Uso do Protocolo IPv6 e de Multicasting para Transmissão de Vídeo

LUCIANO MARTINS[1]
EDSON DOS SANTOS MOREIRA[1]

[1]ICMC- Instituto de Ciências Matemática e de Computação
Universidade de São Paulo
Av. do Trabalhador São-Carlense, 400 - Centro - Cx. Postal 668
São Carlos - São Paulo - Brasil CEP 13560-970
{lumart, edson}@icmc.sc.usp.br

Introdução

Atualmente está ocorrendo uma corrida por novas tecnologias e serviços, ocasionada por diversos fatores, sendo um dos mais importantes a demanda dos usuários por uma infra-estrutura mais poderosa, que pudesse atender às novas aplicações.

O uso da técnica de *multicasting* permite que a largura de banda utilizada pelas aplicações seja economizada e que servidores multimídia economize recursos de sistema.

Um aumento na largura de banda também faz-se necessário, e diversas tecnologias de rede de alta velocidade estão disponíveis atualmente, tais como: *Fast Ethernet* de 100 Mbps, *GigaBit Ethernet* de 1000 Mbps, ATM (*Asynchronous Transfer Mode*) de 155 Mbps e 622 Mbps, FDDI (*Fiber Distributed Data Interface*) de 100 Mbps e futuramente o *10 Gigabit Ethernet* com 10 Gbps.

O protocolo de transporte IPv6 pode fornecer um maior desempenho nas redes, possibilidade de escalabilidade e mecanismos de segurança.

O trabalho tem como objetivos comparar o uso do protocolo IPv6 e da técnica de *multicasting* em sistemas Linux em relação ao protocolo IPv4 e à *unicasting*. A configuração de um sistema com suporte a IPv6 e *multicasting* foi realizada e uma aplicação *client/server*, desenvolvida em JAVA, foi desenvolvida para os testes.

Foram utilizadas tecnologias atuais abertas (gratuitas), como JAVA e *Linux*, APIs (*Applications Program Interfaces*) como JMF (*Java Media Framework*) e JNI (*Java Native Interface*), em conjunto com a ferramenta JIPSY (*Java IP version Six readY*). Os aplicativos *IPTraf e NetPerf*, para análise de desempenho de rede, foram utilizados para as medidas comparativas na transmissão do vídeo em uma rede *Fast Ethernet* (100Mbps).

1. Protocolo IPv6

IPv6 é a nova geração do protocolo IP (versão 4), que tem como principais objetivos o aumento do número de endereços e prover melhores desempenho e segurança na transmissão de pacotes nas redes de computadores. Muitas mudanças ocorreram no protocolo IPv6 em relação ao IPv4, como aumento do número de bits do endereço, utilização de *headers* de extensão e de rotulamentos de fluxo, autenticação, privacidade, e outros.

Os endereçamentos *unicast* e *multicast* do protocolo IPv4 basicamente continuam imutáveis na versão IPv6, porém o espaço de endereço de 32 bits passou para 128 bits e o endereçamento *broadcast* foi excluído e um novo endereço foi inserido: o *anycast*.

O *header* do IPv6 possui um tamanho de 40 bytes fixos, diferentemente do cabeçalho do IPv4, que varia de 20 a 60 bytes, o que dificulta o processamento dos pacotes nos roteadores

A forma mais usual é: X:X:X:X:X:X:X:X, sendo que cada 'X' possui 4 dígitos hexadecimais, cada um com 4 bits.

Atualmente a maioria dos sistemas sistemas operacionais já suportam IPv6, entre eles Linux, Windows NT e Windows 2000 e FreeBSD. A linguagem Java possui uma biblioteca de classes responsáveis pela programação em rede, contendo *sockets unicast* (TCP e UDP) e

também *multicast* (UDP). Esta biblioteca é a java.net. Esta biblioteca não possui suporte para endereços IPv6, sendo necessário um recurso adicional, caso o programador queira utilizar endereços IPv6 em sua aplicação.

Para o sistema operacional Linux, um *kit* chamado JIPSY (*Java IP Six readY*), é necessário. Ele contém o código para a utilização de *sockets* com suporte a endereços IPv6, além de um *makefile* para a compilação dos códigos e geração de uma biblioteca de sistema compartilhada. Habilita uma aplicação escrita em JAVA comunicar-se através de endereços IPv6 e foi desenvolvido usando JNI (*Java Native Interface*), permitindo que programas Java interoperem com aplicações e bibliotecas escritas em outras linguagens, tais como C, C++ e Assembly.

2. Multicasting

O modo mais simples de transmissão de dados para múltiplos receptores é emitir uma cópia de dados para cada um, individualmente. Esta técnica é chamada de *unicasting*, e quanto maior o número de receptores, maior é a largura de banda utilizada e o número de sessões paralelas no servidor é limitado.

Multicasting possui como abordagem a emissão de dados para múltiplos receptores, com uma origem transmitindo somente uma cópia de dados e replicando-os nos *hops* finais. Os receptores, ao contrário do *broadcasting*, devem indicar aos roteadores finais que desejam receber dados de uma sessão *multicast*.

Os endereços IPv4 *multicast*, representados por números decimais através da classe D, vão de 224.0.0.0 até 239.255.255.255, sendo o endereço 224.0.0.0 reservado e o endereço 224.0.0.1 é permanentemente associado a todos os *host*s do grupo de uma rede local, incluído também roteadores participantes no IP *multicast*. Os 28 bits de ordem mais baixa formam o identificador de grupo *multicast*. Quando este endereço é mapeado para endereços Ethernet, apenas os 23 bits de ordem mais baixa são utilizados.

Os endereços IPv6 iniciam com o número hexadecimal FF, seguidos de 4 bits indicadores de tipo de grupo (temporário ou permanente), 4 bits indicativos de escopo do pacote e 112 bits identificando o grupo. Quando este endereço é mapeado para endereços Ethernet, apenas os 32 bits de ordem mais baixa são utilizados.

3. Ambiente de testes e resultados

O ambiente de testes utilizado no trabalho, ilustrado na **Figura 1,** possui quatro PCs (*Personal Computers*) com os nomes: *whitehorse* (servidor), *jamel* (cliente), *terereh* (cliente) e *campari* (medidor de tráfego) e um *hub* 10/100 Mbps de 8 portas da 3Com para conexão das máquinas. Este cenário é ilustrado na **Figura 7:**

Figura 1: Ambiente de rede do trabalho

3.1. Aplicação *Multicast e Unicast* desenvolvida para testes no ambiente

Foram desenvolvidas duas aplicações: uma *unicast* e outra *multicast*. A aplicação cliente não apresenta o vídeo recebido ("*dumb*") e suas funcionalidades são:

Servidor: Possibilidade de receber conexão de mais de um cliente; Emissão de *frames* de vídeo usando sockets *multicast(*ou *unicast)*; Possibilidade de escolha entre endereços IPv4 e IPv6; Tratamento de datagramas UDP para os casos de desordenação e perda.
Cliente: União a grupos *multicast* (se for a aplicação *multicast*); Recepção de *frames* individuais de vídeo por *sockets multicast(*ou *unicast)*; Possibilidade de escolha entre endereços IPv4 e IPv6.

Para os testes desta aplicação foi utilizado um vídeo capturado em laboratório, com as seguintes características: 320x240 *pixels*; 24 bits por *pixel*; 999 *frames*; 15 *frames* por segundo (fps); duração de 66,6 segundos; AVI, sem compressão.

A API JMF foi escolhida para a manipulação de *frames* de vídeo em JAVA, sendo utilizado um módulo que o utiliza para transformação de *frames* de qualquer tipo de vídeo para sua representação RGB (*Red, Green, Blue*), utilizando-se uma *array* de tipo "inteiro".

Com o intuito de tornar mais eficiente o processo de transmissão de *frames* de vídeo na rede, um arquivo pré-formatado foi utilizado, com com extensão ".ll". Após o servidor e o cliente serem inicializados, uma troca de mensagens é feita no momento em que o cliente se conecta ao servidor. Esta comunicação é realizada inicialmente por conexões TCP, e a transmissão de vídeo por UDP *multicasting* ou *unicasting*.

3.2. Testes realizados

Os testes tiveram o intuito de verificar- as diferenças de desempenho nas implementações dos protocolos IPv4 e IPv6 no Linux, usandoe as aplicações de vídeo *unicasting* e *multicasting*. Para os testes com a aplicação sobre o cenário configurado, foi utilizado o aplicativo IPTRAF (*IP Traffic*), cuja função é capturar pacotes da rede usando a biblioteca *libpcap* de captura de pacotes do sistema e realizar uma estatística no número de *frames* Ethernet que estão ocupando a rede, além de medir o *throughput* da rede e do número de pacotes IP e não-IP. A transmissão do vídeo foi feita 10 vezes, ou seja, foram transmitidos na rede 9990 *frames* em um tempo que varia de acordo com o protocolo utilizado, com a técnica (*unicasting* ou *multicasting*) e até mesmo com o número de clientes conectados. Após adquiridos os valores, uma média foi feita e um gráfico foi gerado. As seções seguintes ilustrarão as medidas tomadas.

3.2.1. *Throughput* da Rede e Tempo de Transmissão do Vídeo usando a Aplicação Multicast

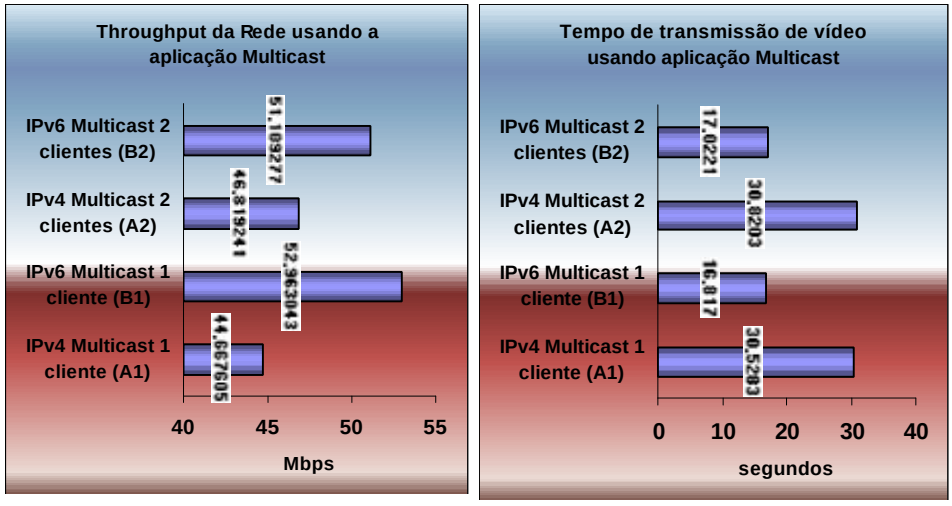

Figura 2: *Throughput* da rede e Tempo de Transmissão do Vídeo usando a Aplicação Multicast

Percebe-se com estes gráficos que a capacidade de transmissão do servidor utilizando a aplicação *multicast* é de aproximadamente 47 Mbps para o protocolo IPv4 e de 54 Mbps para o protocolo IPv6. O fato da aplicação utilizar mais largura de banda com o protocolo IPv6 indica que sua implementação e utilização pelo sistema Linux é mais eficiente em relação ao IPv4.

O tempo de transmissão com o protocolo IPv6 é, em média, 17 segundos, bem menor que o tempo do vídeo original (66,6s), podendo-se atrasar o vídeo no servidor 4 ou armazenar o vídeo no cliente. O tempo de transmissão com o protocolo IPv4 é, em média, de 30 segundos.

O tempo de transmissão do vídeo para cada cliente não mudou com 2 clientes conectados.

3.2.1.2. *Throughput* da Rede e Tempo de Transmissão do Vídeo usando a Aplicação Unicast

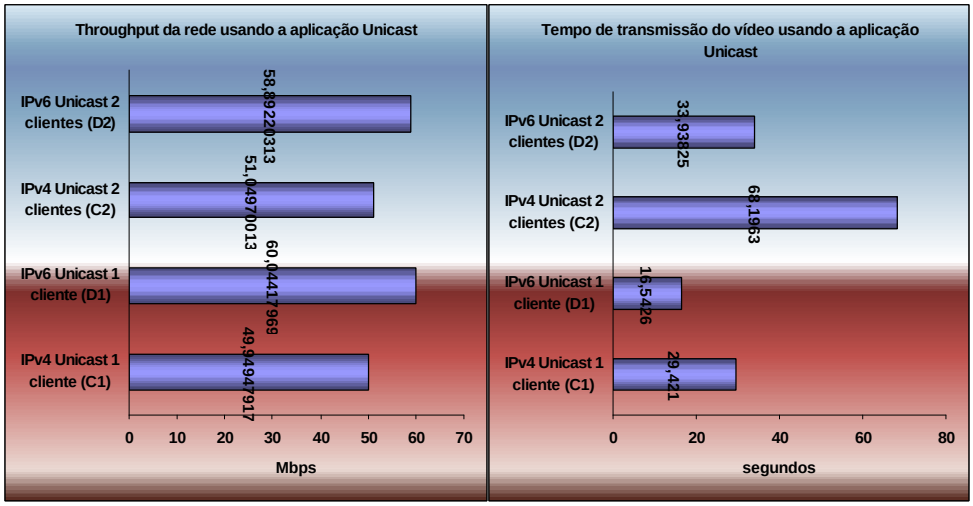

Figura 3: *Throughput* da rede e Tempo de Transmissão do Vídeo usando a Aplicação Unicast

O *throughput* da rede ficou em torno de 50 Mbps com o protocolo IPv4 e 59 Mbps com o protocolo IPv6, percebendo-se uma melhor eficiência no uso do protocolo IPv6.

Porém, o uso de *unicasting* para aplicações de vídeo com mais de 1 cliente conectado não é interessante. Isto pode ser percebido com os valores do gráfico de tempo de transmissão do vídeo, em que com apenas 1 cliente, o tempo médio foi de 29, 42 segundos e com 2 clientes o tempo foi de 68, 19 segundos, usando o protocolo IPv4. O tempo de transmissão para cada cliente praticamente dobrou e, consequentemente a taxa de quadros por segundo caiu para a metade por cliente. Caso um terceiro cliente se conectasse, o tempo supostamente triplicaria, em média, e a taxa de quadros por segundo seria 1/3 da original. Mais uma vez percebe-se que o tempo usando-se o protocolo IPv6 é menor, sendo de aproximadamente 16,5s (59 fps, aproximadamente) para um cliente e 33s (30 fps, aproximadamente) para 2 clientes.

4. Conclusões

Neste trabalho, testes mostraram benefícios do *multicasting* em relação ao *unicasting*, além de uma implementação mais eficiente do protocolo IPv6 em relação ao IPv4 no Linux.

Sendo assim, o uso de aplicações multimídia com o protocolo IPv6 e a técnica de *multicasting* em LANs e WANs pode ser uma solução interessante, provendo um sistema altamente escalável e eficiente. As ferramentas gratuitas utilizadas no trabalho se mostraram bastante interessantes para implementações em âmbito de pesquisa, podendo ser objetos de diversos projetos e aplicações.