MV6 – UM MECANISMO DE TRANSIÇÃO BASEADO EM MÁQUINAS VIRTUAIS

Arthur Bispo de Castro[1*], Cleymone Ribeiro dos Santos[1], Paulo Lício de Geus[1]

[1]Laboratório de Administração e Segurança de Sistemas
Instituto de Computação – Universidade Estadual de Campinas
Caixa Postal 6176 – 13.083-970 – Campinas, SP – Brasil

{arthur,cleymone,paulo}@las.ic.unicamp.br

Abstract. *The IETF has developed tools and mechanisms that permit IPv4 networks to migrate smoothly to IPv6, enabling interoperability between the two protocols. However, several of these mechanisms are disadvantageous in certain scenarios and thus are not recommended to be used. To overcome such problems, this work proposes a new transition model based on virtual machine, called MV6. MV6 offers protection to the system resources via the provision of a robust level of security, and allows systems development without affecting the normal operations of the system.*

Resumo. *O IETF desenvolveu ferramentas e mecanismos, que permitem a redes e máquinas IPv4 migrarem suavemente para IPv6 assim como permitem a interoperabilidade entre os dois protocolos. Entretanto, os vários mecanismos definidos tem desvantagens que não os recomendam em alguns cenários. Desta forma, propõe-se a utilização de um novo mecanismo de transição baseado em máquinas virtuais, chamado MV6. MV6 apresenta características de proteção aos recursos do sistema, já que fornece um nível robusto de segurança, e de permissão do desenvolvimento de sistemas sem atrapalhar as operações normais.*

1. Introdução

O IPv4, versão atual do protocolo IP, não antecipou e, conseqüentemente, não contempla as necessidades atuais da Internet. O problema mais evidente é o crescimento exponencial da Internet e resultante ameaça de exaustão do espaço de endereçamento IPv4. Logo, técnicas como NAT e CIDR foram desenvolvidas para resolver paliativamente este problema.

Aliado ao problema de endereçamento IP, a necessidade de configuração simplificada, a ausência de mecanismos de segurança mais robustos e a necessidade de suporte melhorado para entrega de dados em tempo-real, estimularam o *Internet Engineering Task Force* (IETF) a desenvolver um conjunto de protocolos e padrões conhecido como IP versão 6 (IPv6). Conseqüentemente, o protocolo IPv6, nos últimos anos, vem adquirindo um grau de importância cada vez maior mundialmente, principalmente em países asiáticos e europeus, cuja falta de endereços IPv4 já é bastante evidente.

* Apoio Capes.

Como o conceito de telecomunicações está cada vez mais voltado à convergência das redes e ao "mundo IP", e tendo em vista que o protocolo IPv4 é o principal ator nas infra-estruturas das redes atuais, é bastante coerente que a migração para o protocolo IPv6 seja realizada de uma forma metódica e progressiva.

Em decorrência desta necessidade, o IETF criou o Grupo de Trabalho *Next Generation Transition* (NGTRANS), cujo objetivo era desenvolver ferramentas e mecanismos que permitam que redes e máquinas IPv4 migrem suavemente para IPv6, já que os cabeçalhos dos protocolos IPv4 e IPv6 não são interoperáveis entre si. Durante o período de dois anos, o Grupo NGTRANS criou e disponibilizou inúmeros esboços (*drafts*), que estabeleceram uma miríade de mecanismos para a integração com IPv6, que vão desde um simples tunelamento a complexos mecanismos como Teredo [Huitema,02]. Entretanto, estes vários mecanismos tem desvantagens que não os recomendam em cenários onde: (1) aplicações críticas rodam em IPv4 e precisam iniciar experiência com redes IPv6, (2) redes IPv6 nativas rodam e aplicações não portadas para IPv6 precisam rodar. Desta forma, propomos a utilização de um novo mecanismo de transição baseado em máquinas virtuais, chamado MV6, MV6 apresenta características de proteção aos recursos do sistema, já que fornece um nível robusto de segurança, e de permissão do desenvolvimento de sistemas sem atrapalhar as operações normais do sistema.

O objetivo deste artigo é apresentar este mecanismo de transição baseado em máquinas virtuais. Na seção 2 são explicados os mecanismos de transição disponíveis atualmente. A seguir, na seção 3 é explicitado o que é uma máquina virtual e seus vários tipos. Então, a seção 4 apresenta o modelo de transição proposto, MV6, sendo que na seção 5 os resultados são detalhados. Finalmente, a seção 6 apresenta as conclusões gerais do artigo.

2. Mecanismos para a transição de redes IPv4 para redes IPv6

Os mecanismos de transição das redes IPv4 atuais para as novas redes IPv6 são divididos em três categorias principais: pilha dupla (*dual-stack*), tunelamento e tradução. [Schild and Strauf, 03]

O mecanismo de pilha dupla permite que elementos de rede (máquinas, roteadores etc) implementem e executem as pilhas de protocolos IPv4 e IPv6 simultaneamente, criando assim uma rede IPv6 sobre a infra-estrutura IPv4 já existente. Elementos de rede implementando pilha dupla terão dois endereços de rede (um para cada protocolo, para cada interface). A figura 1 ilustra tal mecanismo [Gilligan and Nordmark, 00].

Pilha dupla é um mecanismo flexível e fácil de usar. Contudo, ele tem as seguintes desvantagens: cada máquina precisa ter as duas pilhas rodando separadamente, o que demanda poder de processamento adicional e memória, assim como tabelas de roteamento para os dois protocolos. Um resolver DNS também precisa ser capaz de resolver ambos os tipos de endereços. Geralmente, todas as aplicações rodando na máquina pilha dupla devem ser capazes de determinar se a máquina está se comunicando com uma máquina IPv4 ou IPv6.

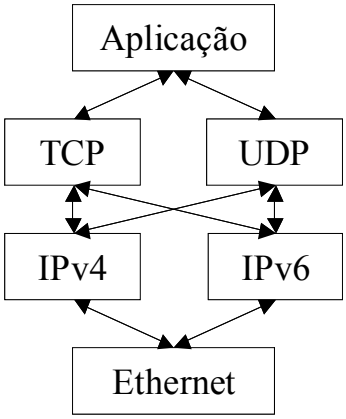

Figura 1: Mecanismo de transição "pilha dupla"

O mecanismo de tunelamento permite que duas redes IPv6 distintas, sejam nativas ou baseadas em pilha dupla, se comuniquem utilizando uma rede IPv4 como se fosse um enlace conectando o roteador de saída de uma rede ao roteador de entrada da outra. Existem tunelamentos que tratam a rede IPv4 como um enlace virtual ponto-a-ponto, e também aqueles que tratam como um enlace virtual NBMA (*Non Broadcast Multi-Access*). Exemplos deste mecanismo são: tunelamento manual [Gilligan and Nordmark, 00], tunelamento automático 6to4 [Carpenter and Moore, 01], tunelamento automático ISATAP (*Intra-Site Automatic Tunnel Address Translation*) [Templin et. al, 02], dentre outros.

Tunelamento tem várias desvantagens: (1) a carga adicional colocada no roteador, já que cada ponto de entrada e de saída precisa de tempo e poder de CPU para encapsular e desencapsular pacotes, (2) os pontos de entrada e saída representam pontos únicos de falha, (3) a solução de problemas se torna mais complexa ao entrar em detalhes de *hop count* e MTU, assim como problemas de fragmentação. Um exemplo deste mecanismo é mostrado na figura 2:

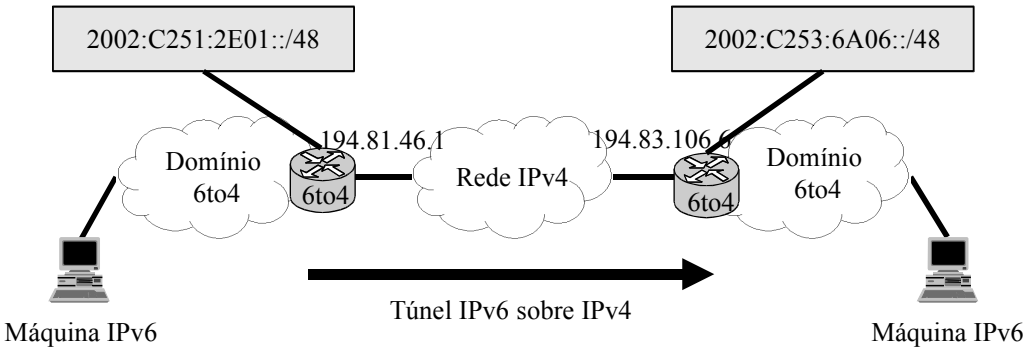

Figura 2: Mecanismo de transição "tunelamento"

Por fim, o mecanismo de tradução é utilizado para comunicação entre nós IPv6 e IPv4. Exemplos de tal mecanismo são o SIIT (*Stateless IP/ICMP Translation*), que não mantém o estado das conexões criadas entre origem e destino, e o NAT-PT (*Network Address Translation - Protocol Translation*), que mantém os mapeamentos entre endereços IPv4 e IPv6 das conexões. O mecanismo de tradução apresenta como vantagem

permitir que máquinas IPv6 se comuniquem diretamente com máquinas IPv4, porém suas desvantagens são: não suporta características avançadas de IPv6 como segurança fim-a-fim, impõe limitações à topologia da rede, pois as respostas de qualquer mensagem enviada pelo roteador de tradução devem retornar para o mesmo roteador de tradução, além do roteador de tradução ser um ponto único de falha. Este mecanismo é recomendável somente quando nenhum outro mecanismo for possível e deve ser visto como uma solução temporária até outra qualquer ser implementada.

3. Máquinas Virtuais

A idéia de máquinas virtuais não é uma novidade. Desde o início da computação ela vem sendo utilizada para estender o multiprocessamento, a multi-programação e o multi-acesso, tornando os sistemas também multi-ambiente [Goldberg, 73]. O enorme crescimento do poder de processamento dos computadores pessoais proporcionou uma expansão em seu uso, permitindo que tanto pequenas empresas quanto usuários domésticos possam se valer das suas vantagens.

De acordo com [Silberschatz, 02] existem duas principais vantagens no uso de máquinas virtuais. Primeiro, para proteger os recursos do sistema, fornecendo um nível robusto de segurança. Segundo, as máquinas virtuais permitem que o desenvolvimento de sistemas possa ser feito sem atrapalhar as operações normais do sistema.

Cada máquina virtual é completamente isolada das outras e, sendo assim, não existem problemas de segurança, com os recursos do sistema completamente protegidos. Por exemplo, aplicações não confiáveis podem ser obtidas na Internet e executadas cada uma em uma máquina virtual separada. Todavia, esse mecanismo possui uma desvantagem, não existe compartilhamento direto de recursos entre máquinas virtuais distintas.

Normalmente, modificar um sistema operacional é uma tarefa difícil. Os sistemas operacionais são programas grandes e complexos, sendo que uma modificação numa parte pode causar problemas estranhos em outras partes. Usando máquinas virtuais, cada desenvolvedor pode ter sua própria máquina, e os danos que poderiam ser causados ao sistema se tornam inócuos.

O uso de máquinas virtuais introduz um maior nível de segurança se valendo de três pontos principais, o isolamento do usuário administrador da máquina virtual, pela adição de mais uma camada de proteção para acesso a máquina real e por sua fácil restauração em caso de problemas. É importante frisar que, com isso, não serão eliminadas as vulnerabilidades presentes nas aplicações. Elas ainda poderão ser exploradas por um atacante, porém com implicações bem menores para a segurança do sistema.

Como existem aplicações que, por questões de implementação, precisam ser executadas com as permissões do usuário administrador, o comprometimento destas expõe inteiramente o sistema operacional ao atacante. Se esta aplicação for executada, ainda como administrador, numa máquina virtual, as permissões que o possível atacante conseguiria seriam o do usuário comum que executou a máquina virtual atacada.

A máquina virtual cria uma dupla camada de isolamento entre as aplicações que estarão sendo executadas no sistema operacional convidado e o sistema operacional hospedeiro. Assim, um atacante, ao obter sucesso em subverter a aplicação e dominar o sistema convidado, tem ainda que comprometer o *software* da máquina virtual e, em se-

guida, obter privilégios de administrador a partir de uma conta de usuário não privilegiado, para enfim atingir a máquina real.

O uso de máquinas virtuais confere algumas características adicionais à segurança, além do isolamento apresentado. Considerando a utilização de arquivos na máquina real como partições de disco virtual, é possível restaurar facilmente o sistema operacional convidado após uma invasão, de duas formas. A primeira consiste em realizar previamente uma cópia de *backup* do arquivo da máquina real correspondente a partição virtual e, após o comprometimento do sistema, restaurar o arquivo. Outra forma é implementar não-persistência no disco virtual, de tal forma que as modificações realizadas sejam registradas em um arquivo a parte. Desta forma, ao reinicializar a máquina virtual, o registro de modificações pode ou não ser efetivado no disco virtual, permitindo de forma simples restaurar a partição ao estado original após uma invasão. Obviamente, também é possível implementar a não-persistência em uma máquina real, porém se faz necessário modificar o sistema de arquivos para suportar essa operação, enquanto na máquina virtual essa operação pode ser realizada pelo *software* da máquina virtual, sem necessitar modificar o sistema de arquivos da máquina virtual.

A simulação de uma máquina é feita a partir da criação de um arcabouço para que um sistema operacional, agora chamado de sistema convidado, possa ser executado como uma aplicação de um outro sistema, chamado sistema anfitrião ou hospedeiro. Este arcabouço é criado a partir de uma aplicação denominada VMM (*Virtual Machine Monitor* - Monitor de Máquinas Virtuais), que é responsável por simular os recursos para a utilização das máquinas virtuais e traduzir suas requisições para a máquina real. Essa indireção adiciona às máquinas virtuais uma grande vantagem na realização de testes durante o desenvolvimento do sistema operacional, pois torna possível emular uma plataforma computacional com recursos distintos do da máquina real, variando em termos de memória, número de processadores, periféricos, discos rígidos, entre outros. Essa emulação também é possível num sistema operacional convencional, porém neste faz-se necessário modificar os módulos envolvidos na operação, podendo interferir nos resultados obtidos, enquanto no caso do sistema operacional convidado a emulação será tratada pela infra-estrutura da máquina virtual.

Para compreender melhor como uma máquina virtual funciona, faz-se necessário uma análise paralela de como funciona um sistema operacional. A principal finalidade de um sistema operacional é o gerenciamento de recursos a serem disponibilizados às aplicações que deles necessitarem. Para tal, apenas o sistema operacional pode fazer acesso direto aos recursos, relegando às aplicações o acesso indireto, através de uma interface bem definida, provida pelo sistema operacional. Essa interface é conhecida como *API* (*Application Programming Interface* - Interface de Programação das Aplicações). Ao receber uma chamada pela *API*, o núcleo do sistema operacional fica responsável por resolver eventuais concorrências por recursos, e executar as operações requisitadas através da camada dependente do *hardware*, que converte os comandos de alto nível do sistema operacional em comandos de baixo nível específicos aos periféricos acionados. O sistema operacional, portanto, pode ser representado como na Figura 3.

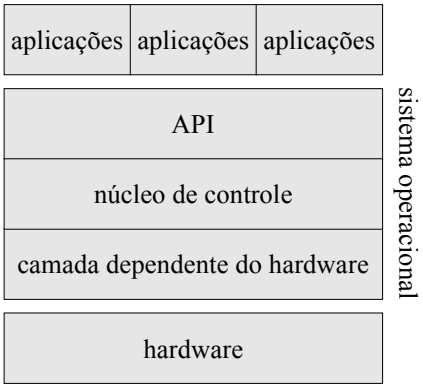

Figura 3: Estrutura de um sistema operacional

Partindo dessa estrutura simplificada de um sistema operacional, com a substituição da camada dependente de *hardware* do sistema convidado por outra camada que possua a mesma interface com o núcleo, porém que utilize chamadas da *API* do sistema operacional anfitrião, Figura 4, ou a emulação da saída da camada dependente de *hardware* do sistema convidado para a uma chamada da *API* do sistema anfitrião, Figura 5, é possível executar o sistema operacional como uma aplicação de outro sistema operacional.

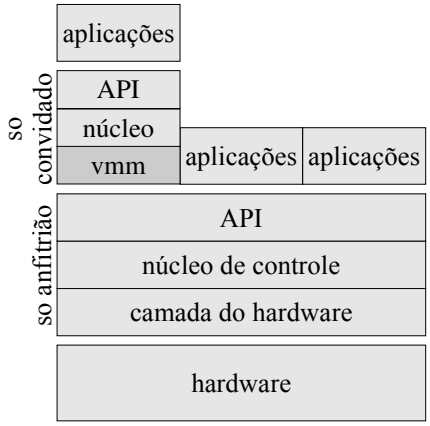

Figura 4: Substituição da camada dependente do hardware do sistema convidado

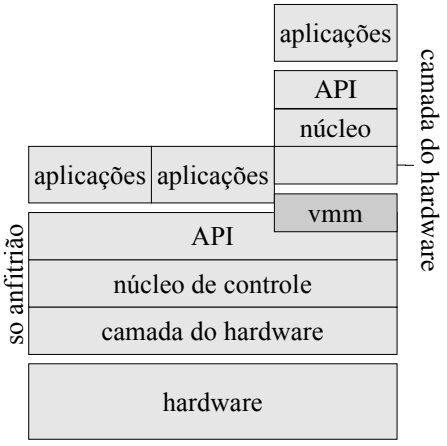

Figura 5: Emulação da saída da camada dependente do hardware para a API do sistema anfitrião

Desta forma, define-se máquina virtual como sendo uma abstração em *software* de um *hardware*, de tal maneira que ele seja executado sobre outro sistema operacional e permita execuções de outras aplicações, inclusive incompatíveis com o sistema hospedeiro, sem que estas possam identificar que estão em um sistema convidado. E, em se tratando de uma aplicação de um sistema, as outras aplicações do sistema hospedeiro não são prejudicadas pela execução de uma máquina virtual. Com isso, vale a pena ressaltar que uma máquina virtual consiste de "duplicata eficiente e isolada de uma máquina real." [Popek, 74].

Apesar de eficientes, as operações privilegiadas podem sofrer aumento no tempo de execução, em decorrência da necessidade de tradução para a *API* do sistema operacional hospedeiro. Testes realizados ([UML, 99] e [VMWare, 99]) indicam que, dependendo da aplicação, o custo da virtualização fica, na média, entre 15% a 30%. Uma das operações que sofre maior impacto neste processo é o acesso ao disco. Como a máquina virtual não deve ter acesso às mesmas partições da máquina real, por questões de segurança, várias verificações adicionais têm que ser realizadas pela *VMM* para garantir o isolamento. Entretanto, esse processo introduz uma interessante vantagem ao uso de máquinas virtuais: como o acesso a disco é emulado, é possível utilizar um arquivo no disco da máquina real ao invés de dispor de uma partição exclusivamente para a máquina virtual. Dessa forma, a criação de máquinas virtuais pode ser realizada a partir de uma instalação única, e o arquivo correspondente ao disco virtual pode ser copiado gerando outra instalação completa, de forma rápida.

Existem várias alternativas interessantes de máquinas virtuais, e para este estudo escolheu-se o uso de duas. A primeira é um gerenciador de código livre, que substitui a camada dependente de hardware do *kernel* do sistema operacional Linux, chamado User-Mode Linux (UML) [UML, 99]. O UML é uma forma segura de rodar novas versões de Linux sem comprometer a instalação e a configuração da máquina real Linux. Uma outra alternativa, desta vez de software proprietário, é o VMWare [VMWare, 99]. Este permite a instalação de vários tipos de sistemas operacionais, sem necessidade de nenhuma alteração no sistema operacional convidado, permitindo o uso da mídia oficial do fabricante. Ele virtualiza completamente o hardware de um computador, inclusive com uma BIOS virtual, convertendo as chamadas que seriam para o hardware para as chamadas de sistemas do sistema operacional hospedeiro. Os prováveis usos para máquinas virtuais são construir redes complexas em um único computador; desenvolver, configurar, testar e implantar novas aplicações sem risco; adicionar e mudar sistemas operacionais sem precisar reparticionar o disco ou reiniciar a máquina; rodar novos sistemas operacionais e aplicações legadas em uma única máquina.

4. Mecanismo de Transição Proposto

O mecanismo de transição proposto, chamado MV6, é baseado na utilização de máquinas virtuais para a criação de um ambiente onde os dois protocolos IPv4 e IPv6 coexistam de forma independente. A idéia básica é formar duas redes independentes e paralelas, Figura 6, uma rodando IPv4 e a outra IPv6, sendo que a rede chamada nativa será a rede do sistema operacional hospedeiro e a rede paralela estará na máquina virtual. É pré-requisito obrigatório que estas redes rodem protocolos distintos.

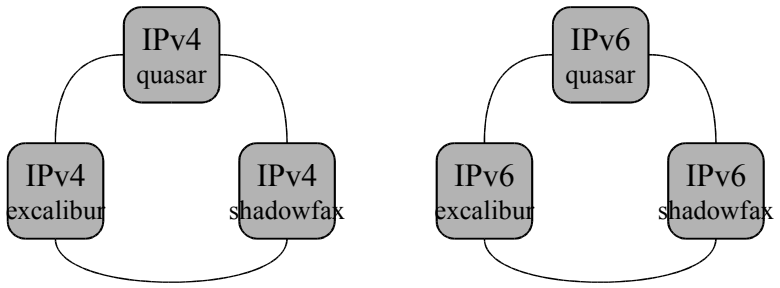

Figura 6: Representação da rede IPv4 e IPv6, que coexistem paralela e independentemente

As principais vantagens do uso de máquinas virtuais no novo mecanismo são: eliminar a necessidade de portar aplicações legadas, testar o novo protocolo em máquinas virtuais antes da implantação, suportar aplicações legadas durante a migração para o novo protocolo além de minimizar o sofrimento dos usuários durante a transição de IPv4 para IPv6.

Existem dois cenários para exemplificar a utilização e aplicação deste mecanismo de transição baseado em máquinas virtuais.

4.1.Cenários de Aplicação

Dois cenários de aplicação deste mecanismo de transição foram mapeados e serão explicados em detalhes. O primeiro cenário tem o protocolo IPv4 como protocolo nativo da rede, ou seja, o IPv4 é o protocolo utilizado desde a formação da rede e com o qual os usuários estão ambientados. Além disto, nesta rede existem aplicações críticas que rodam em IPv4 e que não podem ser comprometidas por nenhum problema na rede. Entretanto, nesta rede os administradores querem se familiarizar com os problemas advindos do uso constante de IPv6 ao mesmo tempo em que criam a cultura do novo protocolo em seus usuários, sem contudo comprometer o funcionamento da aplicação crítica IPv4. Objetivando conciliar estes requisitos contrários, é criado via máquinas virtuais uma rede IPv6 paralela à rede IPv4, que permite a existência da rede IPv4 e o funcionamento de suas aplicações críticas independentemente da existência e do funcionamento da rede IPv6 (Figura 7). Por fim, este cenário é útil quando é necessário ter contato com IPv6 sem interferência no ambiente de trabalho corrente, que se utiliza de IPv4.

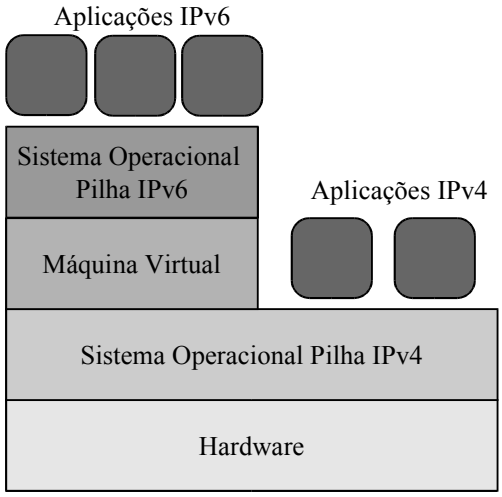

Figura 7: estrutura duma máquina virtual com pilha IPv6.

O segundo cenário tem o protocolo IPv6 como protocolo nativo da rede, já que está rede migrou para IPv6. No entanto, algumas aplicações ainda não foram portadas para o novo protocolo, precisando assim do protocolo IPv4 para funcionarem corretamente. Este cenário busca suprir esta necessidade ao criar uma rede paralela IPv4 à rede nativa IPv6, que permite a coexistência pacífica entre os dois tipos de rede. O usuário utiliza o protocolo IPv6 nativamente e quando precisa acessar uma aplicação que ainda

não foi portada para IPv6, ele usa a máquina virtual com rede IPv4 para, de forma paralela, acessar a aplicação IPv4 (Figura 8).

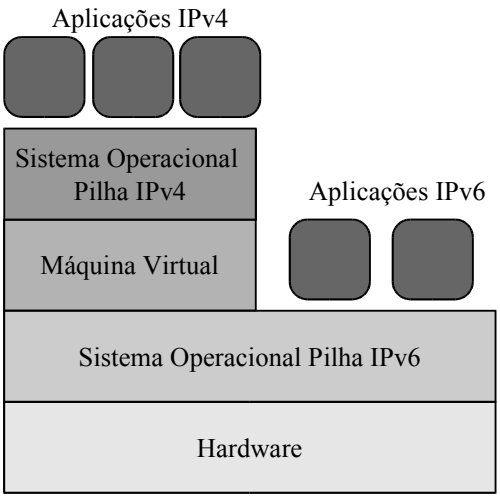

Figura 8: estrutura duma máquina virtual com pilha IPv4.

5. Resultados

Os estudos sobre utilização de máquinas virtuais, ligados ao laboratório em que os autores fazem parte, abrangeram outras áreas que não somente o uso em IPv6. Em principio, seu uso estava ligado à execução de aplicações de outros sistemas operacionais, ou auxiliando na migração e teste de compatibilidade de algumas aplicações entre sistemas distintos. Para isso a máquina virtual mais utilizada era o VMWare, que possibilita que seja executado Windows dentro do sistema operacional Linux, e vice-versa.

Um outro estudo anterior aos mecanismos de transição para IPv6 é o de virtualização de servidores. Com este estudo, muitas das dificuldades que iriam surgir com a proposta de MV6 foram facilmente contornáveis. Para entender sobre que aspectos a experiência adquirida com a virtualização de servidores ajudou nesse processo, torna-se necessário explicar um pouco sobre ela.

A virtualização de servidores está sendo implementada em um laboratório de redes, cuja topologia consistia de um *bastion*, um servidor interno, e um filtro isolando a *DMZ*[a] da rede interna e da Internet (Figura 9). Essa topologia é uma variação da arquitetura *screened subnet*, como descrito em [Nakamura, 03]. O objetivo final é substituir o esquema apresentado por uma máquina real fazendo o papel de filtro, de hospedeiro das máquinas virtuais e de servidor de arquivos.

a DMZ – zona desmilitarizada (*Demilitarized Zone*), uma rede que tem por objetivo isolar a rede interna e a Internet de conexões diretas.

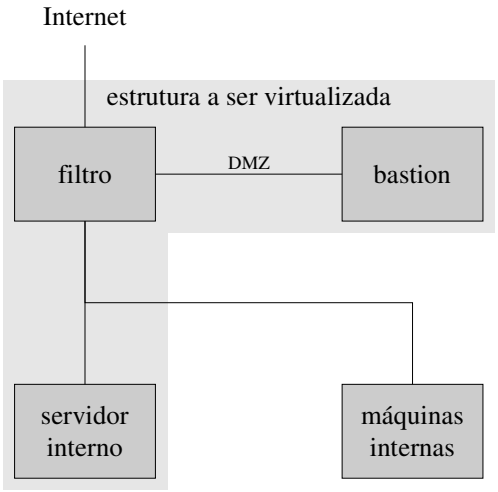

Figura 9: Estrutura básica da topologia da rede antes da inclusão de máquinas virtuais

As máquinas da rede original utilizavam Linux como sistema operacional, e o uso da máquina virtual UML tornou-se a escolha natural. A implantação contou com problemas não previsíveis. O módulo responsável pela interface de rede virtual da máquina real, depois de intensa atividade, deixava todas as interfaces de rede do sistema operacional anfitrião sem funcionamento. O problema, entretanto, não estava ligado a máquina virtual, e sim a inclusão do suporte a IPv6 no *kernel* do sistema Linux. Este problema foi resolvido, primeiramente com a desativação do suporte a IPv6 e, depois, com uma nova versão do *kernel*, com problema corrigido.

Os outros tipos de problemas estão relacionados ao aprendizado, normal, exigido por qualquer outro tipo de programa. Especificamente neste caso, em como compilar corretamente um *kernel* do Linux e de que maneira e qual sistema operacional deve dever instalado.

Desta forma, com a experiência adquirida, o desafio de montagem e configuração da máquina virtual se restringia ao mesmo tipo de problema que instalar e configurar IPv6 numa máquina real.

O primeiro passo ao instalar uma máquina virtual *UML* está na escolha e compilação do *kernel* do sistema convidado. Neste caso foi utilizado o *kernel* 2.4.23 com a atualização do *UML* versão 2.4.23-2. O *kernel* foi compilado para não ter suporte a módulos e com a opção de utilizar a pilha IPv6 habilitada.

O segundo passo foi preparar o arquivo que representa o disco da máquina virtual. Neste ponto, deve-se criar um arquivo esparso do tamanho desejado, 2 Gb neste caso, e formata-lo com algum sistema de arquivos de preferência. Este arquivo deve ser montado em algum diretório e nele instalado os pacotes do sistema operacional, neste caso foi utilizado o Gentoo Linux [Gentoo, 01]. Terminada a instalação e a configuração do sistema, o diretório pode ser desmontado e a máquina virtual pode ser executada.

6. Conclusão

O uso de máquinas virtuais torna-se uma alternativa bastante interessante em muitos casos. Particularmente, o fato da máquina virtual já ser historicamente usada para permitir execução de aplicativos de outros sistemas operacionais ou até mesmo outras arquiteturas, acabou fazendo com que esta proposta alternativa de migração para IPv6 tenha um bom embasamento, tanto técnico, pelo estudo dos autores deste, quanto histórico, dado as outras aplicações semelhantes de máquinas virtuais.

Os testes realizados no laboratório de redes serviram para confirmar o fato citado, indicando que, a migração pode ser feita de forma segura e simples. Os problemas, que aconteceram durante os testes, não foram somente de questão técnica, mas também referentes a recente implementação do suporte a IPv6 no *kernel* do sistema Linux. O mecanismo de transição proposto mostra ser uma proposta que se adequará corretamente ao processo metódico de migração da conectividade de rede IPv4 para IPv6 bem como ao funcionamento de aplicações IPv4 que não migrarão para IPv6.

Referências

C. Huitema (2002). Teredo: Tunneling IPv6 over UDP through NATs. http://www.ietf.org/proceedings/02mar/I-D/draft-ietf-ngtrans-shipworm-05.txt.

C. Schild and T. Strauf (2003). Initial IPv4 to IPv6 transition cookbook for end site networks/universities. http://www.6net.org/publications.

R. Gilligan and E. Nordmark (2000). Request for Comments 2893. Transition Mechanisms for IPv6 Hosts and Routers. http://www.ietf.org/rfc/rfc2893.txt.

B. Carpenter and K. Moore (2001). Request for Comments 3056. Connection of IPv6 Domains via IPv4 Clouds. http://www.ietf.org/rfc/rfc3056.txt.

F. Templin and T. Gleeson and M. Talwar and D. Thaler (2002). Intra-Site Automatic Tunnel Addressing Protocol (ISATAP). http://www.ietf.org/proceedings/02mar/I-D/draft-ietf-ngtrans-isatap-03.txt.

P. R. Goldberg (1973). Architecture of virtual machines. Proceedings of AFIPS National computer conference, New York - NY - USA.

Peter B. Silberschatz and Peter B. Galvin and Greg Gagne (2002). Operating System Concepts. John Wiley & Sons, Inc., New York, 6th edition.

Gerald J. Popek and Robert P. Goldberg (1974). Formal requirements for virtualizable third generation architectures. ACM Press, 17 (7): 412-421.

Jeff Dike (1999). The User-mode Linux kernel HomePage. http://user-mode-linux.sourceforge.net/.

VMWare Inc. (1999). VMWare is Virtual Infrastructure. http://www.vmware.com/.

E. Nakamura and P. L. de Geus (2003). Segurança de Redes em ambientes cooperativos. Editora Futura, São Paulo, Second edition.

Gentoo Technologies Inc. (2001). Gentoo Linux. http://www.gentoo.org/.