Um Framework Baseado em Virtualização para Emulação de Sistemas Distribuídos*

Rodrigo N. Calheiros[1], **César A. F. De Rose**[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681 Telefone: 3320-3500
rcalheiros@inf.pucrs.br, cesar.derose@pucrs.br

1. Introdução

Apesar da ubiqüidade das redes de computadores ter habilitado a ampla utilização de sistemas distribuídos, as técnicas atuais para avaliar estes sistemas, como simulação e emulação, possuem limitações tanto em sua aplicabilidade quanto na confiabilidade dos resultados gerados. Este artigo apresenta uma solução para este problema: um *framework* para emulação de sistemas distribuídos baseado em virtualização. Este *framework* utiliza o Xen [Barham et al. 2003] para acessar um conjunto de máquinas virtuais que compõem um sistema distribuído virtual. Cada máquina real (*host*) de um *cluster* pode hospedar uma ou mais destas máquinas virtuais (*guests*), que são conectadas por uma rede virtual cujas características são definidas pelo usuário do *framework*.

Desta forma, um *cluster* executando Xen torna-se uma plataforma de testes para sistemas distribuídos que permite que experimentos sejam replicados e testados em cenários emulados que dificilmente poderiam ser obtidos em ambientes reais. Os serviços implementados pelo *framework* são os seguintes:

Mapeamento. Durante o mapeamento, as máquinas que compõem o sistema virtual (cuja descrição é fornecida pelo usuário como um arquivo XML de entrada) são mapeadas no sistema real (o *cluster* cuja descrição, fornecida através de um arquivo XML, é conhecida pelo *framework*). O algoritmo de mapeamento utilizado atualmente procura minimizar o número de *hosts* utilizados por um usuário. Ao mesmo tempo, ele procura colocar *guests* que se comunicam no mesmo *host*, minimizando o tráfego de dados pela rede do *cluster*.

Deployment. Na etapa de *deployment*, os *guests* são criados nos *hosts* de acordo com a saída gerada na etapa de mapeamento, utilizando-se imagens de máquinas virtuais previamente geradas e armazenadas. Atualmente, o *framework* é capaz de fazer o *deployment* automaticamente ou de utilizar a ferramenta XSM [Franciosi et al. 2007] para esta tarefa.

Gerência de Rede. Após os *guests* serem criados, as conexões entre eles precisam ser configuradas, a fim de que seja implantado no sistema virtual o comportamento de um sistema distribuído requerido pelo usuário do *framework*. Atualmente, isto é feito com a manipulação do *firewall* no `dom0` de cada *host*, acessado via SNMP [Stallings 1999].

Após a execução destas três etapas, o sistema distribuído virtual está pronto para receber as aplicações que executarão no ambiente emulado. A descrição destas

*Este trabalho foi desenvolvido em colaboração com a HP Brasil P&D.

Tabela 1. Tempo médio para transferência de arquivos na grade virtual.

	com limite de banda	sem limite de banda
Upload	136,34s	5,19s
Download	112,16s	3,46s

aplicações (experimentos) é passada ao *framework* como um arquivo XML, e executada por um módulo de gerência de experimentos. Este módulo também recebe comandos dos usuários, tais como: interrupção e retomada de um experimento, cancelamento de um experimento e resgate de arquivos gerados pelo experimento.

2. Avaliação

Para a avaliação do sistema, foi utilizado um *cluster* de 8 máquinas Pentium 4 2.8GHz com 2,5GB de memória RAM. Cada nó do *cluster* possui o Xen 3.1, que utiliza 328MB de memória. Os 2232MB de memória restantes estavam disponíveis para serem alocados às máquinas virtuais. Sobre este *cluster*, desejava-se construir um ambiente composto de 3 *sites*, com 10, 13 e 9 máquinas, cada uma rodando Linux Debian Etch e utilizando 256MB de memória.

Durante a etapa de mapeamento, foram selecionadas 4 máquinas para abrigar o ambiente (o mínimo de *hosts* para hospedar os 32 *guests*). As máquinas virtuais foram então iniciadas na etapa de *deployment*. O ambiente, na etapa de gerência de rede, foi configurado para limitar o *link* entre os *sites* em 1Mbps.

Neste ambiente então foi instalada uma grade OurGrid [Cirne et al. 2006], onde cada um dos *sites* era um *site* OurGrid, executando o *peer* em uma das suas máquinas virtuais. Um *corepeer* foi instalado no *site* 1, a fim de permitir que os *sites* se descobrissem. Uma máquina virtual em cada *site* executava o escalonador *MyGrid* e disparava uma aplicação contendo 20 tarefas que realizavam 2MB de transferência de arquivos (*upload* de 1MB antes da execução da tarefa e *download* de 1MB após a execução) e realizavam um *sleep* de 10 segundos. As demais máquinas eram *GuMs* do OurGrid, e executavam as aplicações. O escalonador só utilizava recursos de *sites* remotos.

O experimento foi repetido posteriormente sem o limite de banda, e os resultados apresentados na Tabela 1 sugerem que o controle de banda é efetivo. Como trabalhos futuros, estamos investigando e formalizando o problema de mapeamento aplicado no *framework*. Uma interface gráfica também está em desenvolvimento e os aspectos relacionados ao controle de aplicações executando no ambiente virtual estão sendo estudados.

Referências

Barham, P. et al. (2003). Xen and the art of virtualization. In *Proc. of the 19th ACM Symposium on Operating Systems Principles*.

Cirne, W. et al. (2006). Labs of the world, unite!!! *Journal of Grid Computing*, 4(3):225–246.

Franciosi, F. et al. (2007). Deploying and managing Xen sites with XSM. In *Workshop on Virtualization/Xen in HPC Cluster and Grid Computing Environments*.

Stallings, W. (1999). *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*. Addison Wesley, Reading, 3^a edição.