject: CN=host01.domain01.infn.it,OU=server certificate,L=doamin01,O=INFN,C=IT
Subject: CN=host02.domain02.infn.it,OU=server certificate,L=domain02,O=INFN,C=IT

Subject: OU=Class 3 Public Primary Certification Authority,O=VeriSign, Inc.,C=US
Subject: OU=Class 2 Public Primary Certification Authority,O=VeriSign, Inc.,C=US
Subject: OU=Class 1 Public Primary Certification Authority,O=VeriSign, Inc.,C=US
Subject: OU=Secure Server Certification Authority,O=RSA Data Security, Inc.,C=US
Subject: CN=GTE CyberTrust Global Root,OU=GTE CyberTrust Solutions, Inc.,O=GTE Corporation,C=US
Subject: CN=Entrust.net Secure Server Certification Authority,OU=(c) 2000 Entrust.net Limited,
         OU=www.entrust.net/SSL_CPS incorp. by ref. (limits liab.),O=Entrust.net
Subject: CN=Entrust.net Certification Authority (2048),OU=(c) 1999 Entrust.net Limited,
         OU=www.entrust.net/CPS_2048 incorp. by ref. (limits liab.),O=Entrust.net
Subject: CN=Entrust.net Secure Server Certification Authority,OU=(c) 1999 Entrust.net Limited,
         OU=www.entrust.net/CPS incorp. by ref. (limits liab.),O=Entrust.net,C=US
```

Nel wallet, oltre ai DN dei certificati di client e server, deve essere presente anche una Root CA condivisa tra i due interlocutori e che possa verificare i certificati di entrambi. Per questo motivo esportiamo la RootCA presente sul wallet del server in modo da poterla inserire poi nel wallet del client.

```
orapki wallet export -wallet /oracle/db/product/11.2.0/wallet
-dn  "CN=host01.domain01.infn.it, OU=server certificate,L=domain01, O=INFN,C=IT"
-cert host01.domain01_ca.cert
```

8.1.2 Configurazione dei componenti di rete con SSL

Per la riconfigurazione dei componenti di rete si può passare attraverso il classico User Net Manager, oppure agire direttamente sul contenuto dei file. In particolare vengono cambiati i parametri di protocollo e porte, in genere per TCPS si utilizza una porta alta, nell'esempio la 2484 al posto della 1521. I file da modificare sono: *listener.ora, tnsnames.ora* e *sqlnet.ora*.

listener.ora:

```
LISTENER =
   (DESCRIPTION_LIST =
  (DESCRIPTION =
     (ADDRESS = (PROTOCOL = TCP)(HOST = host01.domain01.infn.it)(PORT = 1521))
     )
     (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCPS)(HOST = hoat01.domain01.infn.it)(PORT = 2484))
     )
  )
WALLET_LOCATION =
  (SOURCE=
   (METHOD=File)
(METHOD_DATA=
  (DIRECTORY=/oracle/db/product/11.2.0/wallet)))
```

sqlnet.ora:

```
SQLNET.AUTHENTICATION_SERVICES= (BEQ, TCPS)
SSL_VERSION = 0
SSL_CLIENT_AUTHENTICATION = TRUE

WALLET_LOCATION =
  (SOURCE =
   (METHOD = FILE)
   (METHOD_DATA =
   (DIRECTORY = /oracle/db/product/11.2.0/wallet)
   )
  )
```

tnsnames.ora:

```
TIGER.domain01.infn.it =
  (DESCRIPTION =
(ADDRESS_LIST =
  (ADDRESS = (PROTOCOL = TCPS)(HOST = host01.domain01.infn.it)(PORT = 2484))
)
(CONNECT_DATA =
     (SERVICE_NAME = TIGER)
    )
   )
LION.domain02.infn.it =
  (DESCRIPTION =
(ADDRESS_LIST =
  (ADDRESS = (PROTOCOL = TCPS)(HOST = host02.domain02.infn.it)(PORT = 2484))
```

```
)
(CONNECT_DATA =
     (SERVICE_NAME = LION)
    )
   )
```

8.1.3 Configurazione del database

A questo punto dobbiamo configurare i parametri di gestione dell'autenticazione del database. Per farlo andiamo innanzi tutto a settare i valori corretti per le variabili OS_AUTHENT_PREFIX e REMOTE_OS_AUTHENT. Quindi riavvieremo l' istanza in oggetto

```
alter system set remote_os_authent=FALSE scope=spfile;
alter system set os_authent_prefix='' scope=spfile;
```

8.1.4 Testing Connections

Il primo test da effettuare è, come sempre, tramite il *tnsping* che verifica la connettività di rete tra le istanze. In entrambi i casi la connection string utilizzata dal tnsping deve riportare l'utilizzo del protocollo **TCPS** con la porta asociata, nel nostro esempio la **4848**.

```
$> tnsping LION
...
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL =TCPS)
(HOST = host02.domain02.infn.it)(PORT = 2484))) (CONNECT_DATA = (SID = LION)))
OK (100 msec)

$> tnsping TIGER
...
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL =TCPS)
(HOST = host01.domain01.infn.it)(PORT = 2484))) (CONNECT_DATA = (SID = TIGER)))
OK (100 msec)
```

Il secondo test è quello di tentare una connessione dal client, con utente sys, e monitorarne l'esito sul listener.log del server. Ad esempio potremmo usare il seguente comando dal Primary per tentare la connessione sullo standby e controllaer sul listener.log di quest'ultimo l'esito delle operazioni.

```
$> sqlplus sys/*syspasswd*@LION
```

Riferimenti bibliografici

[1] Oracle. *Oracle 10g New Features Of LogMiner (Doc ID 249001.1)*, December 2012.

[2] Oracle. *Oracle Database Security Guide 11g Release 1 (11.1) - B28531-19*, September 2012.

[3] Oracle. *Certification Information for Oracle Database on Linux x86-64 (Doc ID 1304727.1)*, August 2013.

[4] Oracle. *Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration (Doc ID 413484.1)*, March 2013.

[5] Oracle. *Oracle Database Installation Guide 11g Release 2 (11.2) for Linux E47689-03*, August 2013.

[6] Larry Carpenter Joe Meeks Charles Kim Bill Burke Sonya Charoters Joydip Kundu Michael Smith Nitin Vengurlekar. Oracle Data Guard 11g Handbook. Oracle Press, 2009.