Alma Mater Studiorum · Università di Bologna
facoltà di scienze matematiche, fisiche e naturali
Corso di Laurea Triennale in Informatica

Implementazione di un log server centralizzato per l'adempimento alla normativa 27 novembre 2008 del Garante della Privacy

Tesi di Laurea in Architettura degli Elaboratori

Relatore:
Chiar.mo Prof.
Ghini Vittorio

Presentata da:
Renzini Gabriele

Sessione terza
Anno Accademico 2010/2011

Alle mie donne...

Maria, Amelia, Isabella, Mirella e Ginevra

Indice

Elenco delle figure

Capitolo 1

Studio della legislazione in materia di privacy e trattamento dei dati personali

Il 27 novembre 2008 il Garante della Privacy ha emanato una normativa che interessa tutti coloro che per lavoro si trovano a dover gestire e manipolare dati sensibili e non, soprattutto in ambito informatico.

Il provvedimento segue il Codice in materia di protezione dei dati personali (d.lg. 196/2003), in particolare la normativa fa riferimento agli articoli 31 (Obblighi di sicurezza) e 154 (Compiti) comma 1, del Codice e al suo allegato B (Disciplinare tecnico in materia di misure minime di sicurezza). L'art. 31 del Codice indica al Titolare l'obbligo di adottare misure di sicurezza che siano *"idonee e preventive"*, onde evitare, la perdita o la distruzione, anche accidentale, dei dati, accessi non autorizzati o il trattamento non consono alle finalità di raccolta. L'art. 154 stila i compiti che il Garante ha nei confronti del Codice, nello specifico al comma 1 lett c) viene dato il compito al Garante di prescrivere le misure necessarie affinché il trattamento si svolga

secondo le disposizioni vigenti mentre alla lett h) viene indicato il compito di curare la diffusione della *"disciplina rilevante in materia di trattamento dei dati personali e delle relative finalità, nonché delle misure di sicurezza dei dati"* L'allegato B, infine, è un vero e proprio indice di disposizioni tecniche da mettere in atto da parte del Titolare, del Responsabile ove designato e dell'Incaricato nel caso in cui i dati siano trattati con l'ausilio di strumenti elettronici.

Il provvedimento, invece, si rivolge nello specifico ai cosiddetti "amministratori di sistema" (AdS) ovvero a quelle *"figure professionali finalizzate alla gestione e alla manutenzione di un impianto di elaborazione o di sue componenti"*[1]. Questa definizione è data dal Garante ma lo stesso Garante estende il provvedimento e rende quindi equiparabili anche altre figure professionali quali gli amministratori di basi di dati, gli amministratori di reti e apparati di sicurezza e gli amministratori di sistemi software complessi.

Il motivo per cui questo provvedimento è rivolto in particolar modo agli AdS è che questi per la natura del loro lavoro hanno la possibilità di accedere intenzionalmente o accidentalmente a dati personali, che possono essere sensibili o non, ai quali in realtà non sono autorizzati ad accedere.

Operazioni svolte sistematicamente dagli AdS possono essere equiparabili infatti al trattamento di dati personali, alcuni esempi sono le operazioni di Backup/Recovery dei dati o la manutenzione hardware. Il Garante precisa tra le altre cose che queste operazioni sono da considerarsi al pari del trattamento di dati personali *"anche quando l'amministratore non consulti "in chiaro" le informazioni medesime".*

La criticità del ruolo di AdS è riconosciuta anche dal Codice Penale che prevede delle aggravanti se taluni reati sono commessi da persone che abu-

[1] `http://www.garanteprivacy.it/garante/doc.jsp?ID=1577499`

sano della loro posizione di operatori di sistema. Gli articoli sono quelli che prevedono l'accesso abusivo a un sistema informatico o telematico (art. 615 ter), di frode informatica (art. 640 ter), di danneggiamento di informazioni, dati e programmi informatici (art. 635 bis e ter) e di danneggiamento di sistemi informatici o telematici (art. 635 quater e quinques).

Il Codice, tuttavia, non ha al suo interno una definizione di AdS ma fa riferimento a operazioni che sono comunemente svolte da questo tipo di figure professionali, in particolare l'allegato B del Codice, che riguarda le misure minime di sicurezza, detta le modalità di gestione dei sistemi informatici.

Da quanto espresso fin qui si può dunque intuire la delicata posizione che ricoprono gli AdS che hanno la responsabilità di operazioni che vanno dalla salvaguardia contro il furto o la manipolazione da parte di persone non autorizzate, al ripristino in caso di incidente su dati personali.

Le principali novità apportate dal provvedimento rispetto al Codice sono la nomina degli AdS, la loro verifica da parte del Titolare e la tracciabilità degli accessi logici degli AdS.

La nomina degli AdS spetta al Titolare del trattamento o all'eventuale Responsabile da lui designato. Il Titolare o il Responsabile hanno il compito di scegliere l'AdS valutandone l'esperienza, la capacità e l'affidabilità. Una volta individuati gli AdS dovrà essere stilata una lista degli ambiti di esercizio ai quali i singoli AdS dovranno attenersi, tale lista dovrà comparire in un documento interno da mantenere aggiornato e che dovrà essere disponibile in caso di verifica da parte del Garante. Anche nel caso di sistemi gestiti da esterni il Titolare o il Responsabile dovranno conservare gli identificativi delle persone incaricate all'amministrazione dei sistemi. Gli AdS in seguito alla nomina ricevuta dovranno essere oggetto di verifica almeno annuale da parte del Titolare o del Responsabile che dovranno controllare se l'attività svolta è

rispondente a quanto dettato dalla normativa. Infine il Garante richiede che venga anche tenuta traccia degli accessi logici (login/logout) degli AdS nei sistemi di elaborazione e di archiviazione dei dati. Gli access log dovranno, secondo la normativa, avere le caratteristiche di completezza, inalterabilità e verifica di integrità e dovranno essere mantenuti per un periodo non inferiore ai 6 mesi.

Mentre i primi punti sono strettamente burocratici questa ultima parte della normativa è di fatto quella che deve essere implementata direttamente sui sistemi di elaborazione e di archiviazione dati. Il fatto che il Garante non dia indicazioni su come implementare la soluzione a questo problema lascia aperta la strada a diverse ipotesi, quello che è difficile è riuscire a interpretare correttamente il volere del Garante e cercare di essere il più possibile aderenti a quanto richiesto. Le strade che si possono a questo punto intraprendere sono 2, o acquistare un prodotto che soddisfi queste esigenze oppure implementarlo autonomamente a costi sicuramente inferiori. La seconda soluzione, che è poi quella scelta per lo sviluppo di questo studio, comporta però un problema di fondo che è quello che il controllore è di fatto anche il controllato in quanto chi implementa la soluzione è molto spesso lo stesso AdS che deve essere loggato nelle sue operazioni. Il Garante ad ogni modo non solleva questo problema e non esclude quindi che gli stessi AdS siano poi coloro che realizzano il sistema di logging. C'è da tenere anche conto dei costi che prevede l'acquisto di una soluzione commerciale e spesso imprese o amministrazioni pubbliche di piccole o medie dimensioni non hanno risorse economiche necessarie a coprire questa spesa.

Evitando di trattare le questioni riguardanti i criteri di nomina degli AdS passiamo a valutare quelle che sono le criticità della parte tecnica del provvedimento.

La soluzione a prima vista potrebbe sembrare banale da realizzare ma c'è da tenere conto che il provvedimento è poco chiaro e lascia aperte molte problematiche, che ad oggi non trovano risposta nemmeno nelle FAQ che il Garante ha inserito a margine del provvedimento, su come implementare tecnicamente un sistema che sia rispondente a quanto chiesto.

Una delle questioni più difficili da interpretare è la parte che riguarda le caratteristiche di completezza, inalterabilità e possibilità di verifica dell'integrità dei log. E' evidente che questo provvedimento ha come finalità quella di tracciare gli AdS onde evitare che questi facciano un uso non consono o non lecito dei dati che hanno in gestione ma tiene poco conto dei privilegi di cui godono molti AdS. Gli AdS molto spesso hanno, per la natura del loro lavoro, la possibilità di accedere con privilegi di super utente (es. Administrator per i sistemi Windows o root per quelli Unix/Linux) alle macchine da loro gestite. Con questa premessa le caratteristiche richieste dal Garante sono pressochè impossibili da attuare in quanto i log possono essere manomessi e quindi alterati prima che venga in qualche modo verificata la loro integrità da coloro che dispongono di privilegi di super utente.

Nella FAQ n°12[2] il Garante cerca di esplicitare il concetto di inalterabilità dei log, parla di come deve essere interpretata questa caratteristica e cita per i casi più complessi la possibilità di adottare log server centralizzati e "certificati". A quest'ultimo termine però poi non corrisponde un'ente preposto alla certificazione che possa quindi darci una lista di quelli che sono i cosiddetti sistemi "certificati" e quali sono le caratteristiche che questi sistemi hanno rispetto agli altri. Molti vendors spacciano come certificate soluzioni che poi non sono altro che software open-source già preinstallato su hardware da loro vendoto. Anche queste soluzioni però non garantiscono tutte le caratte-

[2] `http://www.garanteprivacy.it/garante/doc.jsp?ID=1577499#12`

ristiche che il Garante richiede in quanto anche se l'AdS non ha l'accesso al sistema di log centralizzato potrebbe sempre alterare i log prima che questi siano inviati al server, fare in modo che i log non vengano inviati rendendo il server non raggiungibile oppure manomettendo fisicamente il server nel caso l'AdS avesse accesso al locale dove il server è posizionato.

Un altro problema è quello della raccolta degli access log dei DBMS. Anche questi sistemi rientrano nel provvedimento come specificato nella FAQ n°19[3] ma non sempre esiste un modo agevole per tenere traccia degli accessi logici. Per fare un esempio MySQL, uno dei DBMS più diffusi al mondo, non ha un sistema idoneo a effettuare quanto richiesto dalla normativa, l'unico modo è quello di abilitare il "debug mode", in questo caso tutto quello che fa il software (login/query/logout/etc.) viene scritto anche su file, questo è però altamente sconsigliato e lo si può leggere proprio nei commenti del file di configurazione di MySQL (*"# Be aware that this log type is a performance killer."*) in quanto richiede grosse quantità di spazio disco e allo stesso tempo degrada enormemente le prestazioni del sistema.

Altro problema significativo è dato da i software che non prevedono la registrazione degli accessi logici o la prevedono solo in parte. Alcuni software sviluppati anche da grandi aziende non prevedono la registrazione degli accessi in tutti i casi. Per esempio alcuni software hanno integrata la possibilità di registrare ed inviare tramite syslog gli accessi degli utenti solo nel caso che gli accessi avvengano da riga di comando, se invece gli accessi avvengono lato applicativo (per es. da interfaccia web) questi non vengono catturati. Nel caso il software in questione sia open-source chi si trova a doverlo utilizzare potrebbe anche valutare l'ipotesi, se ne fosse in grado o avesse fondi per poterlo finanziare, di adeguarlo a quanto richiesto dal Garante, purtrop-

[3] `http://www.garanteprivacy.it/garante/doc.jsp?ID=1577499#19`

po però questo non sempre è possibile. Nel caso infatti che il software sia sviluppato da aziende private che non rilasciano i sorgenti sarebbe obbligatoriamente necessario chiedere all'azienda di adeguare il software ma non è detta che questo sia fattibile in quanto i costi potrebbero essere troppo elevati o l'azienda potrebbe non avere interesse a questo sviluppo. All'utilizzatore a questo punto non resta che decidere se rischiare e tenere un software non a norma con quanto richiesto dalla normativa oppure cambiare, sempre che esista un'alternativa, e migrare su una nuova applicazione. Entrambe le soluzioni sono comunque dispendiose per chi si trova a doverle affrontare, da un lato infatti c'è il rischio di richiamo da parte del Garante con relativa sanzione pecuniaria, dall'altro c'è un dispendio di tempo e risorse per cambiare e imparare ad usare un nuovo sistema.

Un esempio che questo può accadere è evidente dal provvedimento che il Garante ha preso nei confronti dell'Istituto Poligrafico e Zecca delle Stato s.p.a.[4] dove questo scenario si è presentato. Nella fattispecie il Garante ha accertato la parziale attuazione della normativa in quanto 2 software in dotazione all'Istituto Poligrafico ottemperavano alla normativa solo per quanto riguarda gli accessi di tipo sistemistico mentre non registravano gli accessi di tipo applicativo. Il Garante ha quindi prescritto l'immediata attuazione di quanto specificato nella normativa riguardante gli AdS.

Un sistema che possa quindi essere affidabile al 100% non può esistere, cercheremo comunque di essere più possibile aderenti a quanto richiesto dal Garante sfruttando, per quanto possibile, solo prodotti open-source e cercando di limitare al massimo la possibilità che un AdS con privilegi di super utente possa in qualche modo aggirare il sistema.

[4] `http://www.garanteprivacy.it/garante/doc.jsp?ID=1829641`

Capitolo 2

Studio di fattibilità del progetto in base all'infrastruttura di riferimento

La mia intenzione è quella, per quanto possibile, di utilizzare software libero per lo sviluppo del progetto poiché questo renderà più facile adattare il progetto alle mie esigenze e a quelle di chi ne vorrà usufruire nonché meno oneroso rispetto a soluzioni commerciali che offrono gli stessi servizi.

Prima di iniziare l'implementazione vera e propria del sistema di log centralizzato si sono analizzati i 3 principali sistemi di logging open-source esistenti, ovvero syslog, syslog-ng e rsyslog. Syslog è stato lo standard di fatto su quasi tutte le distribuzioni Linux fino a poco tempo fa, ma rispetto a syslog-ng e rsyslog ha molte carenze di funzionalità che hanno fatto sì che negli ultimi tempi sia stato progressivamente sostituito da uno di questi due syslog-ng o rsyslog a seconda delle distribuzioni.

L'implementazione ha come riferimento l'Università degli
Studi di Perugia e in particolare quella dell'Ufficio Sistemi Gestionali, ufficio

al quale afferisco e che mi ha consentito di portare avanti lo sviluppo. Le scelte legate a tale progetto hanno quindi tenuto conto di quella che è la situazione attuale dell'infrastruttura dell'ateneo perugino. L'Ufficio Sistemi Gestionali gestisce perlopiù macchine Red Hat Enterprise Linux e alcune macchine Windows quasi tutte virtualizzate all'interno di un cluster VMware.

Come sistema operativo per il log server si è scelto Linux per poter mantenere il progetto più possibile "open". Per quanto riguarda la scelta della distribuzione si è optato di mantenere l'omogeneità presente e quindi di adottare Red Hat Enterprise Linux, anche se questa distribuzione è commerciale non pregiudica la possibilità di poter replicare tale sistema su altre distribuzioni totalmente free.

La scelta finale è quindi ricaduta su rsyslog per 3 motivi:

- Rsyslog supporta lo spooling dei messaggi su disco che permette di evitare la perdita di messaggi nel caso in cui il log server non sia raggiungibile mentre syslog-ng lo permette solo nella versione commerciale.

- Rsyslog supporta il protocollo RELP (Reliable Event Logging Protocol) il quale dovrebbe fare in modo che non vi sia la possibilità di perdita di messaggi tra client e server, cosa che non è possibile ottenere con le classiche connessioni TCP[1].

- I sistemi sul quale si sarebbe implementato il sistema di logging avevano già rsyslog installato di default, questo consente una più facile gestione in quanto gli aggiornamenti software sono rilasciati direttamente da Red Hat (Rsyslog è il default anche su CentOS, Debian, Ubuntu e altre distribuzioni).

[1] http://blog.gerhards.net/2008/04/on-unreliability-of-plain-tcp-syslog.html

La versione di rsyslog presente nell'ultima versione di Red Hat è la 4.6.2 ed è quella che verrà usata nel progetto.

In questa fase preliminare si sono effettuati dei test sul protocollo RELP per valutarne l'effettiva efficienza utilizzando due macchine virtuali. I test hanno riguardato perlopiù la simulazione di problemi legati alla connessione di rete tra client e server. I risultati sono stati buoni tranne che per un caso in cui dei messaggi sono risultati persi. L'evoluzione del caso in cui si verifica la perdita dei messaggi è questa:

1- il link tra client e server viene interrotto

2- il server rsyslog viene stoppato

3- il client riavvia il servizio rsyslog

4- il server rsyslog viene riavviato

5- il link viene ripristinato

In questa circostanza il client non riesce ad accorgersi nell'immediato che il server si è stoppato e quindi lo spooling dei messaggi non viene attivato subito. Nel lasso di tempo che intercorre quindi tra l'interruzione del link e l'avvio dello spooling i messaggi risultano persi. Eccetto questo caso il protocollo RELP sembra effettivamente essere in grado di gestire meglio del protocollo TCP i casi di malfunzionamento.

Questa versione oltre al supporto del protocollo RELP ha la possibilità di scrivere i log oltre che su disco anche su DB. Questa funzione sarà sfruttata per rendere accessibili i log via web tramite il software loganalyzer[2] e per aumentare la sicurezza del sistema. Loganalyzer è un software GPL scritto in php in grado di leggere i log presenti nel database ed è sviluppato da Adiscon, la stessa azienda che porta anche avanti lo sviluppo di rsyslog.

[2]http://loganalyzer.adiscon.com/

La possibilità di accedere ai log via interfaccia web servirà ad agevolare il compito del Titolare o del Responsabile del trattamento che ogni anno deve verificare che il compito svolto dagli AdS rientri all'interno di quanto stabilito dalla normativa. Ovviamente l'accesso all'interfaccia web dovrà essere opportunamente protetto in modo da rendere visibili i log solo a chi di dovere. Il salvataggio dei log su DB rende inoltre il sistema più sicuro in quanto replica quelli che sono i log comunemente scritti su disco.

Questo non basta a rendere i log immutabili come richiesto dal Garante in quanto un eventuale accesso fraudolento con privilegi da super-utente potrebbe in ogni caso compromettere la validità dei log ma sicuramente questa opzione, aumentando la complessità del sistema, rende l'alterazione completa dei dati più difficile.

I log oltre ad essere scritti su file e su DB verranno inoltre quotidianamente masterizzati su supporti non riscrivibili e la lista degli hash md5 dei singoli file sarà inviata per email in modo da avvicinarci più possibile alla caratteristica di inalterabilità richiesta.

Dal momento che tutta l'infrastruttura è virtuale anche il log server avrà quindi questa caratteristica. Questo aspetto pone però dei problemi per quanto riguarda l'ultimo processo che abbiamo visto, ovvero la masterizzazione. La scelta a questo punto è stata quella di cercare un modo per far accedere la macchina virtuale ad un device che non fosse vincolato al server sul quale la macchina virtuale si trovava in quel momento, onde evitare problemi con alcune funzioni di VMware, come per esempio il vmotion che sposta le macchine virtuali da un nodo ad un altro del cluster.

La soluzione che è sembrata più opportuna è stata quella di adottare un device usb e di collegarlo ad un server che condividesse il device via ethernet. In questo modo la macchina virtuale non farà riferimento al device fisico del

nodo in cui si trova in quel momento ma farà riferimento ad un device che sarà collegato via rete.

Per poter mettere in pratica questa soluzione quello di cui abbiamo bisogno è un masterizzatore usb e un software che possa condividere il masterizzatore usb via rete. Ci sono svariati software preposti a tale operazione, ma la maggior parte ha licenze commerciali e sono funzionanti solo su piattaforma Windows. Tra i rari progetti degni di nota per la piattaforma linux si segnalano USB/IP project[3], con licenza GPL, mentre con licenza proprietaria ma free USB Redirector[4].

Nel caso specifico il blade di cui disponiamo ha oltre alle macchine del cluster VMware altre 2 macchine dedicate ad altri compiti (una Linux dedicata al sistema di backup e una Windows che serve da Vcenter server per il cluster) e sarà una di queste macchine ad essere adottata come usb server.

Per quanto riguarda la parte client tutte le macchine linux sono già dotate del software necessario in quanto tutte le distribuzioni dispongono di un sistema di logging. Sarà quindi sufficiente configurare il sistema per fare in modo che possa inviare i log al server centralizzato. Per le macchine con sistema Windows non esiste invece la possibilità di inviare nativamente gli eventi generati ad un sistema di logging centrale e questo pone il problema di dover trovare un software in grado di svolgere questo lavoro.

I software che si sono valutati sono sia GPL che commerciali e sono Datagram Syslog Agent[5] (GPL), Winlogd[6] (GPL), NTSyslog[7] (GPL), Intersect

[3]http://usbip.sourceforge.net
[4]http://www.incentivespro.com
[5]http://syslogserver.com/syslogagent.html
[6]http://edoceo.com/creo/winlogd
[7]http://ntsyslog.sourceforge.net/

Snare Agent[8] (GPL e commerciale) e Adiscon EventReporter[9] (commerciale). Dei software sopra elencati nessuno di quelli GPL supporta connessioni TCP ma solo connessioni con protocollo UDP. Questo pone problemi per quanto riguarda la completezza e l'integrità dei log richiesta dal Garante poiché il protocollo UDP non assicura in nessun modo che il pacchetto sia stato effettivamente consegnato al server e che quindi il log sia stato scritto. Per cercare di rispettare quanto scritto nel provvedimento del Garante dovremo in questo caso avvalerci di un prodotto commerciale. Per quanto riguarda le caratteristiche tecniche necessarie allo sviluppo del progetto, la versione commerciale di Snare Agent e EventReporter si equivalgono; la mia scelta quindi è ricaduta sul prodotto Adiscon poiché questa azienda è comunque molto attiva nello sviluppo di software GPL: il progetto rsyslog è infatti portato avanti da loro e spero che l'acquisto possa incentivarne lo sviluppo.

Dal momento che la normativa a proposito degli AdS comprende anche coloro che gestiscono basi di dati, reti, apparati di sicurezza e sistemi software complessi dovremo prevedere non solo la registrazione dei log di sistema delle macchine ma anche di tutti quei software che rientrano nelle categorie elencate dal Garante.

Per quello che riguarda l'Ufficio Sistemi Gestionali rientrano nella normativa anche il sistema di virtualizzazione VMware vSphere, il software di backup IBM Tivoli Storage Manager e i database Oracle. Nessuno di questi software prevede nativamente la possibilità di inviare i propri log ad un server log remoto; sarà quindi necessario adottare misure particolari per colmare questa mancanza.

[8] http://www.intersectalliance.com/projects/SnareWindows/index.html
[9] http://www.eventreporter.com

Capitolo 3

Implementazione del sistema di logging

3.1 Configurazione rsyslog lato server

Il primo passo dell'implementazione è stato quello di preparare una macchina dedicata alla raccolta dei log. Per la prima fase di test è stato adottato un server dedicato dotato di masterizzatore CD per poter ottemperare alle necessità di copiare i log su supporti non riscrivibili. L'installazione minimale di Red Hat prevede già tra i suoi pacchetti rsyslog quindi si è semplicemente proceduto ad installare il sistema con questa caratteristica.

Non avendo idea della mole di dati che sarebbero arrivati al log server si è proceduto con la creazione di una partizione dedicata a questo scopo. Le partizioni del sistema sono state opportunamente configurate per l'utilizzo con LVM così da poterne gestire lo spazio nel migliore dei modi.

Terminata l'installazione si è poi configurato di rsyslog. Per prima cosa si sono abilitati, caricandone i moduli, i protocolli UDP, TCP e RELP dal quale il log server avrebbe accettato le connessioni dai client sulle diverse

porte:

```
# Provides UDP syslog reception
$ModLoad imudp.so
$UDPServerRun 514

# Provides TCP syslog reception
$ModLoad imtcp.so
$InputTCPServerRun 514

# Provides RELP syslog reception
$ModLoad imrelp
$InputRELPServerRun 20514
```

Il motivo per cui si è deciso di abilitare tutti i protocolli è perché il protocollo RELP essendo ancora poco diffuso ed è supportato solo da rsyslog. Il protocollo più diffuso è il TCP ma alcuni dispositivi, quali possono essere per esempio switch o simili, potrebbero supportare il solo protocollo UDP. In questo modo il server, accettando tutti e tre i protocolli, da la possibilità a tutti i dispositivi di inviare log.

A questo punto il server è pronto per ricevere i log dai vari client, ma i log verrebbero tutti scritti nei file di default senza nessun tipo di distinzione in base a data e host di provenienza. Per semplificare la lettura dei log, dal momento che rsyslog supporta i template e che questi permettono di specificare un formato o un nome file personalizzato, si è provveduto a configurarne alcuni allo scopo di separare i log in arrivo in base alla data, all'host e al tipo di applicazione:

```
#### TEMPLATES ####

$template Messages, \
"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/%programname%.log"
$template Secure,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/secure"
$template Maillog,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/maillog"
$template Cron,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/cron"
```

```
$template Spooler,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/spooler"
$template Boot,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/boot.log"
$template Kern,"/var/log/remote-log/%$YEAR%-%$MONTH%-%$DAY%/%hostname%/kern.log"
```

In questo modo i log saranno suddivisi prima per data di ricezione, poi per host e infine per tipo di log. Dal momento che la masterizzazione dei log si è convenuto schedularla una volta al giorno appena passata la mezzanotte, questa suddivisione rende più agevole individuare i log da masterizzare poiché è sufficiente prendere tutto il contenuto della directory indicante la data del giorno precedente.

Oltre ai template, nel file di configurazione sono presenti altri parametri tra cui il tipo di formato adottato per la scrittura dei log, che in questo caso è rimasto il default, e i permessi con cui vengono creati file e directory:

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022
```

Ora c'è da specificare le regole con cui i log vengono direzionati ai vari template in base alla loro facility e alla severity:

```
#### RULES ####

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none                ?Messages

# The authpriv file has restricted access.
authpriv.*                                              ?Secure

# Log all the mail messages in one place.
mail.*                                                  -?Maillog

# Log cron stuff
```

```
cron.*                                                  ?Cron

# Everybody gets emergency messages
*.emerg                                                 *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                          ?Spooler

# Save boot messages also to boot.log
local7.*                                                ?Boot

# Log all kernel messages
kern.*                                                  ?Kern
```

In questo modo ogni tipologia di log ha specificato un suo template e sarà semplice riuscire ad identificare i log che ci occorrono.

Infine non ci resta che configurare rsyslog per inviare i log anche al database MySQL che servirà anche per l'accesso via web ai log. Prima di tutto è necessario caricare il modulo per l'invio dei log a MySQL, nelle righe successive è invece necessario inserire le direttive dove viene specificata la porta di connessione al database, le azioni da intraprendere nel caso il database non sia raggiungibile e infine la direttiva per l'inoltro dei log:

```
$ModLoad ommysql

$ActionOmmysqlServerPort 3306

# An on-disk queue is created for this action. If the remote host is
# down, messages are spooled to disk and sent when it is up again.
$WorkDirectory /var/spool/rsyslog # where to place spool files
$ActionQueueFileName fwdRule1 # unique name prefix for spool files
$ActionQueueMaxDiskSpace 1g   # 1gb space limit (use as much as possible)
$ActionQueueType LinkedList   # run asynchronously
$ActionResumeRetryCount -1    # infinite retries if host is down
$ActionQueueSaveOnShutdown on # save messages to disk on shutdown

*.*       :ommysql:db_host,db_name,db_username,db_password;dbFormat
```

Le direttive centrali sono quelle che servono a far sì che i messaggi non vengano persi nel caso il database non sia per qualche motivo raggiungibile. La prima specifica la directory di spool dove i messaggi saranno scritti su disco nel caso non sia più possibile usare la coda in ram, a seguire è specificato il nome del file di spool, la sua grandezza limite, il tipo di coda, quante volte si tenterà di ricontattare il database e infine la direttiva che salva su disco il contenuto della coda nel caso in cui rsyslog venisse stoppato.

Il passo successivo è quello di creare il database nel quale verranno scritti i log. Lo script per la creazione del database (createDB.sql[1]) è solitamente fornito insieme al pacchetto rsyslog ed è quindi sufficiente eseguirlo come utente root di mysql per completare l'operazione di creazione. Terminata la creazione del database dovremo procedere ad aggiungere un utente che abbia i privilegi di accesso al database:

```
mysql> CREATE USER 'db_username'@'rsyslog_host' IDENTIFIED BY 'db_password';
mysql> GRANT ALL PRIVILEGES
    ->     ON db_name.*
    ->     TO 'db_username'@'rsyslog_host' WITH GRANT OPTION;
```

Dal momento che la password del database compare in chiaro all'interno del file di configurazione è opportuno mettere in sicurezza il file dandone accesso solo all'utente root:

```
# chmod 640 /etc/rsyslog.conf
```

Il database è ora pronto ad essere popolato dai log ricevuti da rsyslog dai vari host, ma questi log dovranno necessariamente essere cancellati dopo un certo periodo di tempo in quanto le dimensioni potrebbero diventare ragguardevoli già dopo poco tempo. Serve dunque uno script che schedulato rimuova i log più vecchi di una certa data con cadenza regolare:

[1]Si veda Appendice B pag. 79 per lo script completo

```
mysql -u db_username -p db_password -e \
"DELETE FROM SystemEvents \
WHERE ReceivedAt < date_add(current_date, interval -365 day)" db_name
```

In questo esempio vengono cancellati i log più vecchi di un anno. Come possiamo vedere anche in questo caso la password è in chiaro all'interno della procedura, occorre quindi modificare adeguatamente i permessi dello script in modo che il file sia leggibile solo dall'utente root come fatto precedentemente per il file di configurazione di rsyslog.

3.2 Configurazione dei client

3.2.1 Client Linux

Tutte le macchine alle quali gli AdS hanno accesso devono inviare i loro log al sistema di logging centralizzato, per fare questo è necessario anche qui intervenire nella configurazione di rsyslog.

Come per la configurazione lato server anche per i client vanno abilitati i vari moduli necessari, se la versione di rsyslog del client supporta RELP procederemo ad attivare il modulo che consente l'invio attraverso questo protocollo:

```
# Provides sending syslog messages over RELP
$ModLoad omrelp
```

se tale modulo non è presente o non è' supportato potremo utilizzare il classico TCP, per l'utilizzo di questo protocollo come per quello UDP non è necessario caricare nessun modulo in quanto supportati nativamente.

Per motivi sia pratici che di sicurezza i log generati dal client verranno scritti anche sul filesystem locale oltre che inviati al server centrale; questo permette di poter consultare i log della macchina senza dover accedere al

server e ci permette di avere riscontro dei log presenti nel server anche lato client, infatti tutti i log presenti sul server dovranno essere presenti nel client e viceversa. Se cosi non fosse si dovrà esaminare il motivo della discrepanza tra client e server.

Le configurazioni delle direttive globali e delle regole possono quindi rimanere quelle di default già presenti nel file di configurazione, in questo caso l'unica eccezione è rappresentata dai kernel log che di default sono commentati e che invece si sono impostati per essere scritti nel file kern.log:

```
#### GLOBAL DIRECTIVES ####

# Use default timestamp format
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

#### RULES ####

# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*              /dev/console
kern.*                                          /var/log/kern.log

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none        /var/log/messages

# The authpriv file has restricted access.
Authpriv.*                                      /var/log/secure

# Log all the mail messages in one place.
Mail.*                                          -/var/log/maillog

# Log cron stuff
cron.*                                          /var/log/cron

# Everybody gets emergency messages
*.emerg                                         *

# Save news errors of level crit and higher in a special file.
```

```
uucp,news.crit                                  /var/log/spooler

# Save boot messages also to boot.log
local7.*                                        /var/log/boot.log
```

Resta da configurare la parte riguardante l'inoltro dei log al server centrale che è del tutto simile alla configurazione che abbiamo già visto lato server per l'invio dei messaggi al database:

```
## Filter duplicated messages
$RepeatedMsgReduction on

# An on-disk queue is created for this action. If the remote host is
# down, messages are spooled to disk and sent when it is up again.
$WorkDirectory /var/spool/rsyslog  # default location for work (spool) files
$ActionQueueType LinkedList   # use asynchronous processing
$ActionQueueFileName syslog1  # set file name, also enables disk mode
$ActionQueueMaxDiskSpace 1g
$ActionResumeRetryCount -1    # infinite retries on insert failure
$ActionQueueSaveOnShutdown on # save in-memory data if rsyslog shuts down

# remote host is: name/ip:port, e.g. 192.168.0.1:514, port optional
# # forward messages to the remote server {textquotedbl}myserv{textquotedbl} on
# # # port 2514
*.* :omrelp:141.250.xxx.xxx:20514

$ActionExecOnlyWhenPreviousIsSuspended on
& /var/log/rsyslog-fail/localbuffer
$ActionExecOnlyWhenPreviousIsSuspended off # to re-set it for the next selector
```

La prima direttiva serve a ridurre quello che è un problema noto di rsyslog se configurato, come in questo caso, per l'utilizzo con protocollo RELP, ovvero l'invio dello stesso messaggio per più volte. Si è ritenuto che la possibilità di avere dei messaggi duplicati sia decisamente migliore di quella di perderne alcuni visto quanto richiesto dal Garante riguardo la completezza e l'integrità dei log.

Le sei direttive successive sono del tutto identiche a quelle già viste nella configurazione lato server e servono nel caso il log server centrale non sia raggiungibile. In questo caso i messaggi vengono accodati e inviati non appena il log server tornerà disponibile. Nel caso il client dovesse essere stoppato i log vengono scritti su disco nella directory di spool specificata.

La direttiva di inoltro fa sì che tutti i log siano inviati tramite protocollo RELP all'IP indicato sulla porta specificata subito dopo l'IP.

Le ultime tre righe indicano un'azione che si attiva solo nel caso le precedenti dovessero fallire (es. errore di sintassi nel file di configurazione, riempimento della coda, etc.) e non sia quindi possibile inoltrare i log al server. Questo caso è molto raro ma, per rimanere più possibile aderenti a quanto richiesto dal Garante della Privacy, si è ritenuto opportuno considerarlo comunque. Dal momento che è difficile accorgersi quando questa azione viene attivata in tutti i client è inoltre presente uno script che controlla quotidianamente quando il file che viene generato ha una dimensione maggiore di 0. Se si presenta questa situazione lo script provvede ad avvertire via email. Vista l'eccezionale rarità di questa situazione il file generato dovrà poi essere copiato manualmente nel log server dopo che l'errore che ha portato all'attivazione di questa direttiva sia stato corretto.

3.2.2 Client Windows

L'installazione dei client Windows è semplice e veloce come tutti i software per Windows. Una volta accettata la licenza, selezionato il path e scelto il tipo di installazione la procedura terminerà in pochi istanti e potremo passare quindi alla configurazione. La configurazione è piuttosto banale, quello che dobbiamo fare è portarci nella sezione che gestisce l'inoltro dei messaggi ad un log server.

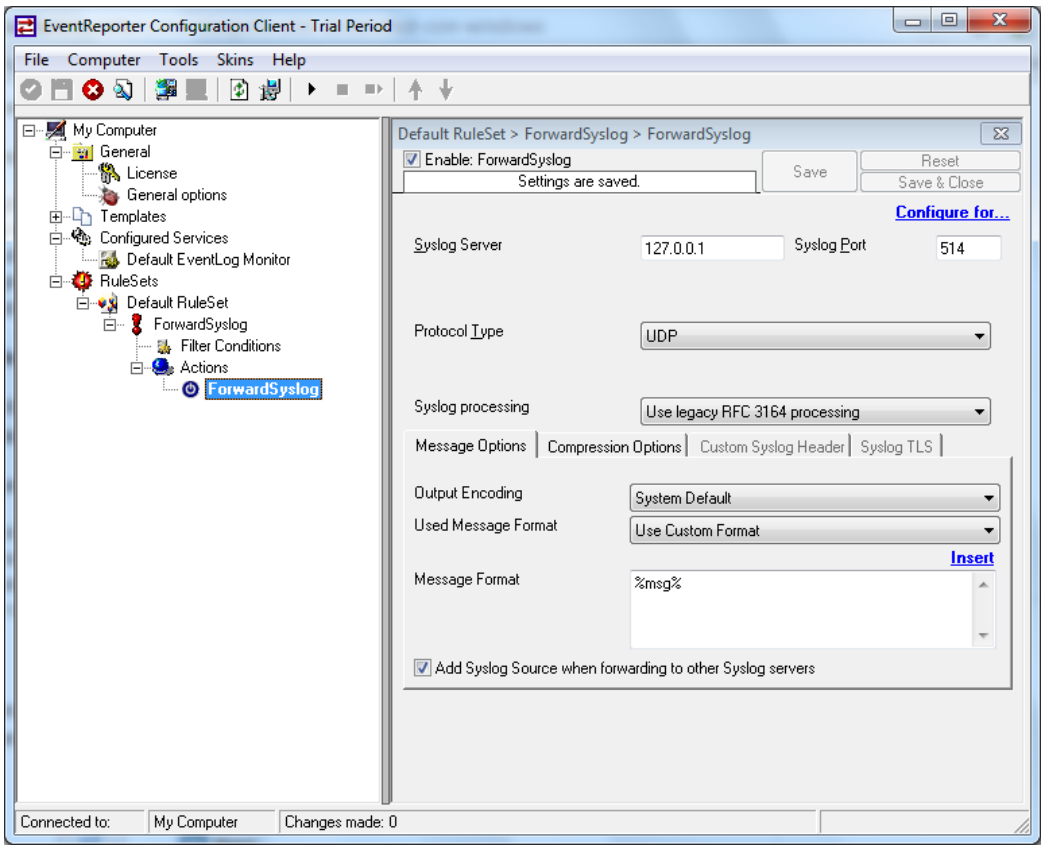

Figura 3.1: EventReporter configurazione di default per inoltro messaggi

Qui dobbiamo inserire l'indirizzo IP del nostro log server, selezionare TCP come tipo di protocollo e attivare l'opzione che permette di salvare su disco i messaggi che non sono stati consegnati a causa di malfunzionamento del server.

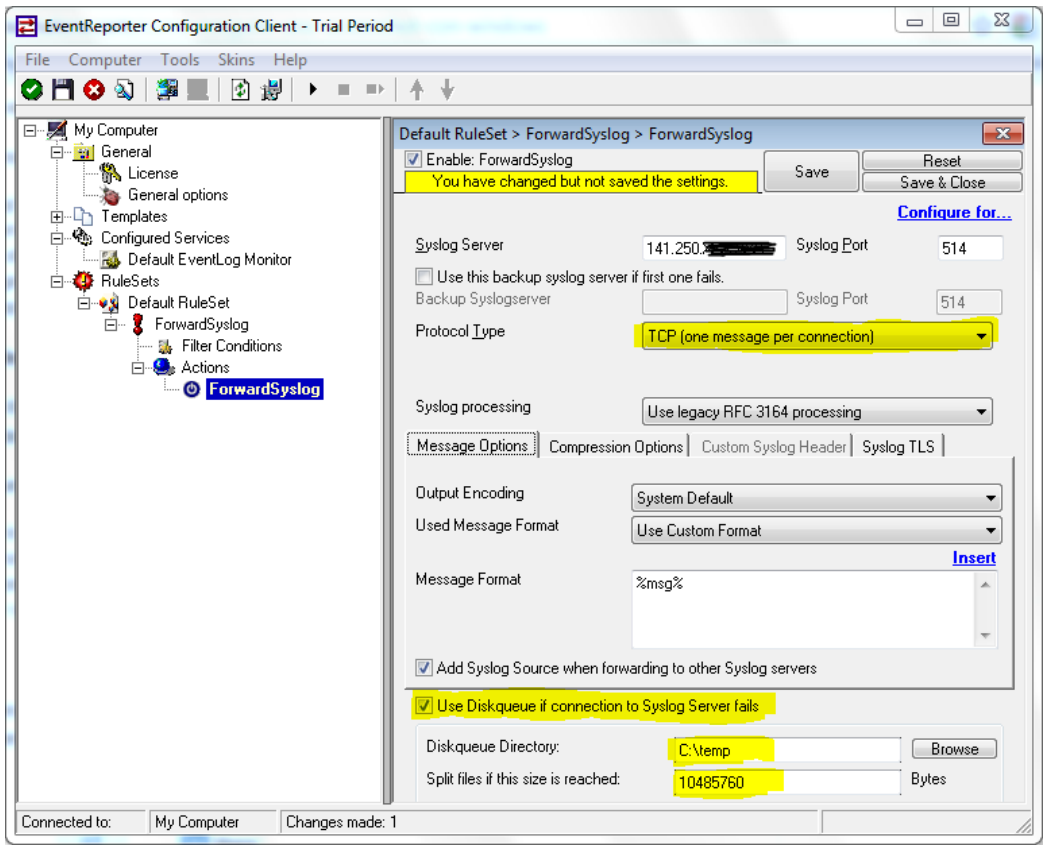

Figura 3.2: EventReporter configurazione per inolto messaggi

3.2.3 Raccolta dei log di IBM Tivoli Storage Manager, VMware vSphere e Oracke DB

Per completare la raccolta dei log che rientrano tra quelli richiesti dal Garante e in gestione all'Ufficio Sistemi Gestionali rimangono ancora quelli di IBM Tivoli Storage Manager, VMware vSphere e dei database Oracle.

Tivoli Storage Manager è il software che si occupa del backup di tutti i nostri server, sia Linux che Windows e dei nostri database Oracle.

VMware vSphere è la piattaforma di virtualizzazione su cui si appoggiano la maggioranza dei nostri server.

Oracle è il database usato da quasi tutti gli applicativi di ateneo (es. rilevamento presenze, contabilità, stipendi, segreterie studenti, etc.) e che quindi contiene la maggioranza dei dati in nostro possesso.

IBM Tivoli Storage Manager

Tivoli Storage Manager rientra sicuramente tra i software i cui log devono essere raccolti, ciò nonostante questo software non prevede nativamente la possibilità di inviare i propri log ne ad un sistema di logging remoto ne locale.

Il primo passo è stato quello di individuare i log generati da Tivoli Storage Manager e controllare se contenevano le informazioni minime richieste dalla normativa, ovvero login e logout. Tivoli Storage Manager dispone di due sistemi di gestione separati, uno da riga di comando e uno da interfaccia web i quali operano come entità separate e che quindi hanno ognuno i suoi file di log.

L'interfaccia a riga di comando di default non ha attiva la registrazione degli eventi su file quindi si è dovuto procedere a modificare la configurazione per fare in modo che tutte le attività venissero registrate[2]. Una volta effettuata la modifica i log sono scritti su un file di testo, resta quindi da passare questi log all'rsyslog locale che provvederà poi ad inviarli insieme a tutti gli altri al server centrale. Per fare in modo che i log generati siano passati a rsyslog si è sfruttato il comando logger che provvede proprio a questo lavoro. Il comando logger in combinazione al comando tail inserito all'interno dell'inittab ci permette di far iniziare l'operazione al boot della macchina e

[2] `http://publib.boulder.ibm.com/infocenter/tsminfo/v6r2/index.jsp?topic=/com.ibm.itsm.srv.ref.doc/r_cmd_events_enable.html&resultof="TEC"`

l'azione respawn ci assicura che il processo viene riavviato nel caso dovesse terminare inaspettatamente. L'opzione -F del comando tail fa in modo che questo sia legato al nome del file e non al suo descriptor, questo serve nel momento in cui il file viene ruotato, le opzioni -p e -t di logger servono invece a specificare facility e severity con cui passare i log a rsyslog e a specificare un tag con il quale i log vengono identificati. Questa seconda opzione ci tornerà utile nel server centrale in quanto i log, essendo suddivisi in base al nome del programma che li ha generati, verranno identificati separatamente grazie al tag applicato.

```
7:2345:respawn:/usr/bin/tail -F \
/var/log/tsm/FILETEXT/filetext.log \
| /usr/bin/logger -p local4.info -t tsm
```

Per l'interfaccia web i passi sono stati praticamente gli stessi, si è provveduto ad abilitare la registrazione su file degli eventi[3] e poi ad inserire nel file inittab una riga del tutto simile a quella usata per i log generati dall'interfaccia a riga di comando.

```
8:2345:respawn:/usr/bin/tail -F \
/opt/ibm/ac/profiles/TIPProfile/logs/server1/trace.log \
| /usr/bin/logger -p local4.info -t tip
```

Non tutti i sistemi Linux dispongono ancora dell'inittab in quanto alcuni (es. Ubuntu) sono passati dal vecchio demone System V al nuovo Upstart, la configurazione sarà leggermente diversa ma il comando da processare sarà sempre lo stesso.

VMware vSphere

Come detto precedentemente la nostra architettura di virtualizzazione è basata sulla tecnologia VMware vSphere, il nostro scenario inoltre comprende

[3] `http://www-01.ibm.com/support/docview.wss?uid=swg21501110`

sia una versione Enterprise del prodotto, sia una versione Free (gratuita ma non open-source). Partiamo dallo scenario più semplice, il singolo host che ha il compito di ospitare le macchine virtuali preposte al salvataggio dei dati per un eventuale ripristino a seguito di un evento disastroso.

VMware ESXi, a differenza della versione Enterprise, è molto semplice da configurare in quanto ha la possibilità di poter inviare nativamente i log ad un server remoto. L'attivazione di questa funzionalità si trova nelle opzioni di configurazione avanzata dove sono presenti le voci Syslog.Remote.Hostname e Syslog.Remote.Port[4]. Impostate queste due opzioni l'inoltro dei log è attivo.

Nella versione Enterprise la soluzione è diversa e suddivisa in due parti, una riguardante le macchine del cluster e una riguardante la parte gestionale vCenter Server.

Per quanto riguarda le macchine del cluster la soluzione è ben documentata[5] e piuttosto veloce da attivare, anche se diversa dalla versione gratuita. Quello che è necessario fare è collegarsi via ssh e modificare il file /etc/syslog.conf inserendo la direttiva per l'invio:

```
*.* @141.250.xxx.xxx
```

Una volta modificato il file è necessario riavviare il servizio e verificare che il firewall integrato abbia abilitata la regola per la connessione al server syslog remoto:

```
esxcfg-firewall -q|grep syslog
```

Nel caso fosse necessario abilitare la connessione nel firewall il comando da eseguire è il seguente:

[4] http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1016621

[5] http://kb.vmware.com/selfservice/microsites/search.do?cmd=displayKC&docType=kc&docTypeID=DT_KB_1_1&externalId=1005030

```
esxcfg-firewall -o 514,udp,out,syslog && esxcfg-firewall -l
```

Come si vede dalla regola di abilitazione del firewall queste macchine inviano i propri log con protocollo UDP. Questo è infatti uno dei casi in cui il dispositivo non supporta il protocollo TCP ed è in grado di usare solamente il protocollo UDP.

Resta infine da raccogliere i log della parte gestionale ovvero del vCenter Server.

Questa parte è risultata essere più complessa di quelle precedentemente viste in quanto questo software non prevede in nessun modo l'invio dei log generati ad un sistema di logging come invece è possibile fare con gli hypervisior ESX ed ESXi. Gli eventi generati dal vCenter Server sono raccolti in dei file di testo contenuti nella directory "C:\Documents and Settings\All Users\Application Data\VMware\VMware VirtualCenter". A prima vista la soluzione poteva sembrare del tutto simile a quella già adottata per l'inoltro dei log del TSM, ovvero utilizzare un software equivalente al logger di linux in grado di leggere dal file e capace di inviare i messaggi al server centrale. Questa soluzione si è rivelata subito non applicabile nello stesso modo che abbiamo visto precedentemente in quanto i log non vengono scritti in un file il cui nome rimane invariato ma in un file che ha nel suo nome un numero che cresce progressivamente al ruotare del file. Purtroppo a oggi non è possibile cambiare questo meccanismo in quanto hardcoded nel software. Si è dovuto quindi cercare una strada diversa per riuscire ad estrapolare i messaggi prodotti. Esaminando i comandi messi a disposizione dalla PowerCLI di VMware, ovvero dall'interfaccia a riga di comando, è possibile riuscire ad estrapolare le informazioni che cerchiamo. Quello che è stato quindi necessario fare è uno script in PowerShell[6] che attraverso i comandi della PowerCLI

[6]Si veda Appendice A pag. 55 per lo script completo

fosse in grado di estrapolare i log prodotti e di inserirli in un file ben preciso che potesse a questo punto essere esaminato da un logger.

Tra i software che svolgono questo lavoro si è anche questa volta optato per un prodotto Adiscon, sia per quanto detto precedentemente, per offrire anche un supporto economico ad un'azienda che è impegnata nello sviluppo di software free, sia per le caratteristiche offerte. MonitorWare Agent[7] è il programma che abbiamo quindi adottato e che offre tra le caratteristiche oltre alla funzione di monitoraggio di file di testo anche la possibilità come EventReporter e rsyslog di mantenere una coda dei messaggi che non è stato possibile inviare al log server. Lo script dovrà quindi essere schedulato per essere eseguito ogni pochi minuti in modo tale che i messaggi possano essere letti dall'agent e inviati tempestivamente al log server.

Oracle DB

Nella maggioranza dei casi gli applicativi impiegati dall'Università di Perugia e gestiti dall'Ufficio Sistemi Gestionali fanno uso di database Oracle.

Oracle, come già fatto per gli altri software precedenti, deve essere opportunamente configurato per fare in modo che i proprio log siano inviati ad un sistema di logging. Dalla versione 10 di Oracle esiste la possibilità di attivare "l'AUDIT_TRAIL" ovvero la traccia delle attività effettuate dagli utenti del database[8]. Tra le varie possibilità c'è anche quella di inviare questa traccia al syslog del sistema operativo specificando facility e severity con cui i messaggi saranno processati. I comandi devono essere eseguiti dall'utente system di Oracle e sono i seguenti:

```
$ sqlplus system
```

[7] `http://www.mwagent.com`

[8] Per Oracle 11: `http://docs.oracle.com/cd/B28359_01/network.111/b28531/auditing.htm`

```
enter password:
SQL> ALTER SYSTEM SET audit_trail=OS SCOPE=SPFILE;
SQL> ALTER SYSTEM SET audit_syslog_level='local3.info' SCOPE=SPFILE;
SQL> ALTER SYSTEM SET audit_sys_operations=TRUE SCOPE=SPFILE;
SQL> AUDIT CREATE SESSION;
```

Il primo comando fa in modo che i messaggi generati da Oracle siano tutti passati anche al sistema operativo e siano quindi scritti su file e non su una tabella del database.

Il secondo comando serve per inviare i log al syslog del sistema operativo con la facility e la severity indicate. Questo comando è utilizzabile solo nel caso in cui il sistema operativo su cui è stato installato Oracle sia un sistema Unix o Linux e, nel nostro caso, tutti i database Oracle sono installati su macchine Linux.

Il terzo comando serve a fare in modo che tutte le operazioni eseguite da utenti con privilegi da SYSDBA siano registrate ed infine l'ultimo fa tenere traccia di tutte le sessioni create da tutti gli utenti, ovvero di tutti i login e logoff.

Quello che resta da fare a questo punto è modificare adeguatamente il file di configurazione dell'rsyslog locale perché possa gestire i messaggi che arrivano da Oracle in maniera del tutto simile a quanto fatto precedentemente per il TSM. La prima modifica riguarda la direttiva che scrive i log in arrivo nel file messages che va cambiata dall'originale:

```
.info;mail.none;authpriv.none;cron.none                /var/log/messages
```

in:

```
.info;mail.none;authpriv.none;cron.none;local3.none                /var/log/messages
```

aggiungendo come si vede l'esclusione di tutti i log con facility local3 che è appunto quella scelta per i messaggi Oracle. Occorre infine aggiungere,

anche in questo caso, una direttiva che indichi che i log ricevuti con facility local3 devono essere scritti su un file specifico:

```
# Oracle audit
local3.*                        /var/log/oracle.log
```

Per rendere le modiche operative è necessario riavviare sia il database che il servizio rsyslog.

Come già fatto per il file di log del TSM anche per Oracle sarà indispensabile programmare una rotazione del file. La regola sarà del tutto uguale a quella già vista con la differenze del path del file che indicherà quello con i log di Oracle:

```
/var/log/oracle/oracle.log {
    daily
    compress
    rotate 28
    sharedscripts
    postrotate
        /bin/kill -HUP `cat /var/run/rsyslogd.pid 2> /dev/null` 2> /dev/null || true
    endscript
}
```

3.3 Lo script di masterizzazione

La masterizzazione è la soluzione che si è scelto di adottare per adempiere alla normativa per quello che riguarda l'inalterabilità dei log, mentre l'hasing dei singoli file è quello che ci garantirà la loro integrità.

Queste operazioni sono svolte quotidianamente da uno script bash[9] che viene eseguito immediatamente dopo la mezzanotte. Onde evitare di dover tutti i giorni sostituire il supporto scritto con uno vergine la soluzione prevede la masterizzazione su uno stesso media per più giorni (28 di default). Questo,

[9]Si veda in Appendice A pag. 59 per lo script completo

oltre a ridurre il quantitativo di supporti utilizzati, ci permette una buona autonomia anche nel caso non vi fosse nessuno in grado di poter sostituire il media per qualche giorno (es. nei week end).

Lo script è diviso in 3 parti, la prima dove vengono inizializzate le variabili usate, la seconda dove sono definite le funzioni e infine la parte di esecuzione vera e propria.

Nella prima parte sono definite tutte le variabili (destinatari delle email, comandi, file di log, etc.) che saranno poi utilizzate nella parte seguente del codice. La definizione delle variabili all'inizio dello script rende facile poter adattare lo script al proprio ambiente senza dover mettere mano alla parte successiva del codice.

```
#================================================
# Inizializzazione Variabili
#================================================
DESTINATION_MD5_MAIL_ADDRESS=youremail@example.com
DESTINATION_ERROR_MAIL_ADDRESS=youremail@example.com
SOURCE_MAIL_ADDRESS=root@‘hostname‘
REPLY_TO_MAIL_ADDRESS=youremail@example.com

# Comandi
CAT_COMMAND=/bin/cat
CDRDAO_COMMAND=/usr/bin/cdrdao
CDRECORD_COMMAND=/usr/bin/wodim
DATE_COMMAND=/bin/date
ECHO_COMMAND=/bin/echo
GAWK_COMMAND=/bin/gawk
GREP_COMMAND=/bin/grep
GZIP_COMMAND=/bin/gzip
LS_COMMAND=/bin/ls
MAIL_COMMAND=/usr/sbin/sendmail
MD5_COMMAND=/usr/bin/md5sum
MOUNT_COMMAND=/bin/mount
MKISO_COMMAND=/usr/bin/mkisofs
SED_COMMAND=/bin/sed
TAIL_COMMAND=/usr/bin/tail
```

```
UMOUNT_COMMAND=/bin/umount

RESCAN_SCRIPT=/usr/local/bin/rescan.bash

DATE=‘$DATE_COMMAND‘
SESSIONS_LIMIT=28
YESTERDAY_DATE=‘$DATE_COMMAND -d yesterday +%F‘

ISO_PATH="/tmp"
LOG_PATH="/var/log/remote-log"

ERROR_LOG_FILE="write_log_to_cd_error.log"
LOG_FILE="write_log_to_cd.log"
MD5_FILE="md5_check.txt"
TEMP_MESSAGE=$ISO_PATH/message_log_to_cd.tmp

NO="no"
CLOSE="false"
CUSTUM_DATE="false"
LOG_DATE=$YESTERDAY_DATE
```

La parte successiva è quella di definizione delle funzioni che saranno poi richiamate nella parte finale. Le funzioni sono: Usage, Rescan, Mail, MD5, MakeISO, MakeCD, Check_MD5, Startlog, Starterrorlog. Endlog.

Vediamole nel dettaglio una ad una.

Usage è la funzione che restituisce un messaggio di testo con le opzioni che si possono passare allo script. Il messaggio con le opzioni valide per questo script è:

```
Usage: $0 options

OPTIONS:
   -h      Mostra questo messaggio
   -s      Numero sessioni limite
   -d      Masterizza i log della data specificata (formato: aaaa-mm-gg)
   -m      Genera file md5 dei log della data specificata (formato: aaaa-mm-gg)
   -p      Specifica il path dove scrivere il file md5
   -C      Esegue il check md5 sui file di una data specifica (formato: aaaa-mm-gg)
   -c      Chiude il cd
```

La funzione Rescan richiama uno script esterno[10] che si occupa di riesaminare i canali scsi della macchina per identificare quale sia il dispositivo di masterizzazione installato. Il valore restituito dallo script viene salvato in una variabile che servirà in fase di scrittura dei log su CD. La funzione Mail viene richiamata tutte le volte è necessario inviare una mail di errore o di avviso agli AdS.

```
Mail() {
        # $1 Subject della mail
        # $2 Header del testo della mail
        # $3 Testo della mail
        # Invia avviso via mail e termina
        $ECHO_COMMAND "$3" > $TEMP_MESSAGE
        $SED_COMMAND -i "1i$2\n" $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
        $SED_COMMAND -i "1iTo: $DESTINATION_ERROR_MAIL_ADDRESS\n\
From: $SOURCE_MAIL_ADDRESS\n\
Reply-To: $REPLY_TO_MAIL_ADDRESS\n\
Subject: `hostname`: $1" $TEMP_MESSAGE \
        2>> $LOG_PATH/$ERROR_LOG_FILE
        $MAIL_COMMAND -t < $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
        rm -f $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
}
```

La funzione si aspetta in input tre parametri, l'oggetto della mail, una intestazione al testo e il testo della mail. Il testo viene salvato in un file temporaneo al quale poi viene aggiunta l'intestazione del testo e gli headers della mail. Il file è quindi pronto per essere passato al comando mail che provvederà ad inoltrare la mail al server di posta. Una volta che il messaggio è stato inviato il file contenente il testo della mail viene cancellato.

La funzione MD5 si occupa di comprimere e poi calcolare l'hash dei file generati, prende in input due parametri: una data, che identifica la directory sulla quale si vuole agire, e facoltativamente il path dove vogliamo sia salvato il file contenente la lista dei file con i rispettivi hash associati.

[10] Si veda Appendice A pag. 75 per il codice dello script

```
MD5() {
    # $1 Nome della directory (data)
    # $2 Se esiste specifica un path diverso dal default
    # Genera l'MD5 dei file nella directory $1 e lo scrive nel file $MD5_FILE
    DIR=‘$LS_COMMAND $LOG_PATH/$1‘ 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
    for d in $DIR
    do
        $ECHO_COMMAND "Compressione file directory $1/$d" >> $LOG_PATH/$LOG_FILE
        FILE=‘ls $LOG_PATH/$1/$d‘ 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        for f in $FILE
        do
            $ECHO_COMMAND "$f" | $GREP_COMMAND -qE ".gz"
            if [ $? -ne 0 ]; then
                # Compressione file
                $ECHO_COMMAND "$GZIP_COMMAND $LOG_PATH/$1/$d/$f" >> $LOG_PATH/$LOG_FILE

                $GZIP_COMMAND $LOG_PATH/$1/$d/$f \
                1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

                if [ $? -eq 0 ]; then
                    # MD5 del file
                    $ECHO_COMMAND "$MD5_COMMAND $LOG_PATH/$1/$d/$f.gz" \
                    >> $LOG_PATH/$LOG_FILE

                    cd $LOG_PATH

                    $MD5_COMMAND $1/$d/$f.gz \
                    1>> $LOG_PATH/$1/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                fi
            else
                $ECHO_COMMAND "File $f gia' compresso" >> $LOG_PATH/$LOG_FILE

                # MD5 del file
                $ECHO_COMMAND "$MD5_COMMAND $LOG_PATH/$1/$d/$f" >> $LOG_PATH/$LOG_FILE

                cd $LOG_PATH
                if [[ -z $2 ]]; then
                    $MD5_COMMAND $1/$d/$f \
                    1>> $LOG_PATH/$1/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                else
                    $MD5_COMMAND $1/$d/$f \
```

```
                    1>> $2/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                fi
            fi
        done
    done
    if [ $? -ne 0 ]; then
        # Errore generazione MD5
        # Invia avviso via mail e termina
        SUBJECT="Errore generazione MD5"
        HEADER="Errore generazione MD5: controllare il log"
        BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
    $ECHO_COMMAND "Exit MD5 function" >> $LOG_PATH/$LOG_FILE
}
```

La funzione scorre tutte le sottodirectory della data specificata e verifica se i file al loro interno sono già stati compressi o meno; nel caso i file risultassero non compressi si procede alla compressione tramite il comando gzip. Del file compresso viene quindi calcolato l'hash MD5 e il valore ottenuto viene scritto in un file contenente la lista di tutti i file esaminati e dei rispettivi hash. Se non indicato diversamente, il file, contenente gli hash, viene scritto all'interno della directory indicante la data passata come primo argomento alla funzione. Questo fa si che in fase di masterizzazione venga incluso nel CD anche questo file che potrà tornare utile per verificare l'integrità dei file di log.

La funzione MakeISO si occupa di generare il file immagine che sarà poi masterizzato nel CD.

```
MakeISO() {
    # $1 Data dei log
    # $2 Tipo di iso da generare
    if [[ -z $2 ]]; then
        $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso
```

```
                -M $CDRW_DEV -C $($CDRECORD_COMMAND
                -s dev=$CDRW_DEV -msinfo) -root $1 $LOG_PATH/$1" \
    1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

    $MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -M $CDRW_DEV -C $($CDRECORD_COMMAND \
    -s dev=$CDRW_DEV -msinfo) -root $1 $LOG_PATH/$1 1>> $LOG_PATH/$LOG_FILE \
    2>> $LOG_PATH/$ERROR_LOG_FILE

    if [ $? -ne 0 ]; then
        # Errore nel generare la iso
        # Invia avviso via mail e termina
        SUBJECT="Errore generazione iso CD"
        HEADER="Errore nella procedura di generazione della iso: controllare il log"
        BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
else
    if [ $2 == "empty" ]; then
        $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/empty.iso -M $CDRW_DEV \
        -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo) $ISO_PATH/empty" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $MKISO_COMMAND -R -l -o $ISO_PATH/empty.iso -M $CDRW_DEV \
        -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo) $ISO_PATH/empty \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        if [ $? -ne 0 ]; then
            # Errore nel generare la iso
            # Invia avviso via mail e termina
            SUBJECT="Errore generazione iso vuota"
            HEADER="Errore nella procedura di generazione della iso: controllare il log"
            BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
            Mail "$SUBJECT" "$HEADER" "$BODY"
            Endlog
            exit 1
        fi
    elif [ $2 == "first" ]; then
        $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -root $1 $LOG_PATH/$1" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
```

```
            $MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -root $1 $LOG_PATH/$1 \
            1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            if [ $? -ne 0 ]; then
                # Errore nel generare la iso
                # Invia avviso via mail e termina
                SUBJECT="Errore generazione iso CD"
                HEADER="Errore nella procedura di generazione della iso: controllare il log"
                BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
                Mail "$SUBJECT" "$HEADER" "$BODY"
                Endlog
                exit 1
            fi
        fi
    fi
}
```

I parametri della funzione sono anche in questo caso due: il primo che identifica la data dei log da masterizzare e il secondo che determina il tipo di immagine da generare. Dal momento che il CD sarà suddivisoi in sessioni, per permettere che possa essere utilizzato per più giorni, l'immagine dovrà tenere conto della sua "posizione" nel CD. Le opzioni per generare il file immagine variano infatti a seconda che l'immagine sia la prima ad essere scritta o una delle successive. Un terzo caso è quello di dover masterizzare un'immagine vuota per poter chiudere il CD. Il secondo parametro serve quindi a identificare in quale dei tre casi ci troviamo.

La funzione MakeCD è quella che si occupa della scrittura vera e propria su CD.

```
MakeCD() {
    # $1 Nome della iso da masterizzare
    # $2 Specifica come chiudere il CD
    if [[ -z $2 ]]; then
        $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -multi -data -v $ISO_PATH/$1.iso" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
```

```
    $CDRECORD_COMMAND dev=$CDRW_DEV -multi -data -v $ISO_PATH/$1.iso \
    1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

    if [ $? -ne 0 ]; then
        SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
        HEADER="Errore nella procedura di masterizzazione della iso su CD:
                controllare il log"
        BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
    rm -f $ISO_PATH/$1.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
else
    if [ $2 == "empty" ]; then
        $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/empty.iso" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/empty.iso \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        if [ $? -ne 0 ]; then
            SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
            HEADER="Errore nella procedura di masterizzazione della iso su CD:
                    controllare il log"
            BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
            Mail "$SUBJECT" "$HEADER" "$BODY"
            Endlog
            exit 1
        fi
        rm -f $ISO_PATH/empty.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
    elif [ $2 == "close" ]; then
        $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/$1.iso" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/$1.iso \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        if [ $? -ne 0 ]; then
            SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
```

```
                HEADER="Errore nella procedura di masterizzazione della iso su CD:
                        controllare il log"
                BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
                Mail "$SUBJECT" "$HEADER" "$BODY"
                Endlog
                exit 1
            fi
            rm -f $ISO_PATH/$1.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        fi
    fi
}
```

Il suo compito è prendere il file immagine precedentemente generato e aggiungere una sessione contenente i dati in esso contenuti. Il file immagine è quindi un parametro necessario alla funzione mentre il secondo parametro facoltativo ci serve per identificare il fatto che il CD deve essere chiuso. Il CD può essere chiuso in due modi: raggiungendo l'ultima sessione disponibile oppure tramite l'opzione -c da riga di comando. Il secondo modo ci torna utile nel caso dovessimo sostituire per qualche motivo il supporto prima di aver raggiunto l'ultima sessione. Al termine del processo di scrittura il file immagine, che viene scritto dalla precedente funzione sul filesystem /tmp, viene cancellato.

Check_MD5 è la funzione che controlla l'hash MD5 dei file masterizzati su CD in una determinata data.

```
Check_MD5() {
    # $1 Nome della directory (data
    Rescan
    $MOUNT_COMMAND $CDRW_DEV /mnt 2>> $LOG_PATH/$ERROR_LOG_FILE
    if [ $? -ne 0 ]; then
        SUBJECT=‘hostname‘": Errore mount CD per check MD5"
        HEADER="Errore nella procedura di check degli MD5:
                impossibile montare il CD, controllare il log"
        BODY=$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
```

```
        exit 1
    fi
    cd /mnt
    MD5_ERROR=‘$MD5_COMMAND -c $1/md5_check.txt 2>> $LOG_PATH/$ERROR_LOG_FILE |\
    $GREP_COMMAND -v "OK$" 2>> $LOG_PATH/$ERROR_LOG_FILE‘

    if [[ ! -z $MD5_ERROR ]]; then
        SUBJECT=‘hostname‘": Errore check MD5"
        HEADER="Errore nella procedura di check degli MD5: controllare il log"
        BODY=‘$ECHO_COMMAND "$MD5_ERROR"‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
    fi
    cd
    $UMOUNT_COMMAND /mnt
    if [ $? -ne 0 ]; then
        SUBJECT=‘hostname‘": Errore umount CD per check MD5"
        HEADER="Errore nella procedura di check degli MD5:
                impossibile smontare il CD, controllare il log"
        BODY=$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
}
```

Quello che deve fare è montare il supporto ed eseguire la verifica di integrità dei file presenti nella directory identificata dalla data che deve essere passata come parametro alla funzione. Se ci fossero delle incongruenze tra l'hash calcolato al momento della masterizzazione e quello calcolato al momento della verifica una mail di avviso viene inviata agli AdS. In questo caso il supporto non potrà essere considerato attendibile e dovrà essere rimasterizzato. Un controllo completo del CD dovrà essere fatto ogni qualvolta questo viene chiuso e sostituito.

Le ultime funzioni sono: Startlog , Starterrorlog e Endlog. Le tre le funzioni servono rispettivamente a scrivere la date e l'ora di inizio nel file di

log dello script, la date e l'ora di inizio nel file degli errori dello script e la date e l'ora in cui termina lo script.

Da come abbiamo visto nella funzione Usage lo script prevede di poter essere eseguito con delle opzioni che ne modificano il comportamento. Questi argomenti vengono controllati prima di iniziare qualsiasi altra operazione proprio perché indicano le azioni da intraprendere. Vediamo nello specifico qual'è il comportamento dello script a seconda delle opzioni che vengono passate quando questo viene eseguito.

L'opzione -h restituisce l'help dello script che elenca tutte le opzioni e la loro funzionalità.

L'opzione -s serve ad impostare un limite diverso di sessioni rispetto al default.

L'opzione -d serve a indicare una data del quale vogliamo masterizzare i log su CD

L'opzione -m genera il file con la lista degli hash MD5 di una data specifica

L'opzione -p può essere utilizzata insieme all'opzione -m per indicare il path dove vogliamo sia salvato il file generato.

L'opzione -C esegue la verifica degli hash MD5 sui file della data indicata.

L'opzione -c, infine, occorre a chiudere il CD e a fare quindi in modo che non vi possano essere aggiunti altri dati.

Se allo script non vengono passate opzioni eseguirà la masterizzazione dei log generati il giorno precedente.

Il codice seguente l'inizializzazione delle eventuali opzioni si occupa di effettuare alcune operazioni preliminari che controllano se il supporto inserito è vuoto oppure è già in parte scritto e in questo secondo caso calcola lo spazio rimanente. Per fare in modo che tutte le informazioni del disco, come per esempio le sessioni presenti, siano correttamente riconosciute è necessario

prima di effettuare le operazioni preliminari eseguire la funzione Rescan che oltre a identificare il device ottico ne forza la rilettura.

```
#================================================
# Operazioni preliminari e CD info
#================================================

Rescan
sleep 5

# Variabili
EMPTY_MEDIA=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GREP_COMMAND empty 2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GAWK_COMMAND ’{print($4)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

APPENDABLE_MEDIA=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GREP_COMMAND Appendable 2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GAWK_COMMAND ’{print($3)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

if [ -z $EMPTY_MEDIA ]; then
    SUBJECT="Errore recupero informazioni CD"
    HEADER="Errore nel recupero delle informazione del CD: controllare il log"
    BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
    Mail "$SUBJECT" "$HEADER" "$BODY"
    Endlog
    exit 1
fi

TOTAL_MEDIA_SPACE=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GREP_COMMAND Total 2>> $LOG_PATH/$ERROR_LOG_FILE |\
$GAWK_COMMAND ’{print($7)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

TOTAL_MEDIA_SPACE=${TOTAL_MEDIA_SPACE%/*
```

A questo punto possiamo procedere con la compressione dei file e la generazione dell’hash associato.

```
#================================================
```

```
# Compressione log e md5
#=============================================

$ECHO_COMMAND "MD5 del: $LOG_DATE" > $LOG_PATH/$LOG_DATE/$MD5_FILE
$ECHO_COMMAND "" >> $LOG_PATH/$LOG_DATE/$MD5_FILE

$ECHO_COMMAND "-------------------------------" \
1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

MD5 "$LOG_DATE"

$ECHO_COMMAND "-------------------------------" \
1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

$SED_COMMAND -i "1iTo: $DESTINATION_MD5_MAIL_ADDRESS\n\
From: $SOURCE_MAIL_ADDRESS\n\
Reply-To: $REPLY_TO_MAIL_ADDRESS\n\
Subject: `hostname`: MD5 check del $LOG_DATE" \
$LOG_PATH/$LOG_DATE/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

# Invio lista MD5
$MAIL_COMMAND -t < $LOG_PATH/$LOG_DATE/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
```

L'operazione di compressione e di hashing è eseguita dalla funzione MD5 che abbiamo già visto. Quello che avviene a questo punto è che una volta generato il file con gli hash questo viene inviato via email all'indirizzo email indicato nelle variabili iniziali dello script. Subito a seguire inizia la parte di generazione della iso e di masterizzazione. Questa è l'ultima parte dello script, ma prima di poter richiamare la funzione per la creazione del file immagine, si deve procedere all'esame di alcune condizioni. In base ad alcuni fattori infatti viene determinato il tipo di immagine da generare e quindi quali parametri passare alla funzione MakeISO. Per prima cosa quindi viene controllato se il supporto inserito è vuoto o meno, nel caso in cui non sia vuoto devo essere sicuro che sia ancora scrivibile; se il CD risultasse essere chiuso una mail sarà inviata per avvisare che è indispensabile sostituire il

media. Se il CD è scrivibile o vuoto il passo successivo è l'esecuzione della funzione MakeISO che provvederà a scrivere il file immagine nel filesystem /tmp. In entrambi i casi la verifica dello spazio rimanente nel supporto è necessaria in quanto questo potrebbe non essere sufficiente per l'immagine generata. Se si dovesse verificare questa condizione una mail di avviso verrebbe inviata agli AdS. Questi dovrebbero provvedere o alla sostituzione del supporto con uno di dimensioni maggiori, nel caso il supporto inserito sia vergine (es. sostituzione del CD con un DVD), oppure con la chiusura del supporto parzialmente usato e con il successivo inserimento di un supporto vuoto. Restano le ultime condizioni da provare prima di richiamare la funzione MakeCD che scriverà i log su CD: verificare l'eventuale raggiungimento del limite di sessioni (che può essere il default oppure un valore specificato dall'opzione -s) o la richiesta di chiusura del CD con l'opzione -c. Nel caso in cui una delle due condizioni dovesse risultare vera il supporto al termine della masterizzazione verrà chiuso, mentre, se entrambe risultassero false, il CD continuerà ad essere scrivibile. In tutti i casi al termine della scrittura viene effettuato il controllo dei dati scritti tramite la funzione Check_MD5 che ci avvertirà se alcuni file appena scritti non dovessero avere lo stesso hash di quelli presenti su filesystem.

3.4 L'interfaccia web

Per rendere più semplice la consultazione dei log si è deciso di renderli disponibili attraverso una interfaccia web e come spiegato precedentemente Loganalyzer è lo strumento che si è scelto.

Loganalyzer è un software scritto in php in grado di accedere ai log scritti su database e che ne permette una facile consultazione oltre ad offrire stru-

menti di ricerca e reportistica. La sua installazione richiede quindi che il server abbia installato un web server e le librerie php. Dal momento che l'Ufficio Sistemi Gestionali aveva già in dotazione un server web con questa caratteristica (apache 2.2.15 e php 5.3.3) si è scelto di sfruttare questa macchina e di lasciare al log server il solo compito di raccogliere i dati. Loganalyzer necessita anche di un suo database per il salvataggio di alcune informazioni (utenti e password, report. etc.) che sarà creato sul log server visto che già dispone di MySQL.

L'installazione si effettua scompattando il pacchetto tar.gz, che si può prelevare direttamente dal sito, in una directory accessibile in lettura dal web server (es. /var/www). Resta da definire un virtual host per poter accedere al sito. La configurazione può variare in base al server web scelto, nel caso di apache un esempio di configurazione può essere questo:

```
<VirtualHost log.dominio.it:80>
    ServerAdmin sysadm@dominio.it
    ServerName log.dominio.it
    DocumentRoot /var/www/log.dominio.it
    AddDefaultCharset UTF-8

    Timeout 3600

    <Directory /var/www/log.dominio.it>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order deny,allow
        deny from all
        allow from xxx.xxx.xxx.xxx/32
    </Directory>

    ErrorLog logs/log.dominio.it-error_log
    CustomLog logs/log.dominio.it-access_log common
</VirtualHost>
```

Come si può vedere nella configurazione compare l'opzione Timeout im-

postata a 3600 sec; un'impostazione che si è rivelata necessaria in quanto le query effettuate su database di notevoli dimensioni impiegano parecchio tempo a restituire il risultato. Questo parametro può essere adattato in base anche alle performance delle proprie macchine aumentandolo o diminuendolo.

Per completare l'installazione ci si deve collegare al sito e inserire tramite interfaccia web le informazioni circa il collegamento dell'applicativo al database e la creazione dell'utente amministratore. Una volta terminata l'istallazione si dovrà creare gli utenti personali per chi sarà autorizzato alla consultazione dei log.

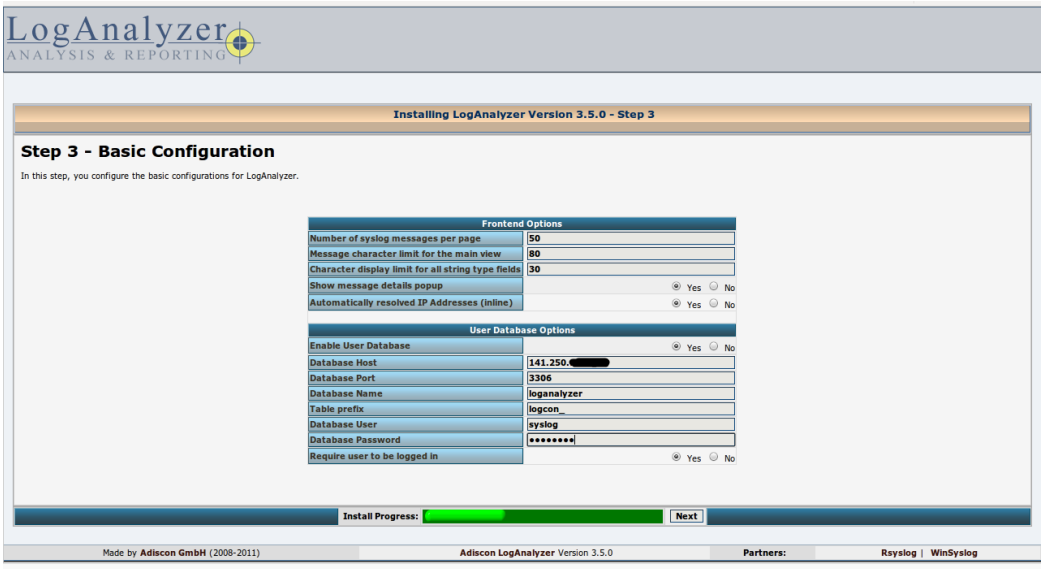

Figura 3.3: Parametri di connessione al database

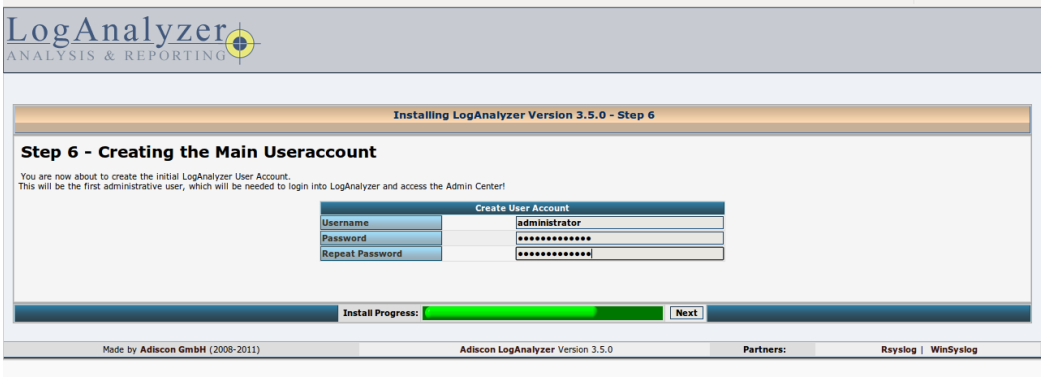

Figura 3.4: Creazione utente amministratore

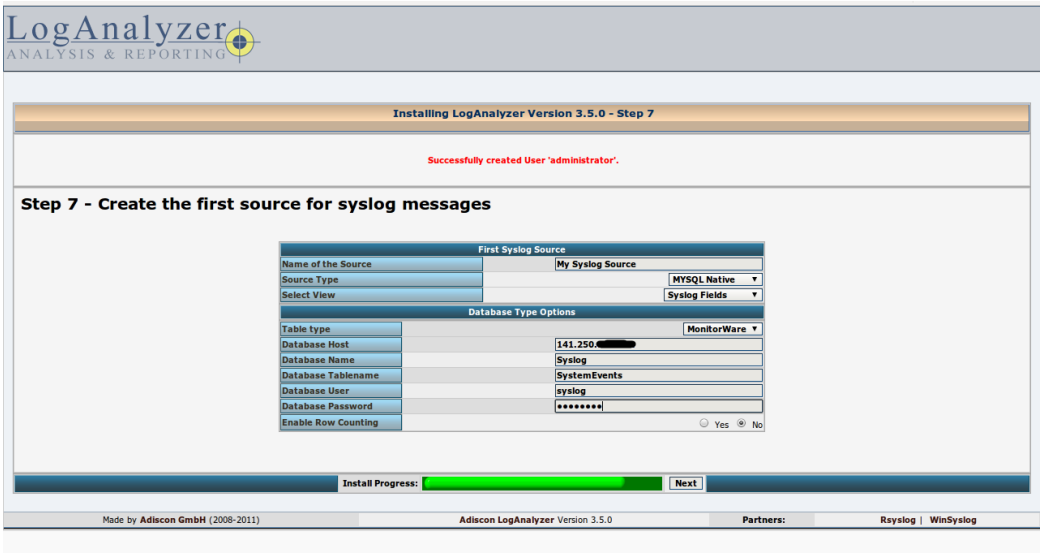

Figura 3.5: Parametri di connessione al database Syslog

3.5 Aspetti di sicurezza

Per terminare il progetto restano da esaminare alcuni aspetti di sicurezza che è opportuno valutare.

Il primo aspetto che abbiamo già incontrato è quello dei permessi dei file. Si è visto che alcuni file, come per esempio il file di configurazione di rsyslog sul log server, contengano al loro interno informazioni che non è opportuno siano visibili agli utenti. Bisognerà quindi porre attenzione affinché questi file abbiano dei permessi piuttosto restrittivi e che quindi l'accesso in lettura sia consentito solamente all'utente root.

Per aumentare ulteriormente la sicurezza si è scelto di attivare un firewall sulla macchina log server che permette la connessione solo alle porte e ai servizi indispensabili al funzionamento del sistema, ovvero le porte 514 per i protocolli UPD e TCP e la porta 20514 per il protocollo TCP che servono a rsyslog per ricevere i messaggi dagli altri server (la porta 20514 è quella

definita per il protocollo RELP), la porta 22 per il collegamento ssh per la manutenzione del server e infine la porta 3306 per il collegamento dell'interfaccia web al database MySQL. Il firewall permette il collegamento alle porte qui elencate solamente agli indirizzi IP che sono strettamente necessari. Nel caso specifico dell'ateneo di Perugia oltre al firewall presente sulla macchina sono presenti nella rete anche altri firewall a protezione dell'intera sala macchine.

Per evitare la perdita dei dati, quando un CD viene rimosso, deve essere consegnato al Responsabile o al Titolare del trattamento e deve essere riposto in un luogo sicuro e il cui accesso sia controllato. Un buon posto può essere una cassaforte ignifuga che è in grado di proteggere i dati oltre che da accessi non autorizzati anche da eventuali eventi disastrosi come per esempio incendi o terremoti. Come ultimo step di verifica, prima di consegnare il CD a chi di dovere, è necessario eseguire lo script[11] che verifica gli hash di tutti i file presenti sul CD.

Oltre alla copia su CD si è scelto di sfruttare la seconda sala macchine che usiamo come disaster recovery per il salvataggio dei log. Un dump dell'intero DB viene infatti effettuato e copiato sul sito remoto tutti i giorni. La copia viene effettuata su canale crittografato e il file è salvato tenendo conto di quanto detto precedentemente sui permessi dei file. Avere una copia dell'intero database ci permette, in caso di disastro, di poter ripristinare il sistema in brevissimo tempo.

Un altro aspetto che riguarda la sicurezza è quello di inibire per quanto possibile agli AdS l'accesso alla macchina contenente i log. Per fare in modo che gli AdS non possano accedere alla macchina una delle soluzioni potrebbe essere quella di far impostare la password di root al Titolare o al Responsabile

[11] Si veda Appendice A pag. 78 per il codice dello script

in modo che gli AdS non ne siano a conoscenza. Per evitare che avviando la macchina in single user mode gli AdS possano aggirare questa restrizione sarà necessario impostare anche una password nel bootloader che come per la password di root non dovrà essere conosciuta dagli AdS ma solo dal Titolare o dal Responsabile. Per permettere che gli AdS possano però compiere il loro lavoro è necessario dare ai loro utenti, tramite il comando sudo, dei permessi per poter eseguire alcune operazioni di ordinaria manutenzione, come per esempio il passaggio degli aggiornamenti di sistema.

Oltre a questo si deve cercare di impostare degli alert che possano avvisare il Titolare o il Responsabile di eventuali operazioni che possono in qualche modo compromettere le caratteristiche di integrità e completezza dei log.

Come ulteriore controllo, quindi, sul log server è in esecuzione uno script che avverte, via email, nel caso in cui la macchina dovesse essere spenta o riavviata[12], o nel caso il processo rsyslog non fosse in esecuzione[13].

[12] `http://www.syntaxtechnology.com/2009/06/email-on-shutdown-and-restart`

[13] Si veda Appendice A pag. 76 per il codice dello script

Conclusioni

Al termine del progetto quello che si può dire è che la normativa del Garante non è di facile attuazione se si vuole tenere conto più possibile di quello che viene richiesto. Le caratteristiche che si devono rispettare nel tenere traccia degli accessi logici degli AdS sono sicuramente impegnative da implementare. La legge, come abbiamo visto, per certi aspetti non è del tutto chiara e chiede che i log che si devono mantenere abbiano delle caratteristiche in alcuni casi quasi impossibili da adottare. Come tutti i sistemi informatici anche questo non è esente da difetti e potrà essere migliorato con il tempo, ad ogni modo si è cercato, anche con la collaborazione dei miei colleghi, di trovare tutti i possibili punti deboli sia della normativa che del sistema.

Le tre caratteristiche richieste dal Garante, inalterabilità, integrità e completezza, sono state tutte analizzate e si è cercato di trovare per ognuna una soluzione adeguata.

La più difficile da rispettare è senza dubbio l'inalterabilità. Rendere inalterabile un file è quasi impossibile se non dopo averlo scritto su un supporto non riscrivibile come si è poi scelto di fare seguendo anche quelle che sono le indicazioni del Garante nelle FAQ della normativa. Nel lasso di tempo che intercorre tra la masterizzazione di una sessione e un'altra i dati non possono però essere esenti dall'essere alterati. Sono vari, infatti, i modi di poter alterare i dati: modificando il contenuto dei file nel log server se in possesso dei

permessi di accesso, manomettendo l'hardware del dispositivo che raccoglie i log oppure alterando i messaggi prima che questi siano inviati al log server.

Per quello che riguarda la caratteristica di completezza si è cercato di ridondare più possibile i dati in possesso così da ridurre la possibilità di perdita di un dato. I log sono infatti presenti sul filesystem della macchina client e su quello del log server, sul database, di cui viene fatto il backup una volta al giorno sulla macchina di disaster recovery e infine su CD una volta masterizzati.

L'integrità è invece garantita dall'hash che viene generato e a sua volta scritto su supporto non riscrivibile oltre che inviato per email al Titolare o al Responsabile. L'hash ci permette di verificare che il dato sia stato scritto correttamente su CD e, a distanza di tempo, ci permette di accertare che anche il dato presente su filesystem non è cambiato dal momento in cui è stato generato l'hash.

Appendice A

Script elaborati

A.1 Script PowerShell per la raccolta dei log del vCenter Server

```
################################################
#
# Copyright (c) 2011 Gabriele Renzini
# Distributed under the GNU GPL v2.
# For full terms see: http://www.gnu.org/licenses/gpl.html GPL version 2.
#
# Ripartizione Servizi Informatici e Statistici
# University of Perugia
#
################################################


######################
# Script parameters #
######################
param($logfile="vcenter.log",$cachefile="cache_vcenter_log_script.txt",
            $scriptlogfile="vcenter_log_script.log",[switch]$help)

####################
# Script functions #
####################
```

```
function funHelp()
{
$helpText=@"

NAME: log_vcenter.ps1

PARAMETERS:
-logfile          where the logs are written        default:vcenter.log
-cachefile        where the script info are stored  default:cache_vcenter_log_script.txt
-scriptlogfile    where the script write his log    default:vcenter_log_script.log
-help             prints help

USAGE:
log_vcenter.ps1 [-logfile <string>][-cachefile <string>][-scriptlogfile <string>]

    Read from the vcenter log file and write the log to the specified file


log_vcenter.ps1 -help

    Displays this help

"@
$helpText
exit
}

############################
# Check script parameters #
############################
if($help){ "Obtaining help ..."; funHelp }

##########################################
# Connect to VMware VirtualCenter server #
##########################################
$vcserver="localhost"

Add-PSsnapin VMware.VimAutomation.Core
connect-VIServer $vcserver

#########################
```

```
# Start log acquisition #
#########################
# Check if file exists
if (test-path $scriptlogfile) { clear-content $scriptlogfile }
add-content $scriptlogfile "-------------------------------------------"
add-content $scriptlogfile ("Procedure started at :"+$(get-date))
add-content $scriptlogfile "-------------------------------------------"

# Check if file exists
if (test-path $cachefile)
{
    $prevlog = get-content $cachefile
    add-content $scriptlogfile ("First log is: "+$prevlog)
}
else
{
    add-content $scriptlogfile ("File "+$cachefile+" doesn't exists")
    add-content $cachefile $(Get-LogType)[0].key
    add-content $cachefile 1
    add-content $scriptlogfile "File created"
    $prevlog = get-content $cachefile
    add-content $scriptlogfile ("First log is: "+$prevlog[0])
}

for($i=0; $i -le $(Get-LogType).Length -1; $i++)
{
    if (!$(Get-LogType)[$i].Key.Contains('alert')
    -and !$(Get-LogType)[$i].Key.Contains('profile'))
    {
        $lastlog = $(Get-LogType)[$i]
    }
}
#$lastlog = $(Get-LogType)[11]
add-content $scriptlogfile ("Last log is: "+$lastlog.key)

$prevnum = [int]$prevlog[0].Substring(10,4)
$currnum = [int]$lastlog.Key.Substring(10,4)

if ($prevnum -ne $currnum)
{
    # Different file
```

```
    $log = get-log -Key $prevlog[0] -startlinenum $prevlog[1]
    $log.Entries > $logfile
    $prevnum = $prevnum+1
    while ($prevnum -ne $currnum)
    {
        $key = "vpxd:vpxd-"+$prevnum.ToString()+".log"
        $log = get-log -Key $key
        $log.Entries >> $logfile
        $prevnum = $prevnum+1
    }
    $lastlog.Key > $cachefile
    $log = get-log -Key $lastlog.Key
    $log.Entries >> $logfile
    $loglines = $log.Entries | Measure-Object
    $loglines.Count >> $cachefile
}
else
{
   # Same file
    $log = get-log -Key $prevlog[0] -startlinenum $prevlog[1]
    $log.Entries > $logfile
    $lastlog.Key > $cachefile
    $log = get-log -Key $lastlog.Key
    $loglines = $log.Entries | Measure-Object
    $loglines.Count >> $cachefile
}

add-content $scriptlogfile "-----------------------------------------------"
add-content $scriptlogfile ("Procedure ended at :"+$(get-date))
add-content $scriptlogfile "-----------------------------------------------"

########################
# End log acquisition #
########################
```

A.2 Script per la masterizzazione e la generazione dell'hash MD5

```
#!/bin/bash

################################################
#
# Copyright (c) 2011 Gabriele Renzini
# Distributed under the GNU GPL v2.
# For full terms see: http://www.gnu.org/licenses/gpl.html GPL version 2.
#
# Ripartizione Servizi Informatici e Statistici
# University of Perugia
#
################################################

#==============================================
# Inizializzazione Variabili
#==============================================
DESTINATION_MD5_MAIL_ADDRESS=youremail@example.com
DESTINATION_ERROR_MAIL_ADDRESS=youremail@example.com
SOURCE_MAIL_ADDRESS=root@‘hostname‘
REPLY_TO_MAIL_ADDRESS=youremail@example.com

# Comandi
CAT_COMMAND=/bin/cat
CDRDAO_COMMAND=/usr/bin/cdrdao
CDRECORD_COMMAND=/usr/bin/wodim
DATE_COMMAND=/bin/date
ECHO_COMMAND=/bin/echo
GAWK_COMMAND=/bin/gawk
GREP_COMMAND=/bin/grep
GZIP_COMMAND=/bin/gzip
LS_COMMAND=/bin/ls
MAIL_COMMAND=/usr/sbin/sendmail
MD5_COMMAND=/usr/bin/md5sum
MOUNT_COMMAND=/bin/mount
MKISO_COMMAND=/usr/bin/mkisofs
SED_COMMAND=/bin/sed
TAIL_COMMAND=/usr/bin/tail
```

```
UMOUNT_COMMAND=/bin/umount

RESCAN_SCRIPT=/usr/local/bin/rescan.bash

#CDRW_DEV=/dev/sr0

DATE=‘$DATE_COMMAND‘
SESSIONS_LIMIT=28
YESTERDAY_DATE=‘$DATE_COMMAND -d yesterday +%F‘

ISO_PATH="/tmp"
LOG_PATH="/var/log/remote-log"

ERROR_LOG_FILE="write_log_to_cd_error.log"
LOG_FILE="write_log_to_cd.log"
MD5_FILE="md5_check.txt"
TEMP_MESSAGE=$ISO_PATH/message_log_to_cd.tmp

NO="no"
CLOSE="false"
CUSTUM_DATE="false"
LOG_DATE=$YESTERDAY_DATE

#================================================
# Funzioni
#================================================

Usage() {
cat << EOF
Usage: $0 options

OPTIONS:
   -h      Mostra questo messaggio
   -s      Numero sessioni limite
   -d      Masterizza i log della data specificata (formato: aaaa-mm-gg)
   -m      Genera file md5 dei log della data specificata (formato: aaaa-mm-gg)
   -p      Specifica il path dove scrivere
   -C      Esegue il check md5 sui file di una data specifica (formato: aaaa-mm-gg)
   -c      Chiude il cd
EOF
}
```

```
Rescan() {
    # Rescan del canale scsi
    CDRW_DEV=‘$RESCAN_SCRIPT‘
}

Mail() {
    # $1 Subject della mail
    # $2 Header del testo della mail
    # $3 Testo della mail
    # Invia avviso via mail e termina
    $ECHO_COMMAND "$3" > $TEMP_MESSAGE
    $SED_COMMAND -i "1i$2\n" $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
    $SED_COMMAND -i "1iTo: $DESTINATION_ERROR_MAIL_ADDRESS\n\
From: $SOURCE_MAIL_ADDRESS\n\
Reply-To: $REPLY_TO_MAIL_ADDRESS\n\
Subject: ‘hostname‘: $1" $TEMP_MESSAGE \
    2>> $LOG_PATH/$ERROR_LOG_FILE
    $MAIL_COMMAND -t < $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
    rm -f $TEMP_MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
}

MD5() {
    # $1 Nome della directory (data)
    # $2 Se esiste specifica un path diverso dal default
    # Genera l'MD5 dei file nella directory $1 e lo scrive nel file $MD5_FILE
    DIR=‘$LS_COMMAND $LOG_PATH/$1‘ 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
    for d in $DIR
    do
        $ECHO_COMMAND "Compressione file directory $1/$d" >> $LOG_PATH/$LOG_FILE
        FILE=‘ls $LOG_PATH/$1/$d‘ 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        for f in $FILE
        do
            $ECHO_COMMAND "$f" | $GREP_COMMAND -qE ".gz"
            if [ $? -ne 0 ]; then
                # Compressione file
                $ECHO_COMMAND "$GZIP_COMMAND $LOG_PATH/$1/$d/$f" \
                >> $LOG_PATH/$LOG_FILE

                $GZIP_COMMAND $LOG_PATH/$1/$d/$f \
                1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
```

```
                if [ $? -eq 0 ]; then
                    # MD5 del file
                    $ECHO_COMMAND "$MD5_COMMAND $LOG_PATH/$1/$d/$f.gz" \
                    >> $LOG_PATH/$LOG_FILE

                    cd $LOG_PATH
                    $MD5_COMMAND $1/$d/$f.gz \
                    1>> $LOG_PATH/$1/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                fi
            else
                $ECHO_COMMAND "File $f gia' compresso" >> $LOG_PATH/$LOG_FILE
                # MD5 del file
                $ECHO_COMMAND "$MD5_COMMAND $LOG_PATH/$1/$d/$f" \
                >> $LOG_PATH/$LOG_FILE

                cd $LOG_PATH
                if [[ -z $2 ]]; then
                    $MD5_COMMAND $1/$d/$f \
                    1>> $LOG_PATH/$1/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                else
                    $MD5_COMMAND $1/$d/$f \
                    1>> $2/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
                fi
            fi
        done
    done
    if [ $? -ne 0 ]; then
        # Errore generazione MD5
        # Invia avviso via mail e termina
        SUBJECT="Errore generazione MD5"
        HEADER="Errore generazione MD5: controllare il log"
        BODY=`$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE`
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
    $ECHO_COMMAND "Exit MD5 function" >> $LOG_PATH/$LOG_FILE
}

MakeISO() {
```

```
# $1 Data dei log
# $2 Tipo di iso da generare
if [[ -z $2 ]]; then
    $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -M $CDRW_DEV
    -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo)
    -root $1 $LOG_PATH/$1" 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

    $MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -M $CDRW_DEV \
    -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo) \
    -root $1 $LOG_PATH/$1 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

    if [ $? -ne 0 ]; then
        # Errore nel generare la iso
        # Invia avviso via mail e termina
        SUBJECT="Errore generazione iso CD"
        HEADER="Errore nella procedura di generazione della iso: controllare il log"
        BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
else
    if [ $2 == "empty" ]; then
        $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/empty.iso -M $CDRW_DEV
        -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo) $ISO_PATH/empty" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $MKISO_COMMAND -R -l -o $ISO_PATH/empty.iso -M $CDRW_DEV \
        -C $($CDRECORD_COMMAND -s dev=$CDRW_DEV -msinfo) \
        $ISO_PATH/empty 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        if [ $? -ne 0 ]; then
            # Errore nel generare la iso
            # Invia avviso via mail e termina
            SUBJECT="Errore generazione iso vuota"
            HEADER="Errore nella procedura di generazione della iso: controllare il log"
            BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
            Mail "$SUBJECT" "$HEADER" "$BODY"
            Endlog
            exit 1
        fi
```

```
        elif [ $2 == "first" ]; then
            $ECHO_COMMAND "$MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso -root $1 $LOG_PATH/$1" \
            1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            $MKISO_COMMAND -R -l -o $ISO_PATH/$1.iso \
            -root $1 $LOG_PATH/$1 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            if [ $? -ne 0 ]; then
                # Errore nel generare la iso
                # Invia avviso via mail e termina
                SUBJECT="Errore generazione iso CD"
                HEADER="Errore nella procedura di generazione della iso: controllare il log"
                BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
                Mail "$SUBJECT" "$HEADER" "$BODY"
                Endlog
                exit 1
            fi
        fi
    fi
}

MakeCD() {
    # $1 Nome della iso da masterizzare
    # $2 Specifica come chiudere il CD
    if [[ -z $2 ]]; then
        $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -multi -data -v $ISO_PATH/$1.iso" \
        1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $CDRECORD_COMMAND dev=$CDRW_DEV -multi -data \
        -v $ISO_PATH/$1.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

        if [ $? -ne 0 ]; then
            SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
            HEADER="Errore nella procedura di masterizzazione della iso su CD:
                    controllare il log"
            BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
            Mail "$SUBJECT" "$HEADER" "$BODY"
            Endlog
            exit 1
        fi
        rm -f $ISO_PATH/$1.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
```

```
    else
        if [ $2 == "empty" ]; then
            $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/empty.iso" \
            1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            $CDRECORD_COMMAND dev=$CDRW_DEV -data \
            -v $ISO_PATH/empty.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            if [ $? -ne 0 ]; then
                SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
                HEADER="Errore nella procedura di masterizzazione della iso su CD:
                        controllare il log"
                BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
                Mail "$SUBJECT" "$HEADER" "$BODY"
                Endlog
                exit 1
            fi
            rm -f $ISO_PATH/empty.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        elif [ $2 == "close" ]; then
            $ECHO_COMMAND "$CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/$1.iso" \
            1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            $CDRECORD_COMMAND dev=$CDRW_DEV -data -v $ISO_PATH/$1.iso \
            1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

            if [ $? -ne 0 ]; then
                SUBJECT=‘hostname‘": Errore masterizzazione della iso su CD"
                HEADER="Errore nella procedura di masterizzazione della iso su CD:
                        controllare il log"
                BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
                Mail "$SUBJECT" "$HEADER" "$BODY"
                Endlog
                exit 1
            fi
            rm -f $ISO_PATH/$1.iso 1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        fi
    fi
}

Check_md5() {
    Rescan
```

```
    $MOUNT_COMMAND $CDRW_DEV /mnt 2>> $LOG_PATH/$ERROR_LOG_FILE

    if [ $? -ne 0 ]; then
        SUBJECT=‘hostname‘": Errore mount CD per check MD5"
        HEADER="Errore nella procedura di check degli MD5:
                impossibile montare il CD, controllare il log"
        BODY=$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi

    cd /mnt

    MD5_ERROR=‘$MD5_COMMAND -c $1/md5_check.txt 2>> $LOG_PATH/$ERROR_LOG_FILE | \
    $GREP_COMMAND -v "OK$" 2>> $LOG_PATH/$ERROR_LOG_FILE‘

    if [[ ! -z $MD5_ERROR ]]; then
        SUBJECT=‘hostname‘": Errore check MD5"
        HEADER="Errore nella procedura di check degli MD5: controllare il log"
        BODY=‘$ECHO_COMMAND "$MD5_ERROR"‘
        Mail "$SUBJECT" "$HEADER" "$BODY"
    fi

    cd

    $UMOUNT_COMMAND /mnt

    if [ $? -ne 0 ]; then
        SUBJECT=‘hostname‘": Errore umount CD per check MD5"
        HEADER="Errore nella procedura di check degli MD5:
                impossibile smontare il CD, controllare il log"
        BODY=$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi
}

#=============================================
```

```
# Inizializzazione file di log
#=============================================

Startlog() {
$ECHO_COMMAND "=============================================" > $LOG_PATH/$LOG_FILE
$ECHO_COMMAND "Inizio procedura di masterizzazione $DATE" >> $LOG_PATH/$LOG_FILE
$ECHO_COMMAND "=============================================" >> $LOG_PATH/$LOG_FILE
}

Starterrorlog() {
$ECHO_COMMAND "=============================================" > $LOG_PATH/$ERROR_LOG_FILE
$ECHO_COMMAND "Errori procedura di masterizzazione $DATE" >> $LOG_PATH/$ERROR_LOG_FILE
$ECHO_COMMAND "=============================================" >> $LOG_PATH/$ERROR_LOG_FILE
}

Endlog() {
$ECHO_COMMAND "=============================================" >> $LOG_PATH/$LOG_FILE
$ECHO_COMMAND "Procedura di masterizzazione terminata $DATE" >> $LOG_PATH/$LOG_FILE
$ECHO_COMMAND "=============================================" >> $LOG_PATH/$LOG_FILE
}

#=============================================
# Inizializzazione argomenti
#=============================================

while getopts "hs:d:m:p:C:c" OPTION
do
     case $OPTION in
         h)
             Usage
             exit 1
             ;;
         s)
             SESSIONS_LIMIT=$OPTARG
             ;;
         d)
             CUSTUM_DATE="true"
             LOG_DATE=$OPTARG
             ;;
         m)
             MD5_DATE=$OPTARG
```

```
            ;;
        p)
            CUSTUM_PATH=$OPTARG
            ;;
        C)
        Startlog
        Starterrorlog
        Check_md5 "$OPTARG"
        Endlog
        exit 0
            ;;
        c)
            CLOSE="true"
            ;;
        ?)
            Usage
            exit 1
            ;;
    esac
done

#================================================
# Avvio procedura senza opzioni
#================================================

Startlog
Starterrorlog


#================================================
# Operazioni preliminari e CD info
#================================================

Rescan
sleep 5

# Variabili
EMPTY_MEDIA=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GREP_COMMAND empty 2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GAWK_COMMAND ’{print($4)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘
```

```
APPENDABLE_MEDIA=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GREP_COMMAND Appendable 2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GAWK_COMMAND ’{print($3)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

if [ -z $EMPTY_MEDIA ]; then
    SUBJECT="Errore recupero informazioni CD"
    HEADER="Errore nel recupero delle informazione del CD: controllare il log"
    BODY=‘$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE‘
    Mail "$SUBJECT" "$HEADER" "$BODY"
    Endlog
    exit 1
fi

TOTAL_MEDIA_SPACE=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GREP_COMMAND Total 2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GAWK_COMMAND ’{print($7)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

TOTAL_MEDIA_SPACE=${TOTAL_MEDIA_SPACE%/*}


#================================================
# Genera MD5
#================================================

if [[ $MD5_DATE ]]; then
    $ECHO_COMMAND "Genera file MD5 della data $MD5_DATE" >> $LOG_PATH/$LOG_FILE
    if [[ $CUSTUM_PATH ]]; then
        MD5 "$MD5_DATE" "$CUSTUM_PATH"
    else
        MD5 "$MD5_DATE"
    fi
    Endlog
    exit 0
fi

#================================================
# Chiusura CD
#================================================
```

```
if [[ $CLOSE == "true" && $CUSTUM_DATE == "false" ]]; then
    $ECHO_COMMAND "Chiusura CD con iso vuota" >> $LOG_PATH/$LOG_FILE
    MakeISO "" "empty"
    MakeCD "" "empty"
    Endlog
    exit 0
fi

#================================================
# Compressione log e md5
#================================================

$ECHO_COMMAND "MD5 del: $LOG_DATE" > $LOG_PATH/$LOG_DATE/$MD5_FILE

$ECHO_COMMAND "" >> $LOG_PATH/$LOG_DATE/$MD5_FILE

$ECHO_COMMAND "--------------------------------" \
1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

MD5 "$LOG_DATE"

$ECHO_COMMAND "--------------------------------" \
1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

$SED_COMMAND -i "1iTo: $DESTINATION_MD5_MAIL_ADDRESS\n\
From: $SOURCE_MAIL_ADDRESS\n\
Reply-To: $REPLY_TO_MAIL_ADDRESS\n\
Subject: `hostname`: MD5 check del $LOG_DATE" \
$LOG_PATH/$LOG_DATE/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

# Invio lista MD5
$MAIL_COMMAND -t < $LOG_PATH/$LOG_DATE/$MD5_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

#================================================
# Avvio generazione ISO da masterizzare
#================================================

if [ $EMPTY_MEDIA = $NO ]; then
    # CD chiuso e non scrivibile
    if [ $APPENDABLE_MEDIA = $NO ]; then
```

```
    # Invia avviso via mail di sostituzione CD
    SUBJECT="AVVISO: Sostituzione CD"
    HEADER="CD non scrivibile: provvedere a inserire un nuovo CD"
    BODY=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
          2>> $LOG_PATH/$ERROR_LOG_FILE‘
    Mail "$SUBJECT" "$HEADER" "$BODY"
    Endlog
    exit 1
fi
# CD aperto
# Procedo a generare la iso
MakeISO "$LOG_DATE"

ISO_SPACE=‘du -m $ISO_PATH/$LOG_DATE.iso 2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GAWK_COMMAND ’{print($1)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

ISO_SPACE=${ISO_SPACE%M*}

# Calcolo spazio rimanente e invio avviso cambio CD
SPACE=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GREP_COMMAND Remaining 2>> $LOG_PATH/$ERROR_LOG_FILE | \
$GAWK_COMMAND ’{print($7)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

SPACE_LEFT=${SPACE%/*}

if [[ $SPACE_LEFT -lt $ISO_SPACE ]]; then
    # Spazio insufficiente
    # Invia avviso via mail
    SUBJECT="ERRORE: Sostituzione CD"
    HEADER="Spazio insufficiente: provvedere a inserire un nuovo CD"
    BODY=‘CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0 \
          2>> $LOG_PATH/$ERROR_LOG_FILE | \
          $GREP_COMMAND Remaining 2>> $LOG_PATH/$ERROR_LOG_FILE‘
    Mail "$SUBJECT" "$HEADER" "$BODY"
    # Chiudi CD e termina
    MakeISO "" "empty"
    MakeCD "" "empty"
    Endlog
    exit 1
fi
```

```
        # 2 - Controlla sessioni rimanenti
        SESSIONS=‘$CDRDAO_COMMAND disk-info --device $CDRW_DEV -v 0
        2>> $LOG_PATH/$ERROR_LOG_FILE | \
        $GREP_COMMAND Sessions 2>> $LOG_PATH/$ERROR_LOG_FILE | \
        $GAWK_COMMAND ’{print($3)}’ 2>> $LOG_PATH/$ERROR_LOG_FILE‘

        let SESSIONS=$SESSIONS+1

        if [[ $SESSIONS_LIMIT -le $SESSIONS ]]; then
            # 2.1 - Se e’ l’ultima sessione masterizza e chiudi il CD
            MakeCD "$LOG_DATE" "close"
            Check_md5 "$LOG_DATE"

            # Invia avviso via mail
            SUBJECT="AVVISO: Sostituzione CD"
            HEADER="Sessioni terminate: provvedere a inserire un nuovo CD"
            BODY="Sessioni limite impostate a: $SESSIONS_LIMIT"
            BODY="$BODY"‘$ECHO_COMMAND -e "\nSessioni presenti nel CD: $SESSIONS"‘
            Mail "$SUBJECT" "$HEADER" "$BODY"
        else
            # 2.2 - Altrimenti masterizza e lascia aperto il CD
            if [[ $CLOSE == "true" ]]; then
                MakeCD "$LOG_DATE" "close"
                Check_md5 "$LOG_DATE"

                # Invia avviso via mail
                SUBJECT="AVVISO: Sostituzione CD"
                HEADER="Il cd e’ stato chiuso manualmente: provvedere a inserire un nuovo CD"
                BODY=""
                Mail "$SUBJECT" "$HEADER" "$BODY"
            else
                MakeCD "$LOG_DATE"
                Check_md5 "$LOG_DATE"
            fi
        fi
    else
        # CD vuoto
        # Procedo a generare la iso
        MakeISO "$LOG_DATE" "first"

        ISO_SPACE=‘du -m $ISO_PATH/$LOG_DATE.iso 2>> $LOG_PATH/$ERROR_LOG_FILE |\
```

```
     $GAWK_COMMAND '{print($1)}' 2>> $LOG_PATH/$ERROR_LOG_FILE`

    ISO_SPACE=${ISO_SPACE%M*}

    # Calcolo spazio rimanente e invio avviso cambio CD
    if [[ $TOTAL_MEDIA_SPACE -lt $ISO_SPACE ]]; then
        # Spazio insufficiente
        # Invia avviso via mail e termina
        SUBJECT="AVVISO: Sostituzione CD"
        HEADER="Spazio insufficiente:
                non e' possibile masterizzare la iso in un CD, inserire un DVD"
        BODY=`$TAIL_COMMAND $LOG_PATH/$ERROR_LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE`
        Mail "$SUBJECT" "$HEADER" "$BODY"
        Endlog
        exit 1
    fi

    if [[ $CLOSE == "true" ]]; then
        MakeCD "$LOG_DATE" "close"
        Check_md5 "$LOG_DATE"

        # Invia avviso via mail
        SUBJECT="AVVISO: Sostituzione CD"
        HEADER="Il cd e' stato chiuso manualmente: provvedere a inserire un nuovo CD"
        BODY=""
        Mail "$SUBJECT" "$HEADER" "$BODY"
    elif [[ $SESSIONS_LIMIT == "1" ]]; then
        MakeCD "$LOG_DATE" "close"
        Check_md5 "$LOG_DATE"

        # Invia avviso via mail
        SUBJECT="AVVISO: Sostituzione CD"
        HEADER="Sessioni terminate: provvedere a inserire un nuovo CD"
        BODY="Sessioni limite impostate a: $SESSIONS_LIMIT"
        BODY="$BODY"`$ECHO_COMMAND -e "\nSessioni presenti nel CD: $SESSIONS"`
        Mail "$SUBJECT" "$HEADER" "$BODY"
    else
        MakeCD "$LOG_DATE"
        Check_md5 "$LOG_DATE"
    fi
fi
```

```
$ECHO_COMMAND "------------------------------" \
1>> $LOG_PATH/$LOG_FILE 2>> $LOG_PATH/$ERROR_LOG_FILE

Endlog

#================================================
# Termine Script
#================================================
```

A.3 Script per il rescan del canale SCSI

```
#!/bin/bash

################################################
#
# Copyright (c) 2011 Gabriele Renzini
# Distributed under the GNU GPL v2.
# For full terms see: http://www.gnu.org/licenses/gpl.html GPL version 2.
#
# Ripartizione Servizi Informatici e Statistici
# University of Perugia
#
################################################

CDRW_DEV=‘cat /proc/sys/dev/cdrom/info | grep name | awk ’{print $3}’‘

HOST_ID=‘ls -lR /sys/ | grep " $CDRW_DEV " | tail -1 |\
awk -F / ’{print $(NF-2)}’ | cut -c 1‘

TARGET_ID=‘ls -lR /sys/ | grep " $CDRW_DEV " | tail -1 |\
awk -F / ’{print $(NF-2)}’ | cut -d : -f 1,2,3‘

CDRW_ID=‘ls -lR /sys/ | grep " $CDRW_DEV " | tail -1 |\
awk -F / ’{print $(NF-2)}’ | cut -d : -f 1,2,3,4‘

# Rescan del canale scsi
echo "- - -" > /sys/class/scsi_host/host$HOST_ID/device/target$TARGET_ID/$CDRW_ID/delete
sleep 1
echo "- - -" > /sys/class/scsi_host/host$HOST_ID/scan
sleep 1
echo "/dev/$CDRW_DEV"

#=============================================
# Termine Script
#=============================================
```

A.4 Script per il controllo di esecuzione del processo rsyslog

```
#!/bin/bash

################################################
#
# Copyright (c) 2011 Gabriele Renzini
# Distributed under the GNU GPL v2.
# For full terms see: http://www.gnu.org/licenses/gpl.html GPL version 2.
#
# Ripartizione Servizi Informatici e Statistici
# University of Perugia
#
################################################

DESTINATION_MAIL_ADDRESS=youremail@example.com
SOURCE_MAIL_ADDRESS=root@‘hostname‘
REPLY_TO_MAIL_ADDRESS=youremail@example.com

DATE_COMMAND=/bin/date
GREP_COMMAND=/bin/grep
MAIL_COMMAND=/usr/sbin/sendmail
PS_COMMAND=/bin/ps
SED_COMMAND=/bin/sed

DATE=‘$DATE_COMMAND‘
PROCESS="$1"

LOG_PATH="/var/log"
MESSAGE="/tmp/msg.txt"


ERROR_LOG_FILE="process_error.log"
LOG_FILE="process.log"

var1=‘$PS_COMMAND -ef | $GREP_COMMAND -v grep |\
$GREP_COMMAND -v check_process | $GREP_COMMAND $PROCESS‘

if [[ -z $var1 ]]; then
```

```
        echo "The process is not running" > $MESSAGE
    /etc/init.d/$PROCESS start >> $MESSAGE

        $SED_COMMAND -i "1iTo: $DESTINATION_MAIL_ADDRESS\n\
Reply-To: $REPLY_TO_MAIL_ADDRESS\n\
Subject: `hostname`: $PROCESS ERROR - $DATE" \
        $MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE

        $MAIL_COMMAND -t < $MESSAGE 2>> $LOG_PATH/$ERROR_LOG_FILE
fi

#=============================================
# Termine Script
#=============================================
```

A.5 Script per il controllo dei file scritti su CD

```
#!/bin/bash

################################################
#
# Copyright (c) 2011 Gabriele Renzini
# Distributed under the GNU GPL v2.
# For full terms see: http://www.gnu.org/licenses/gpl.html GPL version 2.
#
# Ripartizione Servizi Informatici e Statistici
# University of Perugia
#
################################################

FIND_COMMAND=/bin/find
GREP_COMMAN=/bin/grep
MD5_COMMAND=/usr/bin/md5sum

if [[ $1 ]]; then
    PATH=$1
else
    PATH="."
fi

cd $PATH
$FIND_COMMAND . -maxdepth 1 -type d -regex '\./20.*' | while read dir; do
  $MD5_COMMAND -c $dir/md5_check.txt | $GREP_COMMAN -v "OK$"
done

#=============================================
# Termine Script
#=============================================
```

Appendice B

Altri script

B.1 createDB.sql

```
CREATE DATABASE Syslog;
USE Syslog;
CREATE TABLE SystemEvents
(
        ID int unsigned not null auto_increment primary key,
        CustomerID bigint,
        ReceivedAt datetime NULL,
        DeviceReportedTime datetime NULL,
        Facility smallint NULL,
        Priority smallint NULL,
        FromHost varchar(60) NULL,
        Message text,
        NTSeverity int NULL,
        Importance int NULL,
        EventSource varchar(60),
        EventUser varchar(60) NULL,
        EventCategory int NULL,
        EventID int NULL,
        EventBinaryData text NULL,
        MaxAvailable int NULL,
        CurrUsage int NULL,
        MinUsage int NULL,
        MaxUsage int NULL,
```

```
        InfoUnitID int NULL ,
        SysLogTag varchar(60),
        EventLogType varchar(60),
        GenericFileName VarChar(60),
        SystemID int NULL
);

CREATE TABLE SystemEventsProperties
(
        ID int unsigned not null auto_increment primary key,
        SystemEventID int NULL ,
        ParamName varchar(255) NULL ,
        ParamValue text NULL
);
```