An Overview Regarding the State of the Virtualization Technology

eng. Ovidiu Gherman

***Abstract* - The development of cloud and grid services brought into light an entire slew of new technologies. One of the most prominent technologies is virtualization. With hardware support integrated even in consumer-grade equipment, virtualization has the potential to be a game changer in the IT industry. It's set of advantages allows the implementation of various features that until now were hard to manage: backup, custom deployment, job migration. With software platforms being more and more complex, the virtualization is an option that must be considered when using complex services and software.**

***Index Terms* - virtualization, hypervisor, virtualization architecture, libvirt, virtual container.**

I. Introducere

VIRTUALIZAREA resurselor software [1] este o tendință des întâlnită în platformele de calcul moderne. Această tehnologie aduce avantajele indiscutabile – securitate, modularizare, toleranța la erori, ușurința în administrare pe platforme eterogene, etc., avantaje care contrabalansează pierderea de performanță apărută în urma utilizării sistemelor de virtualizare. În scopul ameliorării performanțelor , industria modernă de hardware IT furnizează sisteme cu capabilități de virtualizare hardware integrate, lucru ce ușurează implementarea acestei tehnologii în sistemele de producție.

Administrarea resurselor eterogene (sisteme cu specificații hardware și software diferite) pune din, acest punct de vedere, probleme deosebite. Administrarea eficientă (atât din punct de vedere al utilizatorului care dorește acces imediat la resurse, cât și din cel al furnizorului de servicii care dorește satisfacerea a cât mai multor utilizatori cu un consum minim de resurse) trebuie să se bazeze pe algoritmi de analiză și echilibrare a sarcinilor primite pe resursele disponibile. Este necesară utilizarea unei arhitectură coerente și scalabile care să ofere serviciile așteptate de la un sistem de tip grid într-un mod cât mai satisfăcător pentru cât mai mulți utilizatori.

Este necesară dezvoltarea unei platforme care să îmbine virtualizarea resurselor software (sisteme de operare, aplicații utilizator) cu administrarea resurselor hardware cu necesități specifice (resurse învechite, platforme hardware speciale - de exemplu sistemele bazate pe procesorul PowerXCell 8i, etc.), totul integrat într-un pachet care pe de o parte să ușureze instalarea, configurarea și utilizarea sistemului, astfel încât accentul să fie pus mai puțin pe administrarea efectivă a sistemului și mai mult pe utilizarea sa în producție, iar pe de altă parte să permită unui utilizator al sistemului să își execute aplicațiile într-un mod cât mai simplu și rapid.

II. Tehnici de virtualizare

La ora actuală domeniul virtualizării și a para-virtualizării se bucură de o atenție deosebită în special datorită expansiunii tehnologiilor cloud și a serviciilor de tip HPC/hosting distribuite.

Sistemele virtualizate permite lansarea rapidă în execuție a aplicațiilor dorite în interiorul unui sistem de operare care rulează într-un container virtual, separat de gazda fizică. Aplicația (și sistemul de operare-oaspete) vor accesa resursele hardware prin intermediul unei biblioteci dedicate de virtualizare, care translatează cererile către mașina-gazdă. Acest hipervizor permite accesul la resurse hardware bare metal în mod direct sau indirect (în funcție de gradul de virtualizare) pentru mașina/mașinile curente. Acest nivel de separare permite manipularea containerului în diverse moduri (reconfigurare, salvarea instanței curente aflate în lucru, etc.), izolarea aplicației-utilizator (și a sistemului de operare pe care aceasta lucrează) – lucru ce îmbunătățește securitatea sistemului fizic și stabilitatea sa [2],[3].

Sistemele de operare Linux se pretează foarte bine acestei tehnologii, el suportând infrastructura de virtualizare KVM (Kernel-based Virtual Machine) în mod nativ. Aceasta expune kernelul Linux hipervizoarelor, astfel încât acestea să poată beneficia de acces la resursele fizice în mod direct, fără emulare. Existența API-urilor precum *libvirt* permit dezvoltarea acestor capabilități pentru implementarea hipervizoarelor precum Xen, KVM/QEMU sau VirtualBox sau a sistemelor de management precum OpenVZ. Acestea, la rândul lor, vor administra containerele virtuale permițând execuția sistemului de operare inclusiv și a aplicațiilor instalate în acestea (Figura 1).

Manuscript received December 2, 2014. This work was supported in part by the project "Sustainable performance in doctoral and post-doctoral research PERFORM–Contract no. POSDRU/159/1.5/S/138963", project co-funded from European Social Fund through Sectorial Operational Program Human Resources 2007-2013.

O. I. Gherman is with the Computers, Electronics and Automation Department of Faculty of Electrical Engineering and Computers Science, "Ștefan cel Mare" University of Suceava, 13 University Street, Romania (e-mail: ovidiug@eed.usv.ro).

Această tehnologie are şi dezavantaje. Unul din principalele dezavantaje este ridicarea gradului de complexitate în ceea ce priveşte administrarea sistemului de operare în uz. În mod normal acest lucru este (sau ar trebui sa fie – pentru un sistem de operare modern, în condiţii de utilizare normale) transparent pentru utilizator. Dar prospectul virtualizării (şi avantajele oferite de aceasta) [4] contrabalansează greutatea în utilizare şi administrare. Există la ora actuală implementări de platforme care încearcă să rezolve aceste dificultăţi prin diverse arhitecturi şi middleware-uri care să automatizeze procesul.

Un alt dezavantaj este overhead-ul care apare prin consumul de resurse atât de sistemul de operare gazdă - pe care rulează şi hipervizorul - cât şi de sistemul oaspete cu suita de aplicaţii-client aflată în uz. Pentru sistemele dedicate există însă paşi care pot ameliora situaţia prin folosirea de sisteme de operare preparate pentru virtualizare (sisteme minimale care implementează resurse optimizate pentru rularea containerelor virtuale, inclusiv sisteme de fişiere personalizate [5]), a căror scop este rularea în bune condiţiuni doar a hipervizorului; restul maşinii nu este expusă către utilizator, acesta accesând doar maşinile virtuale care pot fi încărcate în funcţie de natura lucrurilor care trebuie rezolvate. Pentru sistemele de uz general aceste tehnici de obicei nu pot fi implementate pentru ca afectează flexibilitatea sistemului (şi capabilitatea sa de a rula aplicaţii generice).

Totuşi, dacă resursele hardware sunt optimizate pentru rularea aplicaţiilor virtuale (de exemplu prin tehnologiile Intel VT-x, VIA VT sau AMD-V) performanţa acestor implementări este mult îmbunătăţită.

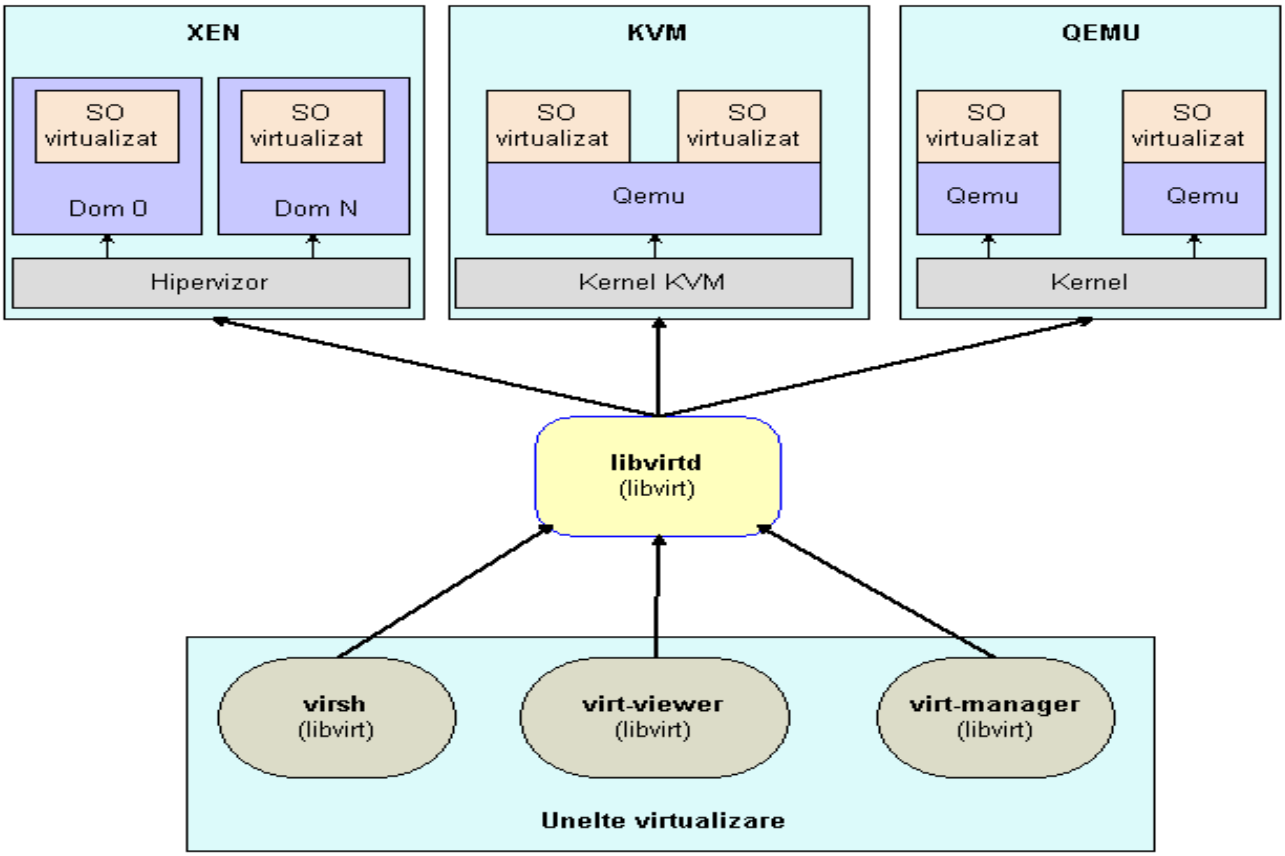

Fig. 1. Mecanismul de virtualizare pentru sistemele de operare Linux.

III. Unelte de virtualizare. Hipervizoare

Există o diversitate de soluţii de virtualizare implementate pe baza acestor mecanisme [6],[7] pentru diverse cazuri de utilizare şi dedicate rezolvării anumitor probleme. Soluţiile integrate oferite de producătorii (RedHat, IBM, VMware) de software (sau software as service) [8],[9] au un grad de personalizare ridicat, dar sunt foarte complexe pentru a putea acoperi o gamă cât mai mare de cazuri de utilizare. Soluţiile dedicate sunt mai uşor de implementat şi utilizat dar au şi aplicabilitate mai redusă.

Există multe platforme de virtualizare care permit administrarea imaginilor virtuale disponibile şi a resurselor pe care se face lansarea cum ar fi Xen [10], QEMU [11] sau OpenVZ. Acestea facilitează rularea sistemelor de operare şi a aplicaţiilor utilizator (inclusiv cele cu necesităţi hardware şi software speciale), actualizarea lor şi separarea nivelului-

utilizator de maşina în sine.

La nivelul aplicaţiilor software există inclusiv soluţii care să permită izolarea resurselor sau lansarea aplicaţiilor în spaţii-utilizator izolate de sistemul de operare; Docker sau BusyBox sunt astfel de sisteme utilizate pe scară largă în acest domeniu. Utilizarea containerelor software pentru a izola aplicaţiile utilizator de sistemul de operare (şi între diferiţi utilizatori) permite creşterea nivelului calităţii serviciilor oferite, permiţând implementarea facilă a unor tehnici necesare în administrarea sarcinilor HPC – asigurarea checkpointurilor, toleranţa la erori, migrarea şi redimensionarea sarcinilor.

IV. CAZURI DE UTILIZARE A SISTEMELOR VIRTUALIZATE

Gridurile computaţionale reprezintă sisteme foarte dinamice care pun probleme importante pentru planificarea sarcinilor datorită naturii lor eterogene şi distribuite. De obicei este necesară menţinerea unui anumit nivel de performanţă, ţinând cont de cerinţele de calitate a serviciilor şi de siguranţă/robustete a sistemului (hardware şi în special software).

Planificarea sarcinilor poate avea - de obicei - unul dintre cele doua scopuri:

- calculele de înaltă performanţă (se încearcă minimizarea timpului de execuţie în programarea paralela sau HPC) – high performance computing;
- execuţia a cât mai multor sarcini care să se finalizeze cu succes, pe unitatea de timp (se doreşte asigurarea unei disponibilităţi crescute a resurselor şi atingerii parametrilor de calitate predefiniţi – mai ales în cazul gridurilor de uz general) – high throughput computing; în acest caz, planificatorul trebuie să decidă ordinea şi locul de execuţie al sarcinilor cu îndeplinirea anumitor criterii - de obicei criterii de calitate a serviciilor (non-trivială).

Maparea sarcinilor şi a restului deciziilor de planificare pe resursele de calcul conform unei planificări poate fi realizată dinamic, în momentul primirii sarcinii (mod on-line sau planificare dinamică) sau atunci când un număr predefinit de sarcini a fost recepţionat (modul batch). Dacă în primul caz planificatorul permite mai multă flexibilitate şi un timp de răspuns scăzut, în cel de-al doilea caz se poate realiza o planificare mai eficientă deoarece planul rezultat poate fi modificat în funcţie de noile sarcinile (replanificare). Replanificarea dinamică pentru fiecare nouă sarcină este mai puţin fezabilă din cauza necesarului computaţional (care se traduce în timp de răspuns crescut din partea planificatorului şi prin resurse ocupate pentru sistemul de management, în detrimentul aplicaţiilor-client), totuşi aduce avantaje certe pentru sisteme distribuite eterogene în care performanţa resurselor variază în timp (adăugări/eliminări de noduri, creşterea încărcării nodurilor, etc.) ceea ce permite o mai bună abordare a caracteristicilor QoS ce vor fi implementate [12].

O dată stabilită planificarea execuţiei, aplicaţiile-client (sau maşinile virtuale ce conţin respectivele aplicaţii) sunt plasate pe maşinile-ţintă şi lansate în execuţie ca sarcini. În continuare, un aspect important este monitorizarea rulării acestora şi asigurarea fiabilităţii sarcinilor respective. În numeroase cazuri se pot produce opriri necomandate ale sarcinilor din diverse motive (lipsa/căderea resurselor necesare din motive hardware sau software, aplicaţiile pot deveni non-responsive din cauza unor probleme de implementare, realocarea prioritară a resurselor, etc.). Pentru a preveni pierderea datelor deja procesate este necesar un mecanism de administrare şi control a acestora. Utilizarea virtualizării oferă această oportunitate prin capacitatea de checkpointing şi de migrare a sarcinilor de pe o resursă pe alta [11].

Checkpoint-urile sunt înregistrări de stare ale sarcinilor în lucru (containerul virtual, starea sa internă şi aplicaţiile care rulează) la un anumit moment din procesul de execuţie al sarcinii. Aceste stări generate – deşi implică un overhead din punct de vedere al resurselor computaţionale şi de stocare – pot fi depozitate pe disc şi arhivate sau utilizate pentru a recupera o stare anterioară a sarcinii eşuate sau oprite. Execuţia poate fi continuată de la punctul de salvare şi finalizată.

Toate aceste înregistrări pot fi stocate la fiecare operaţiune de checkpointing sau doar ultima salvare poate fi memorată, restul fiind distruse (pentru economisirea spaţiului). Dacă sarcina se termină cu succes şi datele sunt extrase cu succes, instanţele salvate se şterg.

Această procedură se poate realiza la două nivele:

- La nivelul aplicaţiei: aplicaţia în sine trebuie să implementeze această facilitate şi să salveze datele intermediare şi starea sa internă. De asemenea aplicaţia trebuie să permită rularea algoritmului intern de la o stare avansată de execuţie. Această abordare este portabilă şi flexibilă dar necesită o dezvoltare corespunzătoare la nivelul programului rulat, ceea ce se traduce prin timp alocat implementării mecanismului respectiv (de obicei o sarcina laborioasă). O asemenea abordare poate creşte reziliența programului la eventualele erori de programare care pot apare (programul poate fi modificat, recompilat şi execuţia sa pornită pe baza datelor intermediare).
- La nivelul sistemului: instanţa salvată este procesată de către sistemul de operare virtualizat. Utilizatorul nu este implicat în proces, acesta petrecându-se automat (în funcţie de anumite criterii prestabilite). Metoda are o flexibilitate şi o portabilitate mai scăzută, are un necesar de resurse de stocare mai ridicat dar poate fi automatizată (la fel ca şi repornirea), transparent pentru utilizator. Totuşi, în cel mai dezavantajos caz de utilizare, dacă aplicaţia este defectuos implementată, aceasta trebuie repornită în cazul unei opriri necomandate, pierzându-se orice avantaj adus de salvarea stării (practic ea este re-rulată).

Această tehnică permite şi operaţiunile de migrare a sarcinilor de lucru (cu containerul în care rezidă) pe alte resurse, la un moment de timp ulterior (migrarea sarcinilor de execuţie). Acest lucru poate permite re-planificarea dinamică a execuţiei sarcinilor în sistem, salvarea instanţelor de lucru făcându-se de către planificator, comandat, pentru a permite migrarea sarcinii. Câtă vreme resursa hardware suportă containerul de rulare, această sarcină poate fi reluată şi finalizată fără a pierde datele intermediare calculate aşa cum se întâmplă la migrarea prin pornirea aplicaţiei pe altă resursă şi eliminarea instanţei vechi - caz în care trebuie să se ţină

seama – la replanificare – și de overhead-ul implicat de acest proces.

V. Concluzii

Virtualizarea și administrarea automata a sarcinilor în sistemele distribuite sunt puncte cheie ale tehnologiei IT actuale. Lărgirea domeniului de aplicabilitate a soluțiilor existente la resurse eterogene poate creste ușurința de administrare și implementare a soluțiilor server/client pentru o gamă largă de dispozitive.

Virtualizarea simplifica managementul resurselor și permite scalarea mai flexibila a acestora pentru sarcini distribuite, în același timp acceptând oferirea de servicii specializate cu efort minimal și o mai bună separație a nivelelor de securitate decât tehnologiile clasice.

Mulțumiri

Această lucrare a beneficiat de suport financiar prin proiectul "Performanta sustenabila în cercetarea doctorala și post doctorala - PERFORM", Contract nr. POSDRU/159/1.5/S/138963", proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane2007-2013.

Bibliografie

[1] Hwang, J. Dongarra, G. Fox, „Virtualization: Physical vs. Virtual Clusters”, TechNet Magazine article, issue Apr. 2012.

[2] C. Li, A. Raghunathan, N.K. Jha, “Secure Virtual Machine Execution under an Untrusted Management OS”, in IEEE 3rd Internațional Conference on Cloud Computing (CLOUD), Miami (USA), pp. 172-179, 2010.

[3] R. N. Calheiros, M. Storch, E. Alexandre, C. A. F. De Rose, M. Breda, „Applying Virtualization and System Management in a Cluster to Implement an Automated Emulation Testbed for Grid Applications”, 20th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), IEEE Computer Society, pp. 97-104, 2008.

[4] O. Gherman, „Keeping Functional Legacy Applications on Grid and Cluster Systems”, The 3rd International Conference on Emerging Intelligent Data and Web Technologies (EIDWT-2012), Bucharest (Romania), 19-21 september, 2012.

[5] M. Harry, J. Miguel del Rosario, A. Choudhary, “VIP-FS: A Virtual, Parallel File System”, in Proceedings of 9th Internațional Parallel Processing Symposium, Santa Barbara (USA), pp. 159-164, 1995.

[6] I. Kim, T. Kim, Y. Ik Eom, “NHVM: Design and Implementation of Linux Server Virtual Machine Using Hybrid Virtualization Technology”, in Internațional Conference on Computational Science and Its Applications (ICCSA), Fukuoka (Japan), pp. 171-175, 2010.

[7] C. Y. Dong, X. Zhang, J. Dai, H. Guan, “HYVI: A HYbrid VIrtualization Solution Balancing Performance and Manageability”, IEEE Transactions on Parallel and Distributed Systems, issue 99, 2013.

[8] ****, „Red Hat Enterprise Linux – Virtualization Guide v5.1”, Red Hat technical document, online: http://www.centos.org/docs/5/html/5.2/Virtualization/, 2008.

[9] D. Quintero, L. Roșca, M. Vanous, R. Wale, J. Yoshino, O. Lascu, „Virtualization and Clustering Best Practices Using IBM System p Servers”, IBM Redbooks, online resource at www.redbooks.ibm.com/abstracts/sg247349.html?Open, 2014.

[10] T. Deshane, P.F. Wilbur, „Xen Hypervisor Deployment, Management, and Cloud Computing Tools”, Clarkson University, online resource at http://cosi.clarkson.edu/docs/installingxen/2010/Fall_2010_LISA_ Xen_Class.pdf, 2010.

[11] F. Bellard, „QEMU, a Fast and Portable Dynamic Translator”, online resource at http://static.usenix.org/event/usenix05/tech/freenix/full_papers/bellard/bellard.pdf, 2005.

[12] H. Nakada, T. Yokoi, T. Ebara, Y. Tanimura, H. Ogawa, S. Sekiguchi, „The Design and Implementation of a VirtualCluster Management System”, Proceedings of 1st IEEE/IFIP International Workshop on End-to-End Virtualization and Grid Management, pp. 61–71, 2007.