Desktop-as-a-Service facile con Virtual Distro Dispatcher

Lamanna, Davide, Binario Etico Soc. Coop., via del Forte Tiburtino, 98/100 Roma, IT, davide.lamanna@binarioetico.org

Nasti, Fabrizio, Binario Etico Soc. Coop., via del Forte Tiburtino, 98/100 Roma, IT, fabrizio.nasti@binarioetico.org

Abstract

Virtual Distro Dispatcher (VDD) è un sistema distribuito che ha lo scopo di proiettare multiple istanze virtuali di sistema operativo, pienamente usabili come desktop attraverso terminali in una rete. I terminali possono essere PC obsoleti o thin client a basso consumo, tipo mini-ITX, tutti gestiti da un potente sistema centrale multiprocessore e clusterizzato. Si può parlare di soluzione Desktop-as-a-Service (DaaS), derivandone il significato da Software-as-a-Service: i desktop possono essere trasformati in un servizio a sottoscrizione, comodo, economico e scalabile. In VDD, macchine virtuali desktop vengono istanziate su un server e quindi fornite ai client nella loro interezza, su richiesta, attraverso una rete.

Parole Chiave: Cloud Computing, XEN, LTSP, Xorg, Trashware, Thin client, VDD-Project

1 INTRODUZIONE

Richard Stallman ha probabilmente ragione quando definisce il Cloud Computing l'ennesima campagna di marketing modaiola, priva di un reale contenuto tecnologico innovativo e ricca, invece, di insidie per la libertà e la privacy degli utenti. Ad ogni modo, è egualmente vero che lavorare sui temi caldi del Cloud Computing ci dà la possibilità di fare passi avanti nello studio e nello sviluppo dei sistemi distribuiti, della computazione su vasta scala, della virtualizzazione, della programmazione parallela e dell'ottimizzazione delle risorse. In quest'ottica, il progetto Virtual Distro Dispatcher (VDD) intende esplorare alcuni aspetti specifici del Cloud Computing e proporre una soluzione software basata su sistemi aperti e liberi e che tenga in considerazione anche le problematiche di privacy. L'idea di base del Cloud Computing è la fornitura di capacità computazionali le più varie in forma di servizio, ovvero "as-a-service". Servizi che puntano sull'affidabilità, vengono configurati in data center di nuova generazione, basati su tecnologie di virtualizzazione, per essere erogati in modo trasparente e standard agli utenti di una rete, nonostante l'insieme eterogeneo e distribuito delle risorse hardware e software che li compongono. E' il concetto di Utility computing, in base al quale risorse computazionali fisiche vengono impacchettate ed erogate, in modo non dissimile a quanto avviene per altri servizi pubblici, come l'elettricità o l'acqua. Questo può avvenire a diversi livelli. Nel caso delle applicazioni si parla di Software-as-a-Service (SaaS). Se si tratta di sistemi, è possibile derivare il concetto di Desktop-as-a-Service (DaaS). Le risorse fisiche degli host (memoria, spazio disco, cicli di processore, ecc.) vengono quantizzate in macchine virtuali multiple, per poi rendere disponibili da remoto queste macchine virtuali, da usarsi come desktop, attraverso la rete. In tal modo i desktop vengono trasformati in un servizio a sottoscrizione economicamente vantaggioso, scalabile e comodo. Un'altra espressione utilizzata in questo contesto è Virtualized Client Computing (VCC).

Virtual Distro Dispatcher è un sistema distribuito che ha lo scopo di proiettare on-demand istanze di sistema operativo (unix-like e non solo) pienamente funzionanti, su terminali di una rete.

2 ARCHITETTURA DI VDD

In VDD, la virtualizzazione è utilizzata per disaccoppiare l'ambiente del client (sistema operativo, applicazioni e dati) dal suo hardware host e isolarlo da altri software o sistemi che eventualmente girano sul client. VDD fa uso di un sistema di virtualizzazione di desktop per ospitare, a bordo di un server (o un gruppo di server in cluster tra loro), istanze di ambienti client completi e isolati; l'interazione con questi desktop virtuali remoti avviene attraverso sessioni utente (grafiche) virtuali. VDD sfrutta la trasparenza di rete di X-Window-System, in base alla quale la macchina dove gira un programma applicativo (l'applicazione client) può differire dalla macchina locale dell'utente (il display server). I client X-Window-System girano su macchine virtuali lato server, dove vengono create sessioni utente multiple all'interno di ambienti virtuali multipli. Il display server di X-Window-System gira sui thin client (terminali). Per i sistemi operativi privi del server X11 (come Windows e ReactOS), è possibile usare il protocollo VNC.

L'architettura di VDD viene illustrata in Figura 1. I terminali possono essere PC obsoleti o thin client a basso consumo, come i mini-ITX, che accedono ad un sistema centrale multiprocessore e clusterizzato. I terminali sono interfacce a vere e proprie macchine isolate, che possono essere create su misura per qualunque necessità, tipo, numero e quantità di risorse, nei limiti delle risorse del sistema centrale, ovviamente. Questo è possibile grazie ad un sistema di virtualizzazione basato su XEN. L'infrastruttura su cui gira questo sistema è un cluster è ad High Availability, che garantisce continuità operativa. In particolare, è stato implementato un sistema di Live Migration, grazie al quale è possibile la migrazione di macchine virtuali da un host fisico ad un altro all'interno del cluster in caso di guasti.

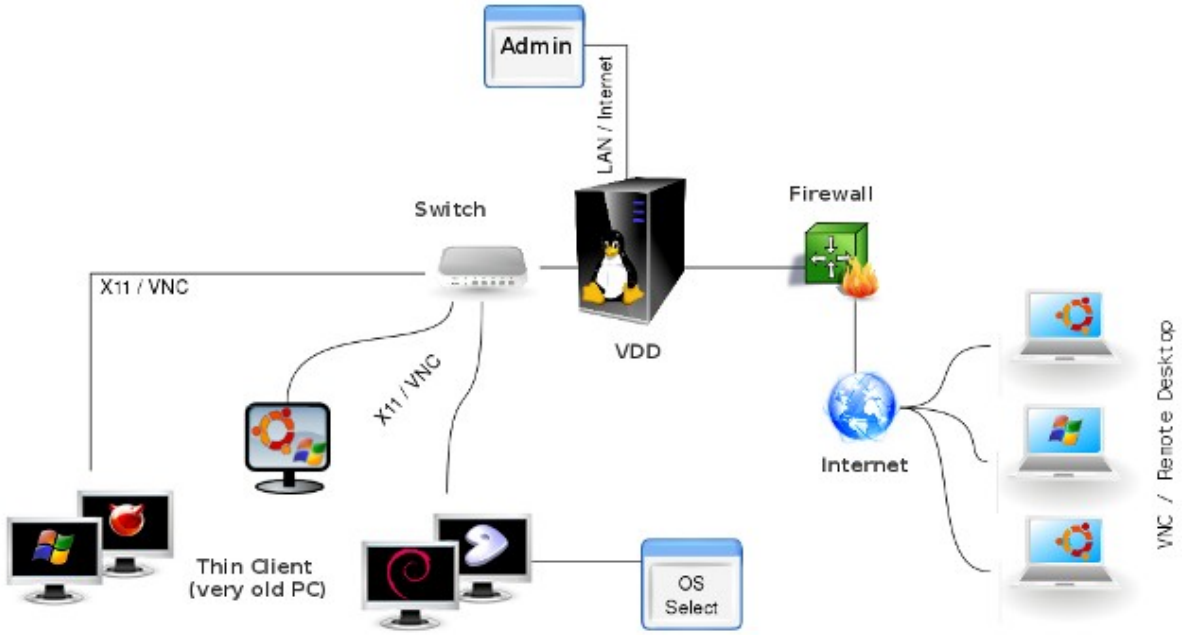

Figura 1. L'architettura di VDD.

Il meccanismo in base al quale funziona VDD è trasparente all'utente, che, persino da un PC obsoleto, può comodamente selezionare una determinata macchina, con determinate caratteristiche, sistema operativo, ambiente grafico e configurazioni, e lavorare su di essa come se fosse fisica, locale, dotata di quelle risorse e delle relative performance. Si noti, infine, che è possibile far funzionare VDD su Internet, anche se le prestazioni sono molto dipendenti dalla disponibilità di banda in upload e ancora scadenti.

3 SCHEMA DI FUNZIONAMENTO

La Figura 2 mostra il modo in cui le macchine fisiche e virtuali, i loro servizi, le loro applicazioni, i terminali e i display, entrano in relazione tra loro, in base a quali comandi e realizzando quali funzioni. La macchina fisica VDD (grande, in verde) ospita al suo interno le macchine virtuali da "dispatchare" (piccole, in giallo), che nell'esempio della figura 2 sono tre, Debian, Ubuntu e Fedora. Le macchine fisiche e virtuali sono equipaggiate con server ssh, per poter accedere ad esse dall'esterno, sia per scopi di manutenzione, che di funzionamento di VDD. La macchina fisica (il server VDD), che, come detto, è in un cluster ad High Availability, è dotata anche di servizi per l'accesso da remoto. In questo modo è possibile configurare VDD da stazioni di amministrazione, sia con la riga di comando che attraverso interfaccia grafica. Nell'esempio, due macchine, Foo e Bar, assolvono allo scopo. La soluzione web è stata progressivamente abbandonata in favore della più efficiente e completa GUI via ssh.

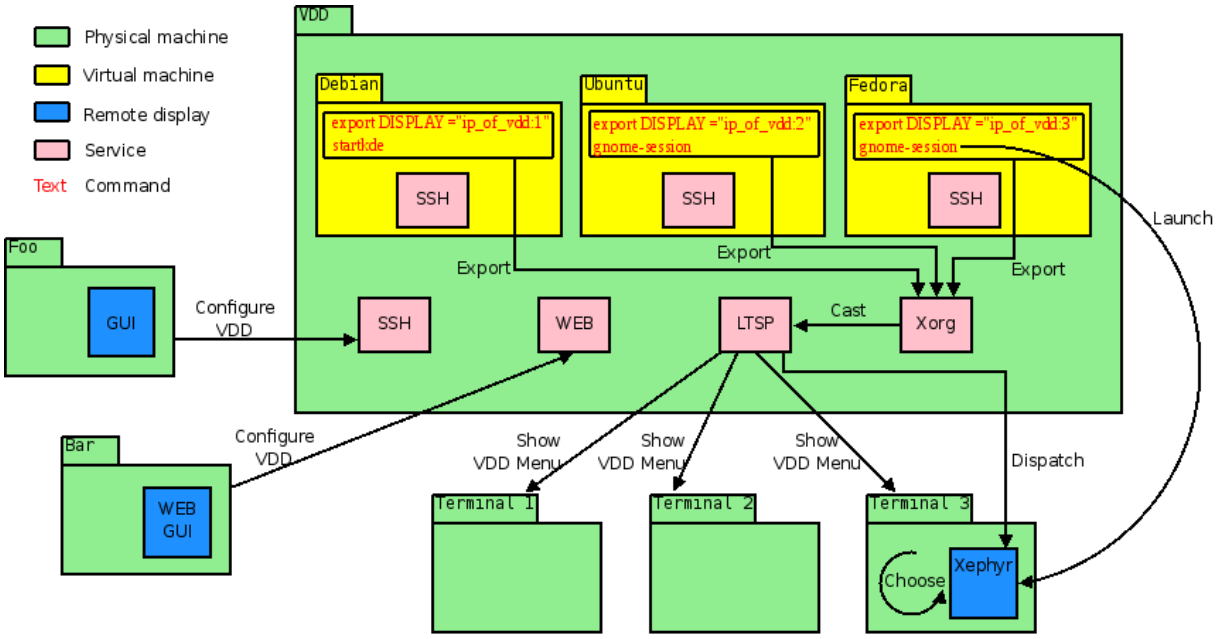

Figura 2. Schema di funzionamento.

Tre thin clients (Terminal 1, Terminal 2 e Terminal 3) sono connessi a VDD attraverso LTSP (Linux Terminal Server Project). Con LTSP è possibile avviare terminali, senza che su di essi sia installato un sistema operativo. Le prestazioni a cui lavorano i terminali sono prossime a quelle del server centrale, anche in presenza di hardware molto datato. Quando i terminali si accendono, eseguono il boot da rete e mostrano (dopo il login) un menu che è parte dell'interfaccia grafica e che consente all'utente di selezionare la distro (o in generale il sistema operativo) su cui intende lavorare. In quel momento, l'utente si trova nell'ambiente del server LTSP e quindi sul server VDD.

La variabile DISPLAY dell'utente su una determinata macchina virtuale, viene impostata per puntare ad un canale del server X che gira sul server VDD. Nell'esempio, i canali 1, 2 e 3, uno per ognuna delle tre macchine.

Dai terminali, che attraverso LTSP mostrano l'ambiente grafico del server VDD, si lancia Xephyr. Xephyr è un server X minimale eseguito in una finestra all'interno della sessione X corrente. Xephyr è contemporaneamente un client X e un server X. E' client del server X reale, e server dei propri client, che nel caso di VDD, lo invocano dai thin client. Xephyr è basato su KDrive, un server X minuscolo, progettato per ambienti con poca memoria a disposizione. KDrive evita allocazione di memoria a runtime e prova ad eseguire le operazioni "on the fly", ogni volta che ciò è possibile. Xephyr

rappresenta, di fatto, un rimpiazzo per Xnest, che consente di eseguire un server grafico X11 come client di sé stesso, all'interno di una finestra autonoma. Rispetto ad Xnest, Xephyr offre prestazioni migliori e supporto per le estensioni più avanzate (quali Composite, Damage, randr, ecc.).

Nell'esempio di figura 2, Xephyr viene lanciato in ascolto sul canale 2, per poter utilizzare la macchina Ubuntu.

Dopo aver avviato il display remoto di Xephyr, è possibile lanciare sessioni grafiche delle macchine virtuali utilizzando gestori di sessioni X11 (startkde, gnome-session, xfce4-session).

Si noti che è perfettamente possibile usare più canali dallo stesso thin client. Questa è una delle features di VDD, che consente di proiettare su ogni terminale uno o più sistemi operativi, passando comodamente dall'uno all'altro attraverso il menu grafico.

Una delle caratteristiche di Xephyr più interessanti per VDD è la modalità a tutto schermo, che rende il sistema completamente trasparente all'utente.

Tutto queste funzioni vengono assolte da script appositi, a cui è collegata l'interfaccia grafica.

4 PRIVACY

Una delle più importanti criticità delle soluzioni Cloud Computing è l'esposizione degli utenti a possibili violazioni della propria privacy, in quanto i dati personali sono in possesso di chi eroga il servizio.

VDD integra al suo interno una soluzione che risolve, almeno in parte, questo genere di problematiche. Gli utenti di VDD possono avere a propria disposizione un volume cifrato su ognuna delle Virtual Machine a cui hanno accesso. Il volume cifrato, un volume logico LVM, è montato sul filesystem del server e condiviso tramite Samba sulle Virtual Machine. Per la cifratura viene usato un sistema basato su dmcrypt-LUKS.

Su un sistema debian-based è necessario installare i pacchetti dmsetup, cryptsetup e, naturalmente samba, e verificare la presenza dei moduli "aes" e "crypt". Si procede quindi alla creazione del volume logico e alla sua cifratura (tramite passphrase) con il programma cryptsetup. Lo stesso programma si occupa poi di decifrare il volume, previo inserimento della corretta passhrase, mappandolo su un dispositivo logico, sul quale è quindi possibile creare il filesystem adeguato tramite le diverse istanze del progrmma mkfs. Sempre tramite cryptsetup si procede infine alla "chiusura" del volume cifrato, la cui preparazione è a questo punto completata.

Il pacchetto libpam-mount consente poi di utilizzare il comando mount.crypt per decifrare e montare il dispositivo sul filesystem del server. Per l'automazione di queste procedure nel contesto VDD si fa uso delle funzionalità messe a disposizione dal comando "sudo", opportunamente configurato per definirne funzionalità e limitazioni (utenti e comandi autorizzati, richiesta o meno di password, ecc.)

Ciascun utente può infine attivare la condivisione del punto di mount del proprio volume cifrato, sfruttando la funzionalità di "usershare" fornita da samba3. A tale scopo, oltre ad attivare e configurare tale funzionalità nel file di configurazione di samba, è necessario che l'utente sia membro del gruppo sambashare. Naturalmente l'utente deve essere aggiunto tra gli utenti samba tramite il comando smbpasswd, con il quale si imposta anche la relativa password. L'utente può quindi attivare la condivisione che sarà accessibile previo inserimento delle credenziali impostate.

L'utilizzo di volumi crittati dà agli utenti la possibilità di tenere i propri dati privati e in sicurezza rispetto ad intrusi. Potrebbe esserci sempre il rischio, però, che vengano sniffati i dati in fase di trasferimento dal thin client al volume cifrato (vedi TEMPEST attack). Mentre in un ambiente LAN controllato questo rappresenta un problema poco rilevante, non è invece da sottovalutare nel momento in cui VDD entra in funzione “in the cloud”, dove certamente il rischio è più alto. In questi casi, bisogna ricorrere ad SSL.

Ad ogni modo, la soluzione proposta non risolve del tutto il problema di privacy dei sistemi basati sul Cloud Computing, in quanto è sempre possibile monitorare le attività degli utenti e profilarli a scopi pubblicitari o di altro tipo.

5 INTERFACCIA GRAFICA

L'ossatura della GUI è costituita da script, disponibili sul sito insieme a tutta la documentazione, integrati nella GUI attraverso Zenity. Gli screenshots di Figura 3 e 4 mostrano alcune delle scelte presentate all'utente.

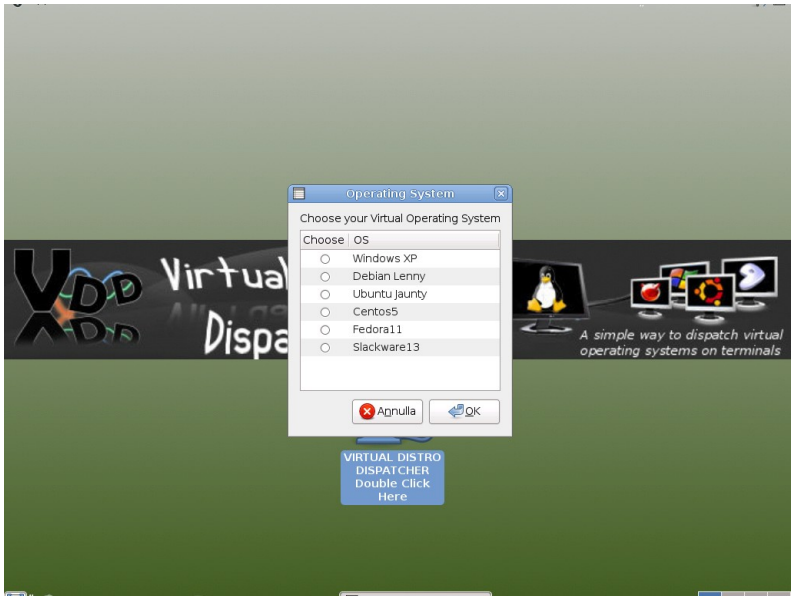

Figura 3. Selezione del Sistema Operativo della macchina da dispatchare.

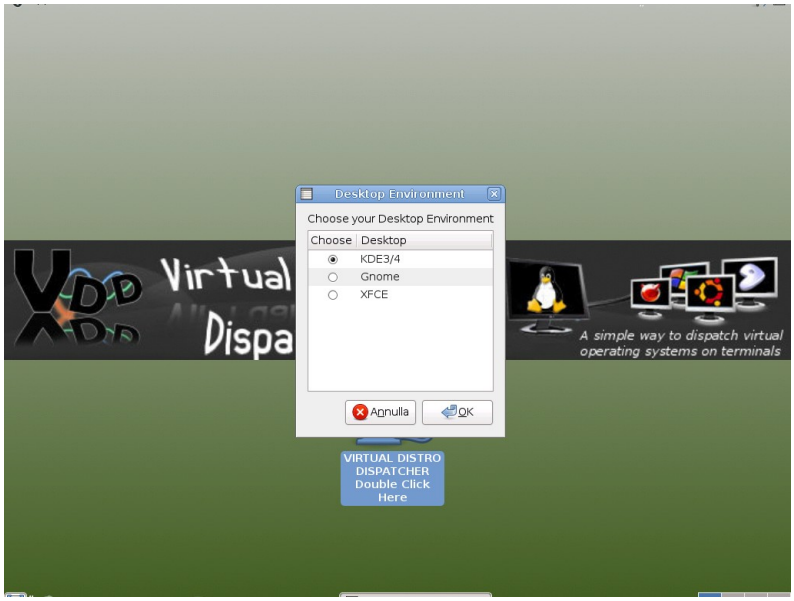

Figura 4. Selezione del Desktop Environment (solo per GNU/Linux).

Per quanto riguarda l'amministrazione, VDD integra al suo interno virt-manager. Per essere gestite, le macchine virtuali devono essere "importate" nello xenstore, in modo che la libreria libvirt possa riconoscerle e gestirle. Ogni volta che si modifica la configurazione, la virtual machine deve essere cancellata dallo xenstore e poi riaggiunta. Una volta inserite nello xenstore, le virtual machine possono essere avviate (e spente) sia da linea di comando che attraverso l'interfaccia grafica.

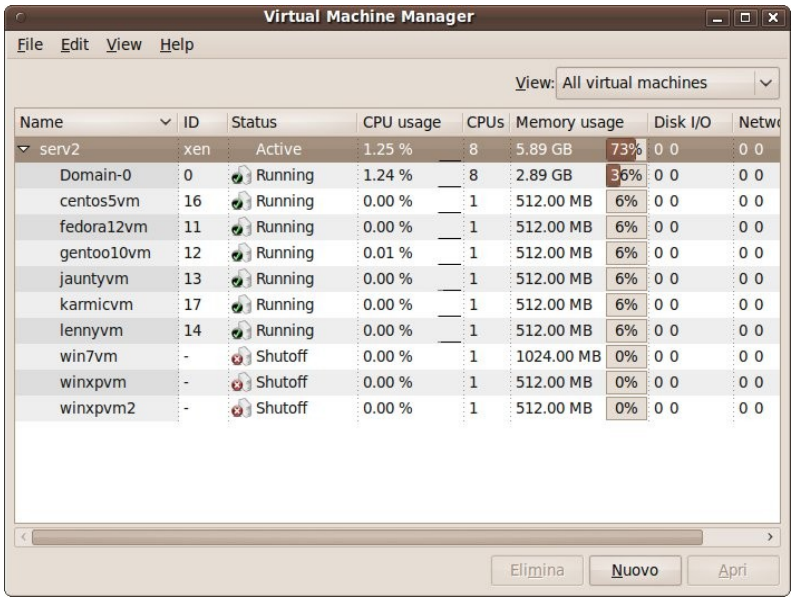

Figura 5. Pannello di controllo delle macchine virtuali.

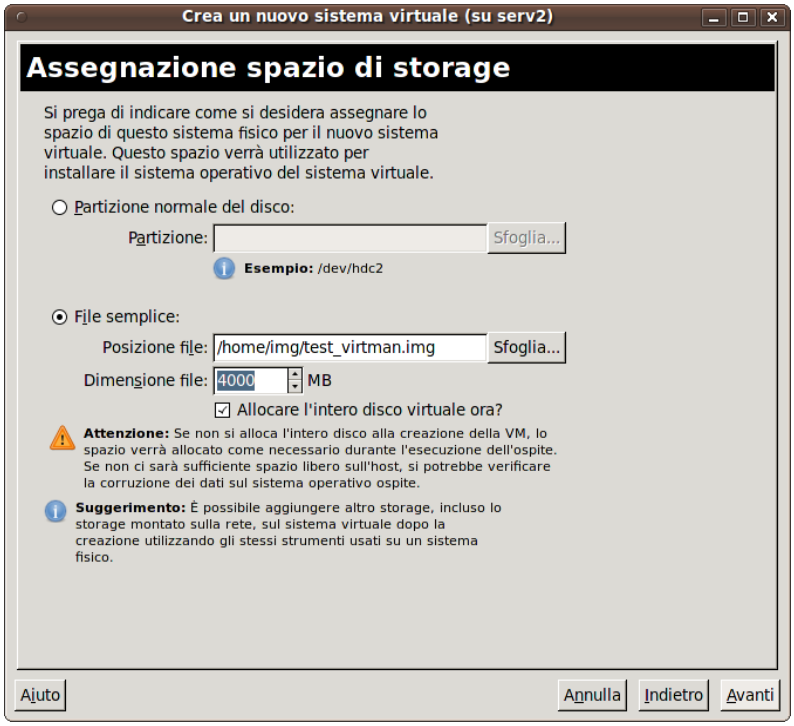

Figura 6. Una fase della creazione di una nuova macchina virtuale.

Sono previste 2 modalità di utilizzo del gestore: in locale (sull'host xen) e da remoto (da un client qualunque). In modalità locale è attiva la funzionalità di creazione di nuove macchine virtuali, sia da linea di comando che da interfaccia grafica. Sia da locale che da remoto, il programma ci consente di visualizzare lo stato delle virtual machine, i relativi dettagli e l'uso di risorse e ci consente inoltre di spegnerle, avviarle, rebootarle (vedi figura 5). Figura 6 e 7 mostrano l'allocazione di spazio, cpu e memoria per una nuova macchina da creare (modalità locale).

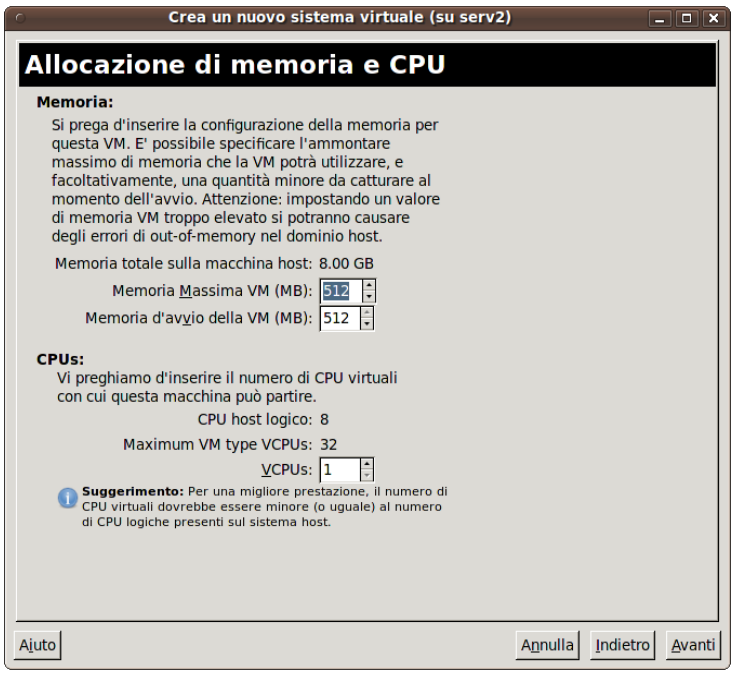

Figura 7. Un'altra fase della creazione di una nuova macchina virtuale.

6 TO DO LIST E COMMUNITY

Grazie al finanziamento della fondazione NLNet , VDD ha avuto una nuova fase di sviluppo. Sono stati completati l'interfaccia grafica, l'analisi delle performance e i test di usabilità. La documentazione, gli how-to e il codice sono disponibili sul sito www.vdd-project.org

Virtual Distro Dispatcher usa software Open Source/GPL e protocolli di comunicazione liberi ed è naturalmente rilasciata come Software Libero. L'infrastruttura consente di far girare sistemi operativi proprietari come macchine guest e questo è regolato da specifiche licenze, costi e limitazioni, che dovrebbero essere presi in considerazione dagli utenti di VDD.

Gli sviluppatori cercano persone e/o organizzazioni che vogliano collaborare, promuovere, sviluppare, discutere. Sul sito è possibile iscriversi alla newsletter, al forum di discussione, e si può richiedere di essere inseriti nella lista del team di sviluppo e poter quindi accedere agli strumenti di lavoro collaborativi presenti nell'area riservata.

Stiamo in questa fase ridefinendo la roadmap. La nuova to_do list conterrà di sicuro uno sviluppo delle feature di clustering; dopo aver implementato l'High Availability, potremmo sperimentare il Load Balancing. Dovremmo lavorare anche su un event log facilmente fruibile. Rispetto all'accounting, stiamo per ora usando quello standard di LTSP, mentre invece si potrebbe pensare di introdurre qualcosa di più configurabile e granulare, integrando OpenLDAP. Abbiamo poi altre idee su possibili sviluppi, che potrete scoprire ed incrementare contattandoci e portando le vostre! :-)

Bibliografia

1.Flavio Bertini, Davide Lamanna and Roberto Baldoni – “Virtual Distro Dispatcher: A costless distributed virtual environment from Trashware” - ISPA '07: Proceedings of the 5th international symposium on Parallel and Distributed Processing and Applications, pages 223—234

2.Jeff Dike – “User Mode Linux” (Bruce Perens' Open Source Series), Prentice Hall PTR (April 22, 2006), 352 pages

3.David Chisnall – “The Definitive Guide to the XEN Hypervisor” - Prentice Hall PTR; 1 edition (November 19, 2007), 320 pages

4.Linux Terminal Server Project – http://www.ltsp.org

5.Gerald J. Popek and Robert P. Goldberg – “Formal Requirements for Virtualizable Third Generation Architectures” - Communications of the ACM, Volume 17, Issue 7, Pages: 412 – 421

6.Michael Rose (Industry Developments and Model) - “Virtualized Client Computing: A Taxonomy” - Dec 2007, http://www.idc.com/getdoc.jsp?containerId=209671

7.C. Hewitt - “ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing” - IEEE Internet Computing, Sept./Oct. 2008, pp. 96-99

8.Ruggero Russo, Davide Lamanna and Roberto Baldoni – “Distributed software platforms for rehabilitating obsolete hardware” - OSS '05: Proceedings of The First International Conference on Open Source Systems, pp. 220-224

9.Brendan Cully, Geoffrey Lefebvre, Dutch Meyer, Mike Feeley, and Norm Hutchinson - “Remus: High Availability via Asynchronous Virtual Machine Replication” - In proceedings of the 5th USENIX Symposium on Networked System design and implementation, pp. 161–174 – Awarded Best Paper

10. Salvatore Cristofaro, Flavio Bertini, Davide Lamanna and Roberto Baldoni – “Virtual Distro Dispatcher: a light-weight Desktop-as-a-Service solution ” - First International Conference on Cloud Computing October 19 - 21, 2009, Munich, Germany, CLOUDCOMP 2009, ICST, 2009