GUILHERME STELLA RAVAGNANI

SIMULAÇÃO DO IP MÓVEL VIA NETWORK SIMULATOR (NS2): UMA PROPOSTA DE REDE WIRELESS

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado II para obtenção do título de Bacharel em Ciência da Computação.

Orientador
Professor Ricardo Martins de Abreu Silva

LAVRAS
MINAS GERAIS - BRASIL
2003

GUILHERME STELLA RAVAGNANI

SIMULAÇÃO DO IP MÓVEL VIA NETWORK SIMULATOR (NS2): UMA PROPOSTA DE REDE WIRELESS

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado II para obtenção do título de Bacharel em Ciência da Computação.

Aprovado em ___ de _____ de 2003.

_____
Professor Ricardo Martins de Abreu Silva
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL
2003

Agradecimentos

Agradeço aos meus pais e irmãos pelo apoio e carinho.

A Natália pela paciência e incentivo.

Em especial a Deus por todas as graças que recebi.

Resumo

Simulação do IP móvel via Network Simulator (NS2):
Uma Proposta de rede wireless

O objetivo deste trabalho consiste em simular o envio de pacotes para um *host* móvel em uma rede *wireless* utilizando o IP Móvel, que é uma alternativa de solução para o problema de transferência de informação entre computadores e outros dispositivos móveis. Este trabalho simula três diferentes cenários de transferência de dados: (i) quando o *host* móvel está em sua rede local (*home link*); (ii) quando o *host* móvel está em uma rede externa (*foreign link*) e por ultimo, (iii) quando o *host* móvel migra pra outras redes externas ao seu domínio. A ferramenta utilizada para implementar esta simulação é o *Network Simulator* (NS), que tem sido utilizado com grande freqüência em pesquisas em redes de computadores sempre com suporte e apoio de várias organizações acadêmicas e comerciais.

Abstract

Simulation of mobile IP through Network Simulator (NS2):
A proposal of wireless network

The goal of this work is to simulate the sending of packets to a mobile host in a wireless network using mobile IP, which is a way to solve transference problems of information between computers and others mobile devices. This work simulates three different sceneries of data transfer: (i) when the mobile host is in its local network (home link), (ii) when the mobile host is in an external network (foreign link) and at last, (iii) when the mobile host goes to other external networks in its dominion. The tool used to make this simulation is the Network Simulator (NS), which has been used very often on researches about computer networks, always with support and help from many academic and commercial organizations.

Sumário

Lista de Figuras

Glossário

Área de cobertura (Coverage area) - a área sobre a qual o sinal de um transmissor (sem fio) pode ser recebido.

ARP (Address Resolutation Protocol) - um protocolo padrão que realiza resolução de endereços entre endereços IP e vários endereços da camada de enlace.

Autenticação (Authentication) - Aprova ou desaprova a identidade reivindicada de alguém ou alguma coisa.

Autorização (Authorization) - o processo de verificação onde comprova-se que a unidade tem permissão de acesso a um recurso.

Backbone - um *link* ou *links* de alta velocidade usados para interconectar os roteadores de provedores de serviço ou organizações.

Binding - uma associação no *home agent* entre o *home address* do nó móvel e seu *care-of address* atual e o tempo de vida remanescente desta associação.

Bit - um dígito binário (0 ou 1).

Byte - um grupo de oito *bits* (também chamado de octeto ou caracter).

Cabeçalho (header) - uma pequena porção de dados anexada aos dados passados abaixo por uma camada superior e enviada através da rede.

Camada física - camada 1 do modelo de referência OSI, responsável por transferir sinais elétricos, eletromagnéticos, ópticos, entre outros, que representam *bits* de informação através de um meio particular de comunicação.

Camada de rede (network layer) - camada 3 do modelo de referência OSI, que é responsável por mover pacotes de uma fonte original para o destino final através de uma topologia arbitrária de roteadores e *links*.

Camada de transporte - Camada 4 do modelo de referência OSI, responsável por mover segmentos contendo dados de camadas mais acima, da origem para o destino. Protocolos da camada de transporte confiáveis como o TCP, garantem a não ocorrência de erros e a entrega em ordem seqüencial de dados das camadas superiores. Outros, como o UDP, provêm pouco mais que a capacidade de multiplexação que o IP provê por si mesmo.

Care-of address - um endereço usado temporariamente por um nó móvel como ponto de saída de um túnel quando o nó móvel está conectado em um *foreign link*.

Collocated care-of address - um endereço temporário designado a uma das *interfaces* do nó móvel.

Congestionamento (Congestion) - uma condição na qual os roteadores de uma rede estão sobrecarregados com pacotes e são incapazes de direcioná-los à frente, devido a velocidade de seus *links* conectados.

Controle de congestionamento (Congestion Control) - um algoritmo do TCP que detecta congestionamento e diminui a quantidade de dados que um remetente pode transmitir durante períodos de congestionamento.

Desencapsulamento - o processo de extrair um primeiro pacote (cabeçalho e *payload*), da região de *payload* de um segundo pacote.

DHCP (Dynamic Host Configuration Protocol) - um protocolo pelo qual um *host* obtêm de um servidor certas informações que ele necessita para comunicar-se, como um endereço IP, comprimento de prefixo e o endereço do servidor DNS.

DNS (Domain Name System) - um sistema na *Internet* que mapeia o nome de *hosts* em endereços IP.

Encapsulamento (Encapsulation) - o processo de colocar um primeiro pacote (cabeçalho mais *payload*) dentro da região de *payload* de um segundo pacote.

Ethernet - um protocolo da camada de enlace designado para compartilhar uma *media* e tipicamente utilizado sobre fio de cobre. A *Ethernet* geralmente opera a 10 Mbps ou 100 Mbps.

Foreign Agent - um roteador com no mínimo uma *interface* no *foreign link* atual do nó móvel. Quando um nó móvel utiliza um *foreign agent care-of address*, o *foreign agent* é responsável pela extração de um pacote do túnel e entrega o mesmo para o nó móvel. Um *foreign agent* poderá também servir como um roteador padrão para pacotes enviados por um nó móvel registrado.

Foreign agent care-of address - um endereço de um **foreign agent** que tem no mínimo uma *interface* no *foreign link* atual do nó móvel.

Foreign link (foreign network) - qualquer outro *link* diferente do *home link* do nó móvel, isto é, qualquer *link* no qual o prefixo de rede não é igual ao prefixo de rede do *home address* do nó móvel.

FTP (File Transfer Protocol) - um protocolo da camada de aplicação na *Internet*, usado para transferir arquivos entre dois *hosts* através da rede.

Home address - um endereço permanente de um nó móvel que é utilizado em correspondência com outros nós, independente da sua localização atual.

Home Agent - um roteador com no mínimo uma *interface* no *home link* do nó móvel. O *home agent* intercepta pacotes destinados para o *home address* e envia-os via túnel para *o care-of address* do nó móvel, quando o mesmo está conectado ao *foreign link*. Um nó móvel informa a seu *home agent* seu *care-of address* atual através de um protocolo de registro autenticado, definido pelo IP Móvel.

Home link (home network) - o *link* que possui o mesmo prefixo de rede de um *home address* permanente de um nó móvel.

Hop - a distância entre uma fonte e um destino medida no número de roteadores que os separam (por exemplo, *hosts* vizinhos estão "zero *hops*" distantes, *hosts* separados por um simples roteador estão "um *hop*"distantes, etc.)

Host - qualquer nó que não seja um roteador.

Host móvel (mobile host) - um nó móvel que mantém conectividade por si mesmo.

IETF (Internet Engineering Task Force) - grupos de trabalho para produzir e definir padrões para a *Internet* ou resolver um problema operacional na *Internet*. O grupo de trabalho *IP Routing for Wireless/Mobile Hosts (mobileip)* produziu as RFC's 2002-2006, que definem a operação do IP Móvel.

Interface - a conexão física de um nó com o *link*.

Interface virtual - uma *interface* que realiza encapsulamento ou desencapsulamento, mas, ao contrário de uma *interface* física, não conecta um nó a um meio físico.

Internet - uma rede global baseada no grupo de protocolos do TCP/IP.

IP (Internet Protocol) - o protocolo da camada de rede da *Internet*. O IP provê a entrega de pacotes pelo melhor esforço, sem conexão, em nome da camada de transporte e dos protocolos das camadas mais altas.

LAN (Local Area Network) – uma rede cobrindo uma área geográfica relativamente pequena, tipicamente na ordem de metros ou centenas de metros.

Link - a *media* sobre a qual os nós podem comunicar-se na camada de *link*, isto é, a camada imediatamente abaixo do IP. Exemplos: *Ethernet*, *links* PPP, X.25, *Frame Relay* ou redes ATM.

Media (Medium) - uma facilidade de comunicação sobre a qual os *bits* podem ser transferidos na camada física, isto é, a camada abaixo da camada de *link* de dados. Exemplos: fio de cobre, fibra-ótica, e canais de rádio.

Mobilidade (mobility) - a habilidade de um nó mudar seu ponto de conexão de uma rede para outra, enquanto mantém todas as comunicações existentes e usando o mesmo endereço IP no novo *link.*

Mobilidade heterogênea (Heterogeneous mobility) - mobilidade através de diferentes tipos redes ou meios diferentes. O IP Móvel é único na habilidade de prover mobilidade heterogênea.

Mobilidade homogênea (Homogeneous mobility) - mobilidade através de um simples tipo de *link* e a *media* subjacente. CPDP e IEEE 802.11 provêm mobilidade homogênea, enquanto o IP Móvel acomoda tanto mobilidade homogênea como heterogênea.

Nó - um *host* ou um roteador.

Nó móvel (mobile node) - um nó com capacidade de mobilidade, isto é, um nó que pode mudar seu ponto de conexão de um *link* para outro, enquanto mantém todas as comunicações existentes e usando o mesmo endereço IP no novo *link*.

OSI (Open System Interconnection) - um modelo de referência de sete camadas para comunicações de computadores.

Pacote (packet) - cabeçalho mais *payload*.

Payload - a porção de um segmento, pacote ou *frame* contendo dados que serão passados abaixo por um protocolo de uma camada superior ou aplicação.

PPP (Point-to-Point Protocol) - um padrão da camada de enlace para encapsulamento de pacotes de vários diferentes protocolos em redes ponto a ponto, como linhas telefônicas.

Prefixo de rede (network prefix) - uma *string* de *bits* que consiste de algum número de *bits* iniciais de um endereço IP. O prefixo de rede é idêntico para todos os nós no mesmo ponto de conexão.

Protocolo - um grupo de regras e procedimentos governando como dois ou mais computadores cooperam para realizar funções específicas através da rede.

Protocolo de roteamento (Routing protocol) - um protocolo usado entre roteadores para trocar informações sobre a localização de destinos na rede (por exemplo: OSPF, RIP, BGP).

Proxy ARP - uma mensagem *ARP Reply* enviada por um nó designado em nome de algum outro nó, que é incapaz de responder a uma mensagem *ARP Request* por si mesmo. Por exemplo, um *home agent* envia mensagens *Proxy ARP* para um nó móvel que registrou seu *care-of address* em um *foreign link*.

Rede móvel (mobile network) - uma coleção de *hosts* e roteadores que são fixos entre si, mas são coletivamente móveis, como uma unidade, em relação ao resto da porção fixa da *Internet*.

Registro (Registration) - o processo pelo qual um nó móvel informa a seu *home agent* seu *care-of address* atual e, em alguns casos, requer serviço de um *foreign agent*.

RFCs (request for Comments) - uma série de documentos que contêm protocolos padrões da *Internet*, como também uma grande variedade de outras informações interessantes e úteis, de relevância para a comunidade da *Internet*.

Roteador (Router) - um dispositivo da camada de rede que direciona pacotes não explicitamente endereçados para ele mesmo.

Tabela de roteamento (routing table) - uma entidade de *software* ou *hardware* dentro de um *host* ou roteador que utiliza isto para selecionar o próximo destino e a *interface* de saída para pacotes que ela transmite.

TCP (Transmission Control Protocol) - um protocolo de transporte confiável, *full-duplex* e orientado a conexão usado através da *Internet*.

Topologia - o *layout* de uma rede, especificamente, a interconexão entre *hosts*, roteadores e pontos de conexão na rede.

Túnel (tunnel) - o caminho seguido por um primeiro pacote enquanto ele é encapsulado dentro do *payload* de um segundo pacote.

UDP (User Datagram Protocol) - um protocolo da camada de transporte não confiável, não orientado a conexão usado através da *Internet*. O UDP é um pouco mais que uma *interface* da camada de aplicação para o IP.

Capítulo 1

Introdução

A necessidade crescente do homem em romper suas próprias limitações buscando soluções tecnológicas ou mesmo novas tecnologias como forma de expandir o universo de possibilidades, faz da informação um bem de valor inestimável. Como é possível estabelecer comunicação a partir de um automóvel, uma aeronave, um táxi, um ônibus ou em situações de calamidade como enchentes e terremotos e até mesmo em uma guerra?

Alguns sistemas de comunicação utilizam os fios de cobre como o par trançado, o cabo coaxial ou utilizam fibra óptica como forma de interconectar equipamentos. Outros sistemas transmitem os dados pelo espaço, como o caso da transmissão por raios infravermelhos, laser, microondas e rádio. A essas redes que utilizam estas técnicas chamamos de redes sem fio [Derfler (1993)].

A evolução da comunicação sem fio e da tecnologia da informática buscam atender diversas necessidades do mercado como, por exemplo, as redes locais sem fio, os serviços celulares, as transmissões via satélite, TV, sistemas de navegação, etc [Ferreira (1999)].

Muitos pesquisadores e visionários sustentam firmemente a posição de que o futuro das redes de computadores será composto de equipamentos portáteis interconectados por *backbones* de fibra óptica de alta velocidade [Tanenbaum (1997)]. Eles são motivados, basicamente, por dois processos distintos que convergem este enfoque a crescente miniaturização dos

componentes eletrônicos e o advento da tecnologia de comunicação pessoal sem fio.

Atualmente organizações de todo porte necessitam acessar e compartilhar informações através de redes digitais. O poder das redes de comunicação foi percebido como elemento indispensável no cenário mundial. Entretanto, muitos dos recursos hoje desfrutados, até poucos anos atrás eram limitados a acessos através de redes locais com fio [Lough (1997)]. Profissionais com necessidades de mobilidade constante e acesso à informação *on-line*, como os médicos, profissionais liberais, usuários do comércio e universitários contribuíram para disseminar a tecnologia das redes locais sem fio (WLAN – *Wireless Local Area Network*), notadamente após a autorização do uso da banda Industrial, Científica e Médica (ISM – *Industrial, Scientific e Medical*) regulamentada pelo órgão de comunicações nos Estados Unidos (FCC – *Federal Communications Comission*) em 1985 e a regulamentação do padrão IEEE 802.11 de redes sem fio em junho de 1997 e as adições em 1999.

O problema é que a maioria dos protocolos de rede e as complexas regras que definem como a informação é trocada entre dois ou mais computadores, foram desenvolvidos no passado, quando os computadores não tinham grandes necessidades de mobilidade. Como resultado disto, muitos destes protocolos falham em operar na presença de computadores móveis, tendo como exemplo um dos principais protocolos utilizados para conectar a Internet de hoje, o IP (*Internet Protocol*).

No IP os pacotes são roteados para seus destinos de acordo com o endereço IP, que são endereços associados a uma localização de rede fixa. Quando o destino de um pacote é um *host* móvel esta associação não ocorrerá, isso significa que cada novo ponto de ligação feito pelo *host* é associado com um novo número de rede, portanto, um novo endereço IP, tornando a mobilidade impraticável.

Devido a estes problemas de mobilidade foi desenvolvido o IP Móvel que é uma solução para o problema de transferência de informação de ou para computadores móveis. O IP Móvel é independente do meio físico através do qual um computador móvel se comunica e ainda permite ao mesmo alterar sua localização sem recomeçar suas aplicações e sem interromper qualquer comunicação em andamento. O IP Móvel também é amplamente empregado nos sistemas de comunicações móveis (sistemas celulares) que começam a ser implantados nos dias de hoje.

O IP Móvel permite que o dispositivo móvel tenha dois endereços, o *home address* (endereço local) que é estático e é usado, por exemplo, para identificar conexões TCP e o *care-of address,* que muda a cada novo ponto de conexão e pode ser visto como o endereço topológico do *host* móvel, ele indica o número de rede e assim indica o ponto de conexão do *host* móvel com respeito à topologia de rede.

Este trabalho objetiva simular o envio de pacotes para um *Host* Móvel (HM) utilizando IP móvel em três situações diferentes:

1. O HM em sua própria rede (*home link*).
2. O HM após ter migrado para outra rede IP (*foreign link*).
3. O HM recebendo pacotes em movimento, ou seja, migrando de uma rede para outra (*handoff*).

O método por simulação permite através de um custo baixo à confecção de modelos complexos e resolução destes com menor desenvolvimento matemático, não obrigando a simplificações do modelo, o que pode resultar em imperfeições na representação do sistema. O método por simulação também permite obter os resultados num curto prazo, para isso, emprega-se todo o poder computacional para a representação do modelo requerido.

Um dos simuladores que tem sido utilizado com grande freqüência em pesquisas em redes de computadores é o *Network Simulator* – NS – [NS-2]. Concebido em 1989 a partir de uma variação do REAL *Network Simulator* [KESHAV], um projeto da *Cornell University*, EUA, o NS tem evoluído desde então, sempre com suporte e apoio de várias organizações acadêmicas e comerciais durante períodos.

No capítulo seguinte é detalhado o IP Móvel bem como a arquitetura e funcionamento do NS em redes sem fio e o suporte para o IP Móvel, no capítulo três é descrito as diferentes situações de simulação objetivadas neste trabalho e o modelo de sua implementação e por fim temos o cronograma de atividades previstos para o desenvolvimento do projeto no capítulo quatro.

Capítulo 2

Referencial Teórico

2.1 IP Móvel

O IP Móvel foi produzido pelo grupo de trabalho IP *Routing* for *Wireless/ Mobile Hosts* (mobileip) do IETF (*Internet Engineering Task Force*) [Perkins], formado em Junho de 1992. O documento padrão do IP Móvel incluía as seguintes RFCs (*Request for Comments*):

- RFC 2002, que define o protocolo IP Móvel;
- RFCs 2003, 2004 e 1701 que definem três respectivos tipos de *tunneling* usados no IP Móvel;
- RFC 2005, que descreve a aplicabilidade do IP Móvel;
- RFC 2006, que define a MIB (Mobile IP Management Information Base). A MIB define uma coleção de variáveis dentro de um *host* que implementa o IP Móvel, que pode ser examinado ou configurado por uma estação de gerenciamento usando a versão 2 do SNMPv2 (*Simple Network Management Protocol*) [RFC 1905].

O termo IP Móvel geralmente refere-se a todos estes documentos que, juntos, formam a tecnologia padrão para mobilidade de dispositivos na Internet.

O IP Móvel é uma solução da camada de rede que permite a mobilidade na Internet. Isto significa que o IP Móvel realiza sua tarefa configurando as tabelas de roteamento em *hosts* apropriados, de modo que pacotes IP podem ser enviados a *hosts* móveis não conectados a seus pontos de acesso padrão

resolvendo o problema primário de roteamento de pacotes IP para *hosts* móveis, o que é um grande passo para prover mobilidade na Internet. De fato, o IP Móvel pode ser considerado como um protocolo de roteamento com o propósito de permitir que pacotes sejam roteados para *hosts* móveis, os quais podem potencialmente mudar suas localizações rapidamente.

Uma solução completa de mobilidade envolveria um aprimoramento nas outras camadas da pilha de protocolos. Portanto, o escopo da solução do IP Móvel é simplesmente a especificação dos mecanismos que são necessários para rotear pacotes para *hosts* móveis, na camada de rede. Outras tecnologias, incluindo modificações do TCP e das aplicações, estão fora do objetivo do IP Móvel.

Como um protocolo da camada de rede, o IP Móvel é completamente independente do meio físico sobre o qual ele opera. Desta forma, um *host* empregando IP Móvel pode mover-se de um tipo de meio para outro, sem perder a conectividade. Por exemplo, o IP Móvel permite a um *notebook* desconectar-se de uma rede *Ethernet* por fio e chavear para uma interface LAN sem fio, sem experimentar uma interrupção no serviço de rede. A habilidade de um *host* móvel mover-se entre diferentes tipos de meio mantendo sua capacidade de comunicar-se é denominada de mobilidade heterogênea. Da mesma forma, o IP Móvel também permite a um *host* móvel mover-se na rede de um link para outro, no mesmo meio (em uma localização diferente) e manter sua conectividade. A tecnologia que permite o movimento entre diversas conexões da rede, no mesmo tipo de meio físico, é conhecida como mobilidade homogênea.

2.1.1 Definição

O padrão proposto para o IP Móvel pelo *Internet Engineering Task Force* (IETF) define três novas entidades funcionais [Sun (2001)] que estão ilustradas na Figura 2.1:

Host móvel: um *host* que pode alterar seu ponto de conexão na Internet *(link)*, mantendo todas as comunicações em andamento.

Home Agent: um roteador com uma interface no *home link* do *host* móvel, no qual:

a) Sempre que o *host* móvel se muda para outra localização, registra o seu novo *care-of address* no seu *home agent* informado sua localização atual;
b) Recebe todos os pacotes destinados ao *host* móvel (*home address*) e envia via *tunnel* para a posição atual do *host* móvel, isto é, para o *care-of address*.

Foreign Agent: Um roteador com uma interface no *foreign link* do *host* móvel que:

a) Informar ao *home agent do host* móvel seu respectivo *care-of address* atual;
b) Responsável por rotear os pacotes gerados pelo *host* móvel enquanto conectado a um *foreign link*, substituindo o *home agent*

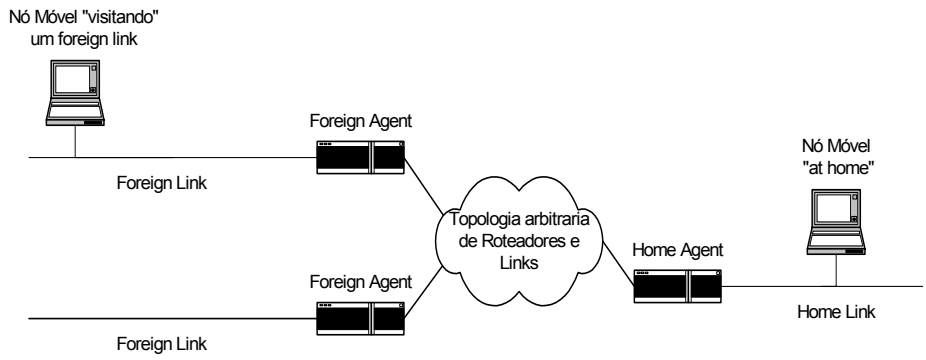

Figura 2.1 - Entidades e relacionamentos do IP Móvel.
Fonte: Dados da Pesquisa

2.1.2 *Home Address, Home Link e Home Agent*

O *home address* é o endereço IP designado ao *host* móvel permanentemente, isto é, da mesma maneira que seria designado para um *host* estacionário ou para um roteador e faz parecer que o *host* móvel está continuamente disponível para receber dados na sua rede local.

O prefixo de rede do *home address* de um *host* móvel define seu *home link* (rede local). Portanto, seu *home link* é a rede na qual o *host* móvel parece alcançável para o resto da Internet. O seu *home agent* é o roteador que tem no mínimo uma interface no *home link* do *host* móvel e é a razão pela qual o *host* móvel é alcançável através de seu *home address* até quando não esta em sua rede local.

O *home link* do *host* móvel não precisa ser uma ligação física, ele pode ser um ponto de conexão virtual existente somente como um software dentro de seu *home agent*. Portanto o home agent pode ser considerado como tendo uma interface virtual através da qual ele se conecta ao seu *home link* virtual [Sun (2001)].

2.1.3 *Care-of Address, Foreign Link e Foreign Agent*

Um endereço *care-of address* é o atual endereço IP do *host* móvel que está visitando um *foreign link*, portanto não localizado em seu *home link*. Este endereço é específico para o *foreign link* atualmente sendo visitado por um *host* móvel, ele é alterado toda vez que o host move-se de um foreign link para outro. O *care-of address* é utilizado como endereço IP de destino do *tunnel*, isto é, quando o *home agent* quer enviar um pacote para o *host* móvel situado em um *foreign link* e deve ser no máximo um passo à frente (*next hop*) a partir do *foreign link* do *host* móvel, esses pacotes destinados para um *care-of address* podem ser entregues usando os mecanismos de roteamento existentes não sendo necessários procedimentos específicos do IP Móvel.

Existem conceitualmente, dois tipos de *care-of address*:

- *Foreign agent care-of address*: É um endereço IP de um *foreign agent* o qual tem uma interface no *foreign link* sendo visitado pelo *host* móvel. Um foreign *agent care-of address* pode ser qualquer um dos endereços IP do *foreign agent*. Deste modo, o prefixo de rede de um *foreign agent care-of address* não precisa ser igual ao prefixo de rede que foi designado para o *foreign link*. Um *foreign agent care-of address* pode ser partilhado por vários *hosts* móveis simultaneamente[Sun (2001)];
- *Collocated care-of address*: É um endereço IP temporariamente designado para uma interface do próprio *host* móvel. O prefixo de rede de um *collocated care-of address* deve ser igual ao prefixo de rede que foi designado para o *foreign link* sendo visitado pelo *host* móvel. Este tipo de *care-of address* poderá ser usado pelo *host* móvel em situações onde não há *foreign agents* disponíveis no *foreign link*. Um *collocated*

care-of address pode ser usado por somente um *host* móvel ao mesmo tempo[Sun (2001)].

2.1.4 *Tunneling*

Um túnel é um caminho criado entre um *home agent* e um *foreign agent*, de forma a permitir que dados destinados a um *host* móvel pertencente àquele *home agent* possam ser entregues ao *host* móvel, no momento em que o mesmo estiver conectado ao *foreign link* associado ao *foreign agent* em questão.

O mecanismo de encapsulamento padrão (Figura 2.2) que tem de ser suportado por todos os agentes móveis que usam IP Móvel é o *IP-within-IP*[Perkins (1996-I)]. Ao usar *IP-within-IP* o *home agent* insere no início do túnel um novo cabeçalho IP à frente do cabeçalho IP de qualquer datagrama endereçado ao *home address* do *host* móvel. O novo cabeçalho do túnel usa o *care-of address* do *host* móvel como endereço IP destino, ou destino do túnel. O endereço IP do início do túnel é o *home agent* e o cabeçalho do túnel usa o número do nível mais alto do protocolo (número 4), indicando que o próximo cabeçalho é novamente um cabeçalho IP. No *IP-within-IP* o cabeçalho original IP é preservado como a primeira parte dos dados a ler do cabeçalho do túnel. Portanto, para recuperar o pacote original o *foreign agent* apenas tem de eliminar o cabeçalho do túnel e entregar o resto ao *host* móvel.

Por vezes o cabeçalho do túnel usa o número 55 no cabeçalho interior. Isto acontece quando o *home agent* usa encapsulamento mínimo em vez do *IP-within-IP*. O processamento no cabeçalho do encapsulamento mínimo[Perkins (1996-II)] é ligeiramente mais complicado do que o processamento no *IP-within-IP*, porque alguma da informação do cabeçalho do túnel está combinada com informação no cabeçalho de encapsulamento mínimo para reconstituir o cabeçalho IP original. Por outro lado, o overhead no cabeçalho é reduzido.

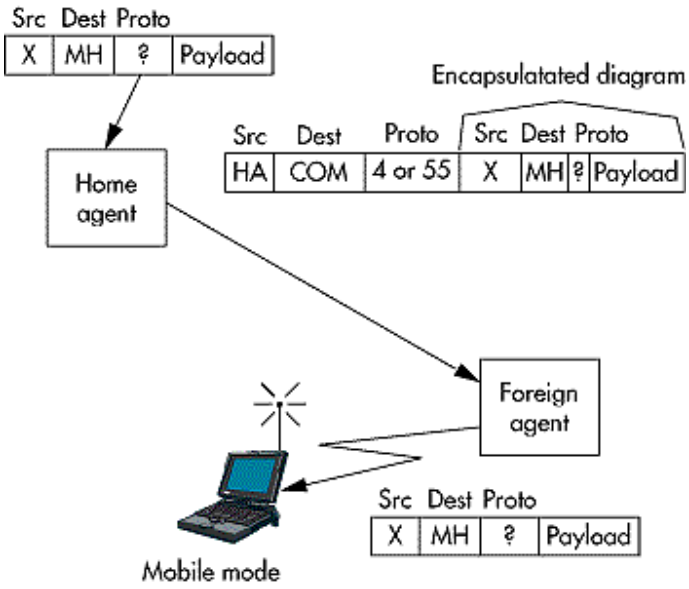

Figura 2.2 - IP *Tunneling*
Fonte: [Perkins]

2.1.5 *Agent Discovery*

O *Agent Discovery*[Sun (2001)] é o processo pelo qual o *host* móvel:

- determina se ele está atualmente conectado em seu *home link* ou em um *foreign link*;
- detecta se ele moveu-se de um link para outro;
- obtém um endereço *care-of address*, quando conectado em um *foreign link*.

O *Agent Discovery* consiste de duas simples mensagens. A primeira, *Agent Advertisement*[Sun (2001)] é usada pelos agentes (*home, foreign* ou ambos) para anunciar suas capacidades para os *hosts* móveis. Especificamente

Agent Advertisements são periodicamente transmitidas como *multicasts* ou *broadcasts* para cada link nos quais um *host* é configurado para operar como *home agent*, *foreign agent* ou ambos. Isto permite ao *host* móvel que está conectado ao *link* determinar se qualquer agente está presente e suas respectivas identidades (endereços IP) e capacidades.

O segundo tipo de mensagem, *Agent Solicitation*[Sun (2001)], é enviada pelos *hosts* móveis que não têm a "paciência" de esperar pela próxima transmissão periódica de mensagens *Agent Advertisements*. O objetivo da mensagem *Agent Solicitation* é forçar qualquer agente no *link* a imediatamente transmitir uma mensagem *Agent Advertisement*. Isto é útil em situações onde a freqüência na qual os agentes estão transmitindo é muito baixa, para um *host* móvel que está se movendo rapidamente de um *link* para outro.

2.1.6 Registro

Um *host* móvel registra-se quando ele detectar que seu ponto de conexão na rede tenha mudado de um *link* para outro. Devido ao fato de que estes registros são válidos somente durante um tempo de vida específico, um *host* móvel precisa registrar-se novamente quando o registro existente está próximo a expirar.

O Registro do IP Móvel é um processo através do qual o *host* móvel:

- Requer serviços de roteamento de um *foreign agent* em um *foreign link*;
- Informa a seu *home agent* seu *care-of address* corrente;
- Renova um registro que está próximo a expirar;
- Cancela o registro quando ele retorna a seu *home link*;
- Tenha múltiplos e simultâneos *care-of address* registrados com seu *home agent*, permitindo que o *home agent* envie via túnel uma cópia dos

pacotes destinados para o *home address* do *host* móvel para cada um dos múltiplos *care-of address*;

- Cancele o registro de um *care-of address* específico enquanto mantêm os outros;
- Dinamicamente verifica o endereço de um *home agent* potencial, se o *host* móvel não tem um conhecimento prévio do seu (ou seus) *home agent(s)*.

Quando um *host* móvel detecta que ele está conectado a seu *home link*, ele cancela o registro com seu *home agent* e começa a comportar-se como qualquer dispositivo fixo ou roteador, isto é, não utiliza nenhuma funcionalidade adicional do IP Móvel. Quando o *host* móvel detecta que ele está conectado a um *foreign link*, ele obtém um *care-of address* e registra o mesmo com seu *home agent*, e esse registro consiste de uma troca de mensagens *Registration Request* e *Registration Reply* entre um *host* móvel e seu *home agent* possivelmente via um *foreign agent* (se existir um no *foreign link*) [Sun (2001)],este processo de registro está ilustrado na Figura 2.3.

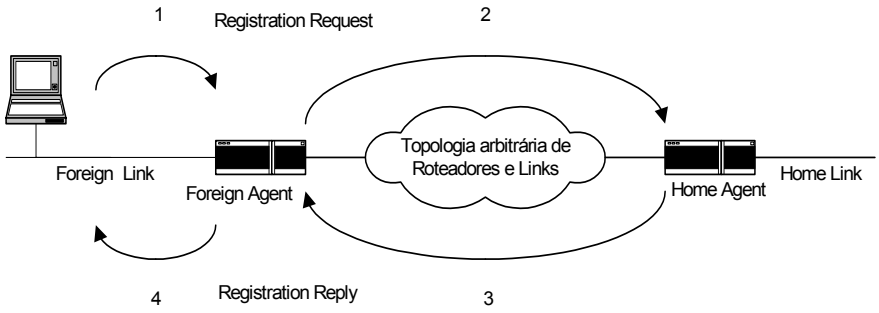

Figura 2.3 - Processo de registro no IP Móvel
Fonte: Dados da Pesquisa

2.1.7 Autenticação

A necessidade de autenticar a informação do registro desempenhou um papel fundamental na especificação do IP Móvel. Cada *host* móvel e o *home agent* têm de partilhar uma associação de segurança e serem capazes de usar *Message Digest* 5 (RFC 1321) [Rivest (1992)] com chaves de 128 bits para criar assinaturas digitais impossíveis de falsificar nos pedidos de registro. A assinatura é calculada aplicando o algoritmo *hash* do MD5 a todos os dados do cabeçalho da mensagem e às extensões que precedem a assinatura.

Para assegurar o pedido de registro, cada pedido deve conter informação unívoca para que na prática dois diferentes pedidos de registro não tenham o mesmo *hash* MD5. Se assim não fosse, o protocolo estaria sujeito a ataques por resposta, nos quais *hosts* maliciosos poderiam guardar registros válidos para respostas posteriores, destruindo a possibilidade do *home agent* estabelecer posteriormente um túnel com o *care-of address* do host móvel. Para assegurar que isto não acontece o IP Móvel inclui na mensagem de registro um campo especial de identificação que muda em cada novo registro. A semântica exata do campo de identificação depende de vários parâmetros, que estão descritos com grande detalhe na especificação do protocolo. A grosso modo, existem duas formas principais de tornar o campo de identificação único [Perkins]:

- Usar um selo temporal

  Assim cada novo registro terá um selo temporal posterior diferindo assim de registros anteriores.

- Usar um número pseudo-aleatório

Usando bits suficientes é muito improvável que dois valores independentes escolhidos para o campo de identificação sejam iguais. Quando

este método é usado, o IP Móvel define um método que protege tanto o pedido de registro como a resposta de se repetirem e usa 32 bits aleatórios no campo de identificação.

Se o *host* móvel e o *home agent* se afastarem demasiadamente na sincronização no uso de selos temporais ou se perderem o rastro aos números aleatórios esperados, o *home agent* rejeita os o pedido de registro e inclui informação que permita uma nova sincronização dentro da resposta [Perkins].

O campo de identificação também é usado pelo *foreign agent* para fazer corresponder pedidos de registro pendentes às respostas aos registros quando estes chegam ao *home agent* e para subseqüentemente ser capaz de fazer chegar à resposta ao *host* móvel. O *foreign agent* também guarda outra informação para os pedidos pendentes, incluindo o *home address* do *host* móvel, o endereço MAC do *host* móvel, o número do porto fonte para o pedido de registro do *host* móvel, o tempo de vida do registro proposto pelo *host* móvel e o endereço do *home agent*.

O *foreign agent* pode limitar os tempos de vida dos registros a um valor configurável que ele põe nos seus anúncios de agente. O *home agent* pode reduzir o tempo de vida do registro, que ele inclui como uma parte da resposta ao pedido, mas nunca o pode aumentar.

No IP Móvel os *foreign agents* são, sobretudo passivos, fazendo essencialmente o que lhe dizem para fazer, como entregar pedidos de registro e respostas para frente e para trás entre o *home agent* e o *host* móvel, etc. O *foreign agent* também desencapsula o tráfego vindo do *home agent* e envia-o ao *host* móvel. Note-se que os *foreign agents* não têm de se autenticar perante o *host* móvel ou o *home agent*. Um *foreign agent* fictício pode-se fazer passar por um *foreign agent* verdadeiro simplesmente seguindo o protocolo e oferecendo anúncios de agente ao *host* móvel. O agente fictício pode, por exemplo, então se recusar a fazer seguir os pacotes desencapsulados para o *host* móvel quando os

recebe. No entanto, o resultado não é pior do que se algum host fosse levado a usar o roteador errado, o que é possível usando anúncios de *routing* não autenticados como está especificado no RFC 1256[Deering (1991)].

2.1.8 Descoberta Automática do *Home Agent*

Quando o *host* móvel não consegue contatar o seu *home agent* o IP Móvel tem um mecanismo que deixa o *host* móvel tentar registrar-se com outro *home agent* desconhecido na sua rede local. Este método de descoberta automática do *home agent* é conseguido usando um endereço IP de *broadcast* em vez do endereço IP do *home agent* como destino do pedido de registro. Quando o pacote de *broadcast* chega à rede local, outros *home agents* da rede irão enviar uma mensagem de rejeição ao *host* móvel. No entanto, os seus avisos de rejeição contêm os seus endereços para o *host* móvel os usar numa nova tentativa de registro. Note-se que este *broadcast* não é um *broadcast* na Internet, mas um *broadcast* direcionado que chega apenas a *hosts* IP da rede local.

2.1.9 Resumindo o Funcionamento do IP Móvel

O funcionamento do IP Móvel se dá da seguinte forma:

1) *Home agents* e *foreign agents* notificam suas presenças em todos os *links* conectados através de *multicast* ou *broadcast* periódicos de mensagens especiais do IP Móvel chamadas *Agent Advertisements*;
2) *Hosts* móveis "escutando" estas mensagens (*Agent Advertisements*) e examinando seus conteúdos determinam se eles estão conectados em seus *home links* ou em um *foreign link*. Enquanto conectados em seus respectivos *home links*, *hosts* móveis agem como *host* estacionários,

isto é, eles não utilizam nenhuma funcionalidade do IP Móvel. Os passos a seguir consideram que o *host* móvel está conectado em um *foreign link*;

3) Um *host* móvel conectado em um *foreign link* adquire um *care-of address*. Um *foreign agent care-of address* pode ser lido de um dos campos dentro das mensagens *Agent Advertisement*. Um *collocated care-of address* deve ser adquirido por algum procedimento de designação como, por exemplo, o DHCP, o PPP ou por uma configuração manual;
4) O *host* móvel registra o *care-of address* adquirido no passo anterior com seu *home agent*, utilizando uma troca de mensagens definida pelo IP Móvel. No procedimento de registro, o *host* móvel solicita o serviço de um *foreign agent*, se um estiver presente *host* link. Para fins de segurança, as mensagens de registro devem ser autenticadas;
5) O *home agent* ou qualquer outro roteador no *home link* avisa que o prefixo de rede do *home address* do *host* móvel pode ser atingido (*advertise reachability*), deste modo os pacotes destinados para o *home address* do *host* móvel são atraídos. O *home agent* intercepta estes pacotes, possivelmente usando um *Proxy* ARP e os envia via *tunnel* para o *care-of address* que o *host* móvel registrou no passo anterior;
6) No *care-of address* (no *foreign agent* ou em uma das próprias *interfaces* do *host* móvel) o pacote original é extraído do *tunnel* e então entregue ao *host* móvel;
7) Na direção reversa, pacotes enviados pelo *host* móvel são diretamente roteados para seus destinos, sem qualquer necessidade de *tunneling*. O *foreign agent* trabalha como um roteador para todos os pacotes gerados pelo *host* móvel visitante.

2.2 Network Simulator (NS)

O NS é um simulador que tem sido utilizado com grande freqüência em pesquisas em redes de computadores. Atualmente o seu desenvolvimento é suportado pelo DARPA *(Defense Advanced Research Projects Agency*, EUA) através do projeto SAMAN [SAMAN] e pela NSF (*National Science Foundation*, EUA) através do projeto CONSER [CONSER], em colaboração com outros pesquisadores como o centro ICIR [ICIR]. O simulador já recebeu apoio do *Lawrence Berkeley National Laboratory*, do Xerox PARC (*Palo Alto Research Center*), da Universidade da Califórnia em Berkeley, *Sun Microsystems* e também agrega diversos módulos contribuídos por pesquisadores independentes. É um software de código livre e fornecido gratuitamente [NS-2]. Uma lista de discussão é mantida pelos desenvolvedores, onde os pesquisadores de diversas partes do mundo podem trocar idéias e experiências, e também propor correções para o código do simulador, que depois de avaliadas podem ser incorporadas. Estes pesquisadores, oriundos de países como Estados Unidos, Índia, Inglaterra, Itália, Taiwan e também Brasil, contribuem para o valor desta ferramenta.

O NS é um simulador de eventos discretos, focado para o desenvolvimento de pesquisas em redes de computadores. Ele prevê suporte a TCP e variantes do protocolo, *multicast*, redes sem fio (wireless), roteamento e satélite.Tem facilidades de *tracing*, que é a coleta e registro de dados de cada evento da simulação para análise posterior. Possui um visualizador gráfico para animações da simulação (*nam – network animator*), *timers* e escalonadores, modelos para controle de erros e algumas ferramentas matemáticas como gerador de números aleatórios e integrais para cálculos estatísticos. Inclui também uma ferramenta de plotagem, o *xgraph*, e vários tipos de geradores de tráfego.

O NS foi desenvolvido na linguagem orientada a objetos C++, de forma modular. O uso desta linguagem nos módulos confere velocidade e mais praticidade na implementação de protocolos e modificação de classes. A interface com o usuário, configuração, estabelecimento de parâmetros e manipulação de objetos e classes é feita em modo texto, através da linguagem interpretada Otcl [OTCL], que também é orientada a objetos.

2.2.1 Estrutura

A interface entre o usuário e o NS dá-se através da linguagem *script* Otcl. Segundo os desenvolvedores, a divisão em duas linguagens (Otcl e C++) objetiva dar ao simulador tanto velocidade e poder, quanto flexibilidade e facilidade de mudança de parâmetros. O núcleo do simulador é escrito em C++, conferindo velocidade, mas esta linguagem torna-se lenta para manipulação constante ou mudança de parâmetros. Otcl, por ser interpretada, é bem mais lenta, porém pode ser facilmente alterada. Além do mais, os objetos compilados são disponibilizados para o interpretador Otcl por *linkagem*, o que virtualmente cria um objeto Otcl para cada objeto C++, e que podem ser manipulados através das facilidades da Otcl. A Figura 2.4 mostra a construção geral do *ns* . Um usuário comum atua no perímetro "tcl", escrevendo *scripts* em Otcl e executando simulações. Os escalonadores de eventos e os componentes de rede são implementados em C++ e disponibilizados ao interpretador Otcl através de uma replicação feita pela camada tclcl, que recria os objetos C++ em objetos Otcl, e que podem finalmente ser manipulados por esta última (processo denominado *linkage*). Todo o conjunto constitui-se no NS, que é um interpretador de Otcl com bibliotecas de simulação para redes de computadores.

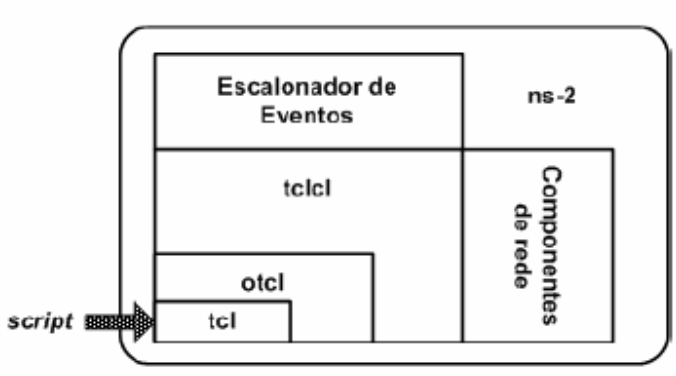

Figura 2.4 - Arquitetura do NS

Fonte: [Marcos]

Montar uma simulação no NS é preciso então, primeiramente, escrever um *script* em Otcl. Este *script* contém as seguintes partes básicas:

- criação do objeto Simulador;
- abertura de arquivos para *tracing* e análise posterior;
- criação da topologia de rede;
- criação de nós ou nodos;
- conexão dos nós entre si (*links*);
- criação das filas de saída;
- criação dos agentes de 4ª. Camada e conexão com *hosts*;
- criação dos geradores de tráfego (nível de Aplicação) e conexão com agentes de 4ª. Camada (nível de Transporte);
- programação dos escalonadores e *timers;*
- fechamento da simulação, animação e geração de estatísticas.

O processo de simulação pode ser assim resumido (Figura 2.5):

- confecção do *script* (arquivo texto comum);
- execução do *script* com o comando *ns nomedoscript.*tcl;

- arquivos de *tracing* serão gerados com registro de cada evento simulado.

- Após conclusão da simulação:
- imprimir estatísticas calculadas no *script;*
- visualizar os eventos com o *nam;*
- analisar resultados através dos arquivos de *tracing* com apoio de ferramentas apropriadas (*awk*).

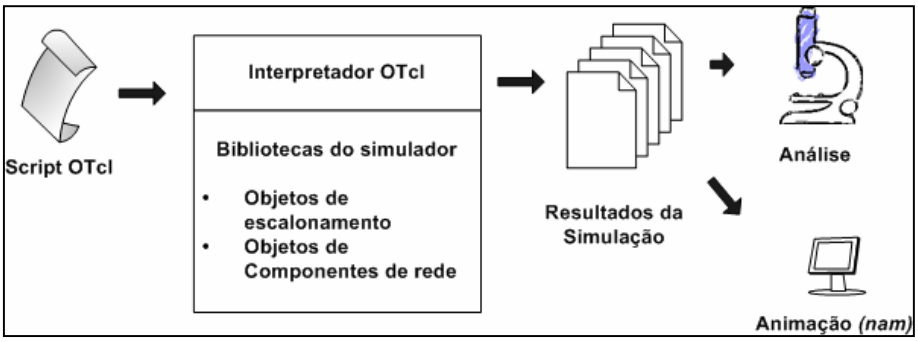

Figura 2.5 - Esquema de utilização do NS

Fonte: [Marcos]

Um ponto importante a observar é que o NS não fornece estatísticas de simulação de modo automático; estas devem ser obtidas através de procedimentos matemáticos no *script* ou pela manipulação de objetos especiais chamados monitores. Pode-se, ainda, usar ferramentas para análise dos arquivos de *tracing* gerados durante a simulação, que são os verdadeiros resultados da simulação, estes arquivos com formatação específica registram cada evento gerado pelos escalonadores. As ferramentas para análise dos arquivos de *tracing* devem então ser capazes de ler os dados gravados nestes arquivos e efetuar os cálculos desejados. Uma destas ferramentas, muito utilizada, é o *awk* [Brown], uma linguagem desenhada para buscar padrões dentro de um arquivo e efetuar

ações programadas. O animador *nam* pode também ser usado para analisar visualmente a simulação e obter algumas estatísticas, mas ele não é apropriado para análises mais profundas. Se nada for feito, o simulador apenas rodará o *script*, gerará os arquivos de saída (*tracing*) e encerrará, sem nada mostrar ao usuário. Em primeiro momento, esta característica um tanto não-amigável e não-imediatista do NS pode frustrar o usuário iniciante. Entretanto, essa é a realidade de bom número dos simuladores existentes na categoria do NS, que não foram feitos com ótica didática.

2.2.2 Simulação de um Modelo

O modelo proposto para simulação, apresentado na topologia ilustrada na Figura 2.6, contém dois geradores de tráfego, um do tipo CBR (*Constant Bit Rate*) [Ferguson (1998)] e outro do tipo exponencial *on-off*. Todos os nós têm filas de saída tipo *droptail*, com exceção do nó 2, que recebe uma fila tipo *Stochastic Fair Queueing* (SFQ) [Ferguson (1998)]. O nó 3 receberá o tráfego dos geradores. O gerador exponencial estará no nó 0 e enviará seus dados para o *host* 3, através do nó 2. O gerador CBR estará no nó 1 e enviará para o nó 3, também através do nó 2.

O primeiro passo é a criação do objeto simulador, que é feita da seguinte maneira:

```
set ns [new Simulator]
```

No modelo proposto foi estipulado o tempo máximo de simulação:

```
set MAX_TIME 500
```

Após a criação do objeto simulador, abre-se os arquivos *tracing* para posterior análise:

```
set nf [open out.nam w]
set tr [open out.tr w]
$ns trace-all $tr
$ns namtrace-all $nf
```

O primeiro arquivo (*out.nam*) conterá o registro de eventos simulados no formato para leitura do animador *nam*, e o segundo arquivo (*out.tr*) conterá os registros dos eventos de maneira mais completa, permitindo análises profundas da simulação.

A topologia da rede é criada da seguinte maneira:

```
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $n0 $n2 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 10Mb 10ms DropTail
$ns duplex-link $n2 $n3 10Mb 10ms SFQ
```

Nas quatro primeiras linhas do código acima são criados os nós. Configura-se então a ligação entre estes nós, com *links* do tipo *duplex*, cada um com largura de banda de 10Mbits e atraso de propagação de 10 ms.

A *procedure* abaixo cria uma 4ª. Camada do tipo UDP, um gerador de tráfego exponencial *on-off* e faz a conexão de ambos, entre os nós 0 e 3. O manual do NS disponível em [NS-2] traz mais detalhes sobre a configuração dos geradores de tráfego. A *procedure* também define uma cor para uso do *nam*.

```
Proc attach-expoo-traffic {node sink size burst idle rate class color} {
        set ns [Simulator instance]
        set source [new Agent/UDP]
        $ns attach-agent $node $source
        $source set class_ $class
        $ns color $class $color
        set traffic [new Application/Traffic/Exponential]
        $traffic set packetSize_ $size
        $traffic set burst_time_ $burst
        $traffic set idle_rate_ $idle
        $traffic set rate_ $rate
        $traffic attach-agent $source
        $ns connect $source $sink
        return $traffic
}
set traffgen0 [attach-expoo-traffic $n0 $sink0 1000 800ms 2ms 5M 1 Green]
```

Para o nó 1, as linhas abaixo criam a 4ª camada tipo UDP e um gerador de tráfego CBR, e também a conexão destes entre os nós 1 e 3.

```
Set udp1 [new Agent/UDP]
$udp1 set class_ 2
$ns attach-agent $n1 $udp1
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 1000
$cbr0 set interval_ 0.005
```

```
$cbr0 attach-agent $udp1
$ns connect $udp1 $sink0
```

As linhas abaixo trazem a programação do escalonador de eventos:

```
$ns at 0.0 "$traffgen0 start"
$ns at 1.0 "$cbr0 start"
$ns at [expr $MAX_TIME-1] "$traffgen0 stop"
$ns at [expr $MAX_TIME-1] "$cbr0 stop"
$ns at $MAX_TIME "finish"
```

As primeiras quatro linhas definem em que intervalos de tempo cada gerador de tráfego deve iniciar e parar suas atividades. A expressão *[expr $MAX_TIME-1]* significa o tempo total da simulação menos um segundo, que é quando o tráfego interrompe suas atividades.

O código abaixo define o processo de finalização da simulação, chamado pelo escalonador. Os arquivos de *trace* são fechados e mostram-se algumas estatísticas coletadas durante a simulação (através do objeto *LossMonitor*). O visualizador *nam* também é executado aqui. A Figura 2.6 traz o *nam* em ação.

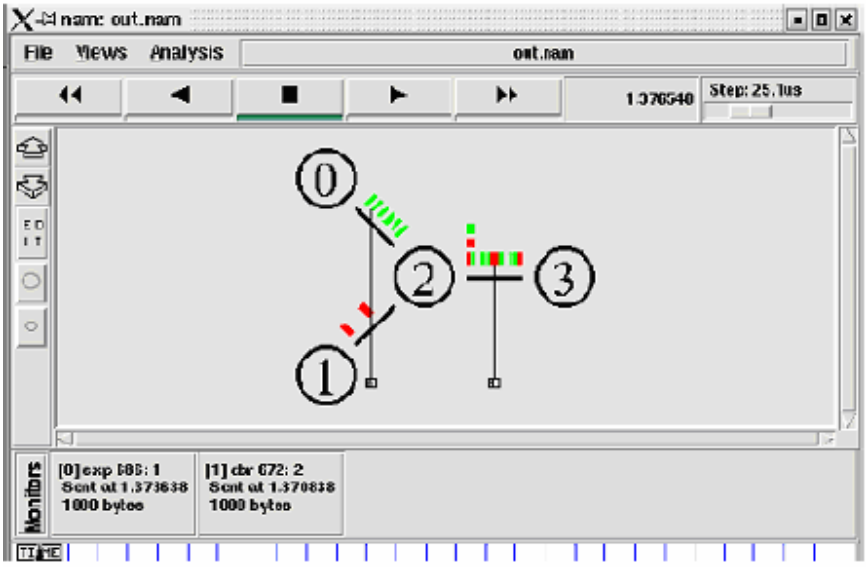

Figura 2.6 - Tela do *nam* executando a simulação

Fonte: [Marcos]

```
Proc finish {} {
global ns nf tr MAX_TIME sink0
$ns flush-trace
close $nf
close $tr
set now [$ns now]
puts "Estatisticas:"
puts "Tempo Simulacao: $now s"
puts "Pacotes recebidos no nodo 3: [$sink0 set npkts_]"
puts "Bytes recebidos no nodo 3: [$sink0 set bytes_]"
puts "Utilizacao do link: [expr [$sink0 set bytes_ ]*8. / (10000000. *
$now) *100.]%"
exec nam out.nam &
```

```
exit 0
}
```

A ultima linha do script, finalmente, inicia a execução do simulador.

```
$ns run
```

Para este modelo e parâmetros utilizados, o *script* forneceu os seguintes resultados:

Estatisticas
Tempo Simulacao: 500 s
Pacotes recebidos no nodo 3: 384377
Bytes recebidos no nodo 3: 334576500
Utilizacao do link: 53.532239999999994%

2.2.3 O Modelo *Wireless* no NS

O modelo *wireless* consiste essencialmente no *MobileNode* (mobilenode.h), com características suportando adicionais que permite simulações de redes ad hoc multi-hop, LANs *wireless* etc. Um nó móvel é derivado do nó básico o *Node*(node.h) com funcionalidades adicionadas de um nó *wireless* e mobilidade como a habilidade de mover-se dentro de uma dada topologia, habilidade de receber e transmitir sinais a partir de uma canal *wireless* etc. Uma diferença principal entre eles, é que um *host* móvel não está conectado por meio de links a outros *hosts* ou *host* móvel [NS-2] .

Na criação de um objeto *MobileNode* é especificado a criação de um agente de roteamento, cria a pilha de rede consistindo de uma camada de enlace (*link layer*), camada mac e uma interface de rede com uma antena, interconectando

estes componentes e conectando a pilha ao canal de comunicação, ilustrado na Figura 2.7.

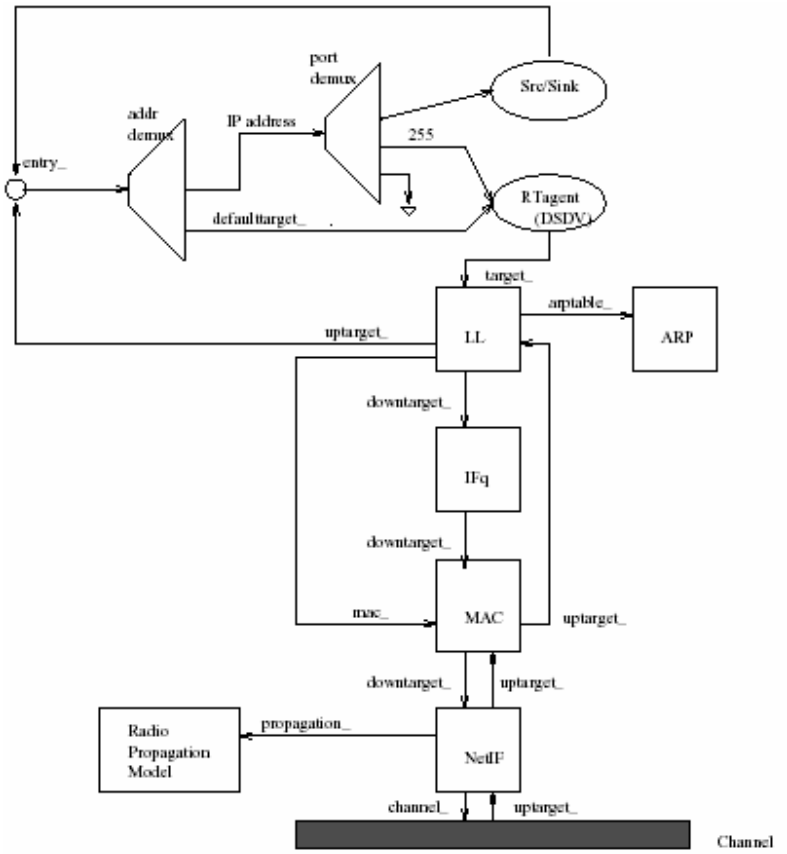

Figura 2.7- Esquema de um *MobileNode*
Fonte: [NS-2]

2.2.4 IP Móvel no NS

O cenário do IP Móvel no NS consiste de *home agents* e *foreign agents* e tendo *hosts* moveis se movimentando entre eles. O *home agent* e *foreign agent* são essencialmente estações base, enquanto os *hosts* móveis são basicamente

MobileNode descritos acima. Os métodos e procedimentos para a extensão do IP Móvel estão descritos em mip.cc e mip.h, mip-reg.cc, ns-mip.tcl e ns-wireless-mip.tcl[NS-2].

O *home agent* e *foreign agent* têm um agente registrador (reg_agent) como ilustrado na Figura 2.8 que envia sinais para os *hosts* móveis quando necessário e responde as solicitações dos *hosts* moveis que por sua vez também possuem um agente registrador que recebe e responde as solicitações do *home agent* ou *foreign agent*.

Os *agents* (*home* e *foreign)* enviam *broadcasts* ou mensagens de advertência para os *hosts* móveis. O endereço usado pelas estações base para enviar os sinais é o *care-of address* do *host* móvel, que é alterado sempre que ele migra para outro domínio.

Quando um pacote é destinado ao *host* móvel o *home agent* verifica se o *host* esta em seu domínio através do seu *care-of address*, caso o *care-of address* não seja compatível com o endereço do *home agent* ele ativa o encapsulador e envia através da técnica de tunelamento para o destino instanciado pelo *care-of address* do *host* móvel (endereço do *foreign agent*). O *foreign agent* recebe o pacote (decapsulator), remove o encapsulamento e envia para o *host* móvel. Caso o *host* esteja no seu domínio o pacote é enviado diretamente [NS-2].

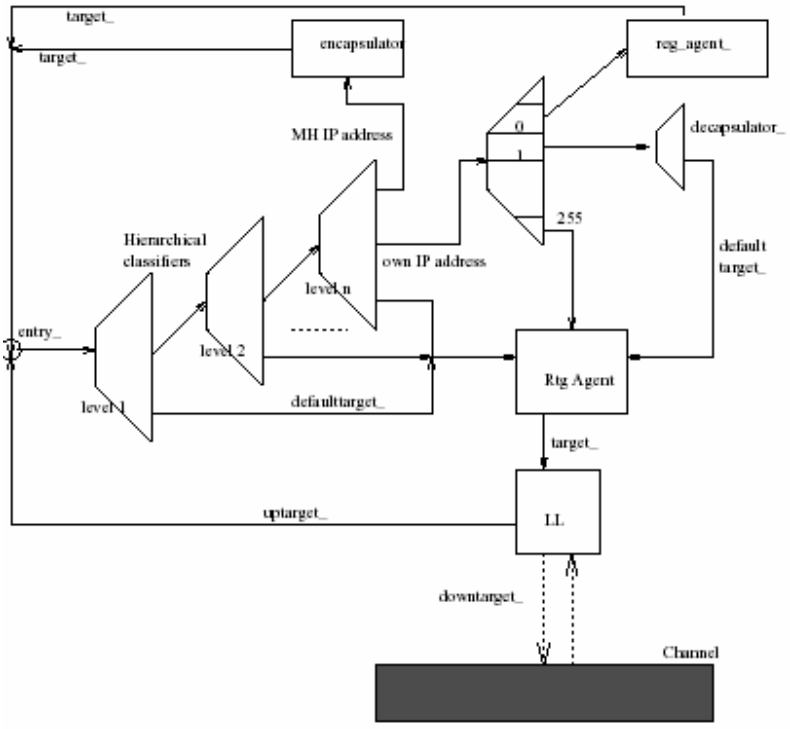

Figura 2.8: Esquema de uma estação base *wireless*
Fonte: [NS-2]

Capítulo 3

Metodologia

3.1 Tipo de Pesquisa

O objetivo deste trabalho é simular o envio de pacotes para um hot móvel em uma rede sem fio utilizando a ferramenta de simulação *Network Simulator* (versão 2.26) e o IP Móvel como protocolo de transferência de dados entre os dispositivos móveis.

O trabalho foi dividido em três partes, ou seja, três situações diferentes de simulação, sendo cada uma delas detalhadas a seguir:

Roteamento de pacotes para um *host* móvel em seu *home link*

Pacotes destinados para o *home address* de um *host* móvel são roteados para o *home link* do *host* móvel como resultado de um funcionamento normal de um roteamento por prefixo de rede. Desta forma, não são necessários procedimentos de roteamento especiais para a entrega de pacotes para um *host* móvel que está conectado no seu *home link*. Isto implica que as regras para o roteamento de pacotes são idênticas às regras de roteamento de pacotes de qualquer host IP ou roteador convencional.

Além disso, não são necessárias regras especiais para rotear pacotes gerados pelo *host* móvel quando o mesmo está conectado ao seu *home link*. Como qualquer outro *host* ou roteador, cada host móvel utiliza sua própria tabela de roteamento para selecionar o próximo *hop* apropriado para cada pacote

gerado. Similarmente, nenhuma regra especial é necessária para gerar ou atualizar as entradas na tabela de roteamento de um *host* móvel enquanto o mesmo está conectado em seu *home link*.

Roteamento de pacotes para *hosts* móveis conectados no forein link

O procedimento para o roteamento de pacotes para um *host* móvel que está conectado no *foreign link* está ilustrado na Figura 3.1 e pode ser sumarizado da seguinte forma:

1. Um roteador no *home link*, possivelmente o *home agent*, notifica alcance para o prefixo de rede igual àquele do *home address* do host móvel;
2. Pacotes destinados ao *home address* do *host* móvel são, portanto, roteados em direção ao seu *home link* e, especificamente, em direção ao *home agent* do *host* móvel;
3. O *home agent* intercepta os pacotes destinados para o *host* móvel, assumindo que o *host* móvel registrou um ou mais *care-of address* e envia uma cópia, via túnel, para cada *care-of address*;
4. Em cada *care-of address*, um endereço de um *foreign agent* ou um endereço *collocated* dentro do próprio *host* móvel, o pacote original é extraído do túnel e entregue para o *host* móvel.

Os dois casos, um *foreign agent care-of address* e um *collocated care-of address*, o *home agent* recebe um pacote destino a um de seus *hosts* móveis e procura pelo *binding* correspondente. O *home agent* então envia o pacote pelo túnel para o *care-of address*, não se importando se o *host* móvel ou o *foreign agent* é o ponto de saída do túnel. Em ambos os casos, o pacote é originado no *host*, e direcionado para o *home address* do *host* móvel e também em ambos os casos o pacote encapsulado é enviado do *home agent* para o *care-of address*.

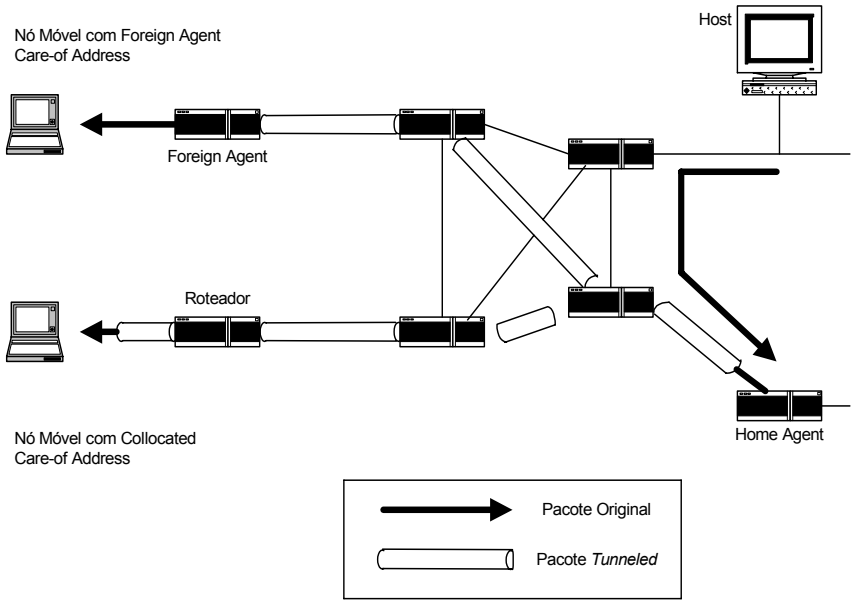

Figura 3.1 - Roteando pacotes para um *host* móvel em um *foreign link*

Fonte: Dados da Pesquisa

Roteamento de pacotes para *hosts* móveis migrando de um forein link para outro (*handoff*)

A função que permite manter a continuidade da comunicação quando um usuário passa de uma célula para outra, é conhecida como *handoff* [Alencar (1998)].

O roteamento de pacotes para *hosts* em movimento pode ser resumido da seguinte forma como ilustrado na Figura 3.2.

1. O host ao migrar para um novo *forein link* notifica o *foreign agent* anterior (auxiliará o *host* no processo de transição entre os *forein links)*, para atualizar o seu *care-of address* (novo endereço do *foreigen agent*);

2. O *foreign agent* anterior intercepta os pacotes enviados ao host móvel e o reencapsula com o novo *care-of address*, para não haver perdas de pacotes durante a transição;
3. O *foreign agent* atual envia uma mensagem para o *home agent* para atualizar o seu registro (*care-of address* atual);
4. O *foreign agent* anterior fica encarregado por essa transmissão até o novo registro ser efetivado.

Após o registro, o roteamento será idêntico ao roteamento de um *host* conectado no *forein link*.

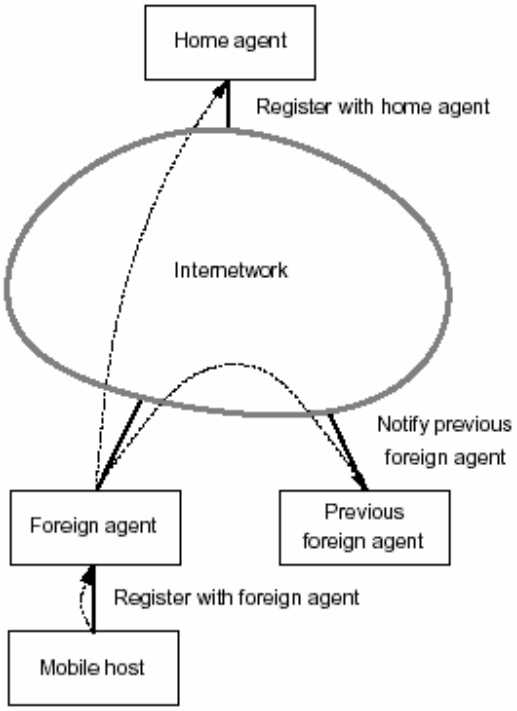

Figura 3.2 - Registro durante o *handoff*.

Fonte: [Alencar (1998)]

3.2 Implementação

A implementação de simulações no *Network Simulator* foi realizada na linguagem OTcl utilizando o paradigma de orientação a objetos, para simplificar o projeto foi feito o uso das bibliotecas de rede sem fio e IP Móvel disponíveis no pacote do NS, este pacote esta disponível em http://www.isi.edu/nsnam/dist/ns-allinone-2.26.tar.gz para *download*, é gratuito e com código aberto, existe em diversas plataformas, mas com melhor desempenho em ambiente UNIX, utilizou-se a distribuição Red Hat 7.3 do sistema operacional Linux para o desenvolvimento destas simulações.

Capítulo 4

Resultados e Discussões

4.1 Implementação

A simulação do IP móvel através do *Network Simulator* foi totalmente implementada na linguagem OTcl, denominada de projeto.tcl (Anexo A). Este arquivo está com o código todo escrito em inglês para futuras utilizações. Segue abaixo uma detalhada explicação de todos os procedimentos e objetos utilizados nesta implementação.

```
set opt(mac)     Mac/802_11
```

O objeto MAC simula o protocolo de acesso que é necessário para dividir cada ambiente (*wireless/ wired*) em uma área de rede local.

Para enviar, o objeto MAC é responsável por adicionar o cabeçalho MAC e transmitir o pacote pelo canal, ele recebe simultaneamente pacotes partindo do classificador da camada de ligação. Quando recebe um pacote do canal, o objeto MAC se responsabiliza por enviar o pacote de dados para a camada de ligação. A classe MAC802_11 está implementada em 802_11.{cc,h} no pacote do NS.

```
set opt(chan)    Channel/WirelessChannel
```

O canal simula a transmissão de pacotes na camada física. Ele deixa o objeto MAC tomar conta da contenção e detectar colisões de pacotes, se mais de

uma transmissão ultrapassa no tempo, o canal incrementa o *flag* de colisão. Checando este *flag* o objeto MAC pode detectar e cuidar das colisões [NS-2].

```
set opt(netif)  Phy/WirelessPhy
```

A ligação da interface de rede serve como uma interface de hardware na qual é usada pelo host móvel para acessar o canal. A interface de divisão de mídia *wirelles* está implementada como classe Phy/WirelessPhy [NS-2].

Esta interface trata de colisões e recebimento de pacotes através de um modelo de propagação de rádio transmitido por uma interface de outro *host* para o canal. A interface marca cada pacote transmitido com um meta-dado que revela a interface de transmissão alguns dados, tais como, o poder de transmissão, o comprimento de onda, etc [NS-2]. Este meta-dado que está presente no cabeçalho do pacote é utilizado pelo modelo de propagação na interface de rede de recebimento, para determinar se o pacote tem o poder mínimo para ser recebido e/ou capturado e/ou descartado pelo *host* receptor [NS-2]. O modelo se aproxima a interface de rádio DSSS (*Lucent WaveLan direct-sequence-spread-spectrum*) e está implementado em phy.{cc,h} e wireless-phy.{cc,h}.

Estes modelos são usados para predizer o poder do sinal de recebimento de cada pacote. Na camada física de cada *host wireless*, há um limiar de recebimento. Quando um pacote é recebido, se seu poder de sinal for inferior ao limiar de recebimento, ele é marcado com erro e descartado pela camada MAC [NS-2].

```
set opt(ll)     LL
```

A camada de ligação é responsável por simular o protocolo de ligação de dados e preencher o endereço MAC de destino no cabeçalho MAC do pacote. Nesta atual implementação esta tarefa envolve duas instâncias: achar o endereço

IP do próximo *hop* do *host* e através deste endereço IP descobrir o correto endereço MAC (ARP). Nos *hosts* móveis o modulo ARP resolve todas conversões do endereço IP para o endereço de hardware (MAC). A classe LL está implementada em ll.{cc,h} e ns-ll.tcl no pacote do NS.

```
set opt(ifq)     Queue/DropTail/PriQueue
```

A interface de fila representa o local onde os pacotes poderão ser guardados (ou descartados).O planejamento de pacotes refere-se ao processo de decisão usado para escolher qual pacote será ou não utilizado. O gerenciamento do buffer refere-se a um tipo particular de fila usado para ajustar a ocupação da mesma [NS-2].

Neste trabalho o tipo de fila utilizado é o *drop-tail* (classe derivada da classe *Queue*) sendo implementada com planejamento FIFO e gerenciamento de buffer com perda de pacotes em caso de falta de memória (*drop-on-overflow*) característica presente na maioria dos roteadores da Internet hoje em dia [NS-2].

A classe *PriQueue* (classe derivada da classe *DropTail*) é implementado como uma fila que da preferência aos pacotes roteados, inserindo-os na cabeça da fila. Ela também é responsável por executar um filtro sobre todos os pacotes da fila e remover os pacotes que tiverem os endereços de destino especificados em alguma requisição. A classe *PriQueue* está implementada em *priqueue.*{cc,h} no pacote do NS.

```
set opt(prop)    Propagation/TwoRayGround
```

Neste trabalho é utilizado o modelo de propagação *Two-Ray Ground* que considera ambas as direções e a reflexão no terreno do caminho, com isso, este modelo mostra maior precisão em longas distâncias do que outros modelos, como o *Free-space*. Esta classe está implementada em *tworayground.*{cc,h} no pacote do NS.

```
set opt(adhocRouting)   DSDV
```

Agente roteador DSDV (*Destination Sequence Distance Vector*). Neste protocolo de roteamento as mensagens são trocadas entre *hosts* móveis vizinhos. A atualização do roteador pode ser despertada ou rotineira. A atualização é despertada quando a informação do roteador de um vizinho força uma troca na tabela de roteamento.

Um pacote no qual o roteador para seu destino é desconhecido é armazenado (*cache*) até que uma consulta de roteamento esteja sendo feita ou até que uma réplica seja enviada para o destino. Há um tamanho máximo de buffer para armazenar os pacotes que estão aguardando por informações de roteamento, depois de preenchido o buffer, os pacotes são descartados [NS-2].

Todos os pacotes destinados aos *hosts* móveis são roteados diretamente pelo seu endereço *dmux* para a porta *dmux*. A porta *dmux* entrega os pacotes para o respectivo agente de destino. Nos *hosts* móveis é utilizada a porta 255 para anexar um agente roteador. Este protocolo é principalmente implementado em C++, suas classes estão no diretório dsdv e os procedimentos descritos acima estão implementados na classe dsdv.tcl, todas incluídas no pacote do NS.

```
$ns node-config -addressType hierarchical
```

Este comando configura o tipo de endereçamento do *host*.

O roteamento hierárquico é principalmente desenvolvido, entre outras coisas, para reduzir a requisição de memória para simulações com topologias muito grandes. A topologia é quebrada em várias pequenas camadas hierárquicas, reduzindo desta forma o tamanho da tabela de roteamento. O tamanho da tabela é reduzido de n^2, para roteamento horizontal, para cerca de $log\ n$ para roteamento hierárquico. Até a atual implementação este tipo de roteamento suporta no máximo 3 (três) níveis hierárquicos [NS-2].

Para ser capaz de utilizar este roteamento foi necessário definir a hierarquia da topologia, assim como os *hosts* com endereçamento hierárquico.

No roteamento horizontal, todos *hosts* sabem tudo sobre os outros *hosts* da topologia, assim o resultado é uma tabela na ordem de n^2. No roteamento hierárquico, cada *host* tem informação somente dos *hosts* que fazem parte do seu nível. Para todos os outros destinos fora de seu nível ele envia os pacotes para o roteador de fronteira. Com isso há uma redução no tamanho da tabela de roteamento para a ordem de aproximadamente *log n* [NS-2].

```
AddrParams set domain_num_ 3
lappend cluster_num 4 1 1
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 1 1 1 2 1
AddrParams set nodes_num_ $eilastlevel
```

A classe *AddrParams* é utilizada para armazenar a topologia hierárquica (níveis hierárquicos), como o número de áreas em cada nível (domínios, clusters e *hosts*) [NS-2].

Assim definimos a topologia com 3 (três domínios) , sendo D1, D2 e D3 , o domínio D1 possui 4 clusters (C11, C12, C13 e C14) ,o domínio D2 e D3 possui 1 cluster cada um, sendo os clusters (C21) e (C31) pertencentes a cada domínios respectivamente. Então o numero de *hosts* em cada um destes clusters são 1, 1, 1, 1, 2 e 1 respectivamente.

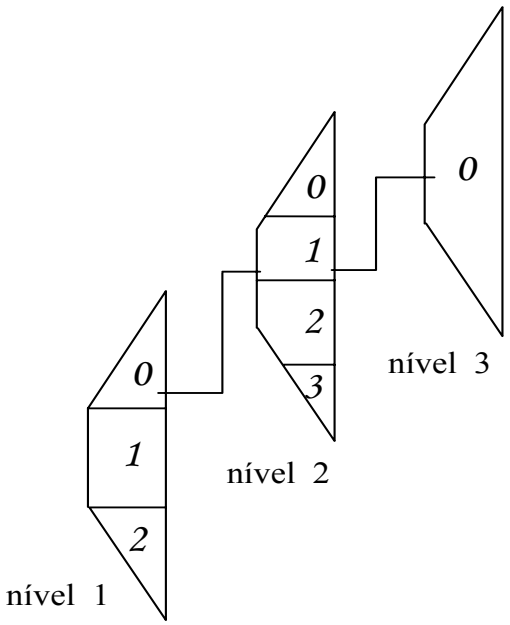

Figura 4.1 – Representação do *host* 0.1.0 (roteamento hierárquico)
Fonte: Dados da pesquisa

Na Figura 4.1 o nível 1 (um) significa o domínio, o nível 2 (dois) mostra todos os clusters dentro do domínio do *host* e o nível 3 (três) determina o *host* dentro de um particular cluster.

```
set namtrace [open projeto-out.nam w]
```

Nam é uma ferramenta de simulação baseada na linguagem Tcl/Tk que é usada para visualizar as simulações realizadas com o NS e também gerar um arquivo de dados sobre a simulação (*Nam trace file*) [NS-2].

O propósito por trás do *nam* é de criar um animador que seja capaz de ler grandes arquivos de animação e também ser flexível o bastante para visualizar qualquer situação de simulação de rede [NS-2]. Para gerar uma animação o *nam* armazena poucas informações na memória, a maioria dos comandos são escritos em um arquivo que são lidos conforme for necessário [NS-2].

O primeiro passo para se utilizar o *nam* é produzir um arquivo de traço. O arquivo de traço contém informações da topologia como os *hosts*, ligações, filas, conexões de cada *host*, informações de traço do pacote, etc. Normalmente este arquivo é gerado pelo NS.

Quando o arquivo de traço é gerado, ele está pronto para ser animado pelo *nam*. No início, o *nam* lerá o arquivo de traço, criará a topologia, uma janela do tipo pop-up e por fim construirá a interface, iniciando a animação no tempo 0 (zero) [NS-2]. O nam através de sua interface prove controles sobre vários aspectos da simulação como avançar, parar, retornar, acelerar, desacelerar a simulação, editar a posição dos *hosts*, etc.

```
$ns namtrace-all-wireless $namtrace $opt(x) $opt(y)
```

Este comando é utilizado para indicar ao arquivo de traço do *nam* de que ele está habilitado a inicializar a documentação (detalhamento) dos movimentos do *host* para serem visualizados no *nam*. As coordenadas X e Y da topologia *wireless* também é passada como parâmetro para este comando [NS-2].

```
$HA shape "hexagon"
$HA color "red"
$ns at 0.0 "$HA   label HA"
```

O *host* pode ter 3 (três) formas, (circulo, quadrado, hexágono), mas uma vez criado esta forma não poderá ser alterada durante a animação.Os *hosts* podem apenas trocar de cor dinamicamente [NS-2]. O nome das cores utilizado deverá ser obrigatoriamente um dos nomes listados no banco de dados de cores, o X11 (arquivo rgb.txt).

Pode-se atribuir também um rótulo (*label*) ao *host*, permitindo com isso apresentar uma animação mais organizada e com um grau maior de interpretação.

```
set tracefd  [open projeto-out.tr w]
$ns trace-all $tracefd
```

Este comando é usado para configurar o agente de traço. Todos os traços são escritos no arquivo projeto-out.tr.

O suporte de traço para simulações sem fio atualmente utiliza objetos cmu-trace. Estes objetos são utilizados para traçar os pacotes que são descartados ou perdidos, recebidos e enviados pelo agente, roteadores, camada MAC ou interface de pilha no NS. Os métodos e procedimentos utilizados pa implementar o suporte a traço em cenários *wireless* podem ser encontrados como trace.{cc,h} e ns-cmutrace.tcl no pacote do NS .

```
set topo   [new Topography]
```

Este comando cria uma instancia da topografia [NS-2].

```
$topo load_flatgrid $opt(x) $opt(y)
```

Este comando define as fronteiras da topologia [NS-2].

```
create-god [expr $opt(nn) + $num_bs_nodes]
```

Este comando é utilizado para criar um instancia God. O número de *host*s móveis é passado como argumento, no qual é usado pelo God para criar a matriz de informações de conectividade da topologia (número de *hops* entre cada *host*) [NS-2].

```
set temp {0.0.0 0.1.0 0.2.0 0.3.0}
for {set i 0} {$i < ($num_wired_nodes)} {incr i} {
    set WD($i) [$ns node [lindex $temp $i]]
}
```

Este comando cria e retorna uma instancia do *host*, neste caso o *host* criado tem endereçamento hierárquico [NS-2] representado pela *array* temp. Estes *hosts* são criados antes da configuração do nó (detalhada abaixo), fazendo com que eles não sejam hosts sem fio.

```
$ns node-config -mobileIP ON \
                -adhocRouting $opt(adhocRouting) \
                -llType $opt(ll) \
                -macType $opt(mac) \
                -ifqType $opt(ifq) \
                -ifqLen $opt(ifqlen) \
                -antType $opt(ant) \
                -propType $opt(prop) \
                -phyType $opt(netif) \
                -channel $chan_ \
                -topoInstance $topo \
                -wiredRouting ON \
                -agentTrace ON \
                -routerTrace OFF \
                -macTrace ON
```

A interface de configuração do *host* consiste de duas partes. A primeira parte detalha a configuração do *host* e a segunda parte na verdade cria o *host* como especificado [NS-2].

A configuração consiste essencialmente de definir as diferentes características do *host* antes de criá-los. Nesta configuração pode-se definir o tipo de endereçamento utilizado na simulação, definir os componentes de rede para *hosts* móveis, habilitar ou desabilitar as opções de traço como agente/

roteadores/ níveis MAC, selecionar o tipo de protocolo de roteamento para *hosts wireless* ou definir seu modelo de energia [NS-2].

Nesta configuração o flag mobileIP foi ativado, habilitando o uso do IP móvel para as estações base e ao host móvel que serão criados a seguir.

`$HA random`	`motion 0`

O *flag random-motion* é utilizado para habilitar ou não movimento randômico de um *host* móvel, se o *flag* estiver habilitado (*random-motion* 1), destinos randômicos é atribuido ao *host* [NS-2].

```
$MH set X_ 5.0000000
$MH set Y_ 215.0000000
$MH set Z_ 0.0000000

$ns at 4.0000000 "$MH setdest 420.0000000 200.0000000
20.0000000"
```

O *host* móvel é planejado para mover em uma topologia de 3 (três) dimensões, no entanto a 3ª (terceira) dimensão (Z) não é utilizada. Isto é, o *host* móvel é instruído a mover-se sempre no plano terrestre com Z sempre igual a 0 (zero). Sendo assim o *host* móvel tem as coordenadas X, Y e Z(0) que são continuamente ajustadas para determinar seu movimento. Há dois mecanismos para induzir o movimento em um *host* móvel. O 1º (primeiro) método é o movimento randômico que não é utilizado nesta simulação e o 2º (segundo) método atribui ao *host* uma determinada posição inicial e uma futura posição de destino, como exemplificado acima na função *setdest* que configura o destino e a velocidade de locomoção (m/s) do *host* móvel [NS-2].

```
set HAaddress [AddrParams addr2id [$HA node-addr]]
```

```
[$MH set regagent_] set home_agent_ $HAaddress
```

Este comando define o *home agent* (HA) do *host* móvel e isso é informado ao agente de registro (regagent_).

O *home agent* (HA) e o *foreign agent* (FA) tem um agente de registro (regagent_) que envia sinais para os *hosts* móveis, ativa o encapsulador e o desencapsulador, como também exige e responde as solicitações partidas dos *hosts* móveis (MHs). Os MHs também possuem um agente de registro que recebe e envia aos sinais e envia solicitações para o HA e o FA [NS-2].

```
$ns duplex-link $WD(0) $WD(3) 5Mb 2ms DropTail
```

Este comando cria uma ligação bi-direcional entre o *host* WD(0) e o *host* WD(3). Este procedimento essencialmente cria um ligação dupla partindo de duas ligações simples, uma partindo do WD(0) para o WD(3) e outra partindo do WD(3) para o WD(0). Defini-se também a largura de banda (5Mb), o atraso (2ms) e o tipo de fila utilizado na ligação (DropTail) [NS-2].

```
$ns duplex-link-op $WD(1) $WD(3) orient up
```

Este comando é utilizado para modificar alguns atributos da ligação como a orientação, a cor, o rótulo e a posição da fila [NS-2].

```
set tcp [new Agent/TCP]
```

Este comando cria uma instancia de um agente de envio TCP, este agente tenta capturar os comportamentos essenciais do TCP como controle de erro e congestionamento, mas não tem a intenção de ser uma replica completa da implementação real do TCP [NS-2].

```
set sink [new Agent/TCPSink]
```

Este comando cria uma instancia de um agente de recebimento TCP, este agente é responsável por retornar os ACKs para os objetos fontes TCP e o tamanho do ACK pode ser configurado [NS-2].

```
$ns attach-agent $WD(0) $tcp
$ns attach-agent $MH $sink
```

Este comando anexa o agente de envio tcp ao *host* WD(0) e o agente de recebimento ao *host* MH, terminando portanto a configuração de uma conexão fim-a-fim [NS-2].

```
$ns connect $tcp $sink
```

Este comando ativa a conexão entre os agentes *tcp* e *sink* [NS-2].

```
set ftp [new Application/FTP]
$ftp attach-agent $tcp
```

O agente TCP não gera nenhuma aplicação de dados por si próprio, portanto deve-se utilizar alguma aplicação para gerar os dados. Neste trabalho optou-se por uma aplicação FTP, devido a este tipo de simulação gerar um enorme trafego de dados [NS-2].

Estes comandos criam uma aplicação (FTP) e anexa ela ao agente de envio tcp respectivamente[NS-2].

```
$ns initial_node_pos $MH 12
```

Este comando define a posição inicial do *host* e o seu tamanho no *nam* [NS-2].

```
$ns at 4.0  "$ns trace-annotate \"mobile node goes to
     FA's dominion\""
```

Este comando insere uma observação, esta observação aparece na janela do *nam* e é utilizada pelos eventos para controlar a animação [NS-2].

4.2 Topologia

A topologia utilizada nesta simulação é composta por 4 (quatro) dispositivo*s*, nos quais representam uma rede local (LAN), com o propósito de ser a fonte de envio de dados para o *host* móvel, duas estações base destinadas a trabalharem como home agent ou foreign agent e 1 (um) *host* móvel que é responsável pela movimentação necessária para a criação dos 3 (três) diferentes tipos de simulação (Figura 4.2).

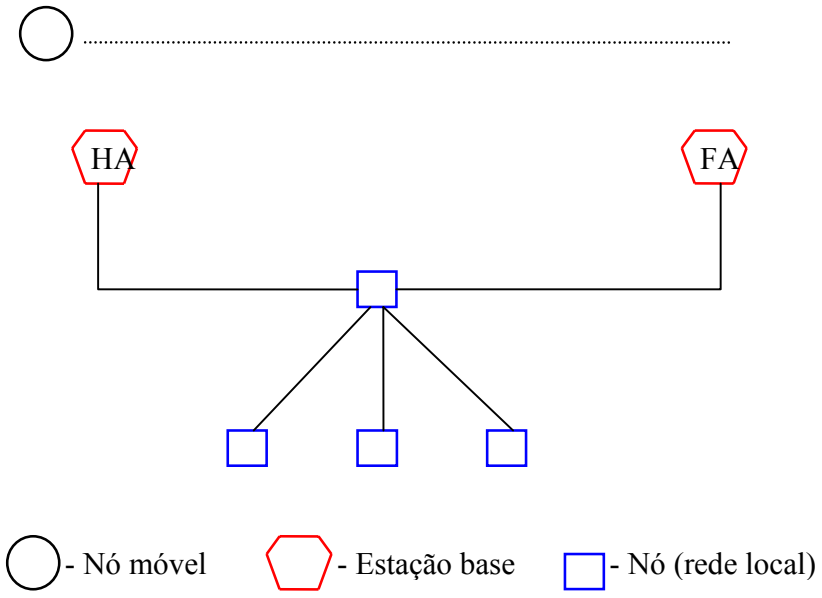

Figura 4.2 – Topologia da simulação
Fonte: Dados da pesquisa

4.3 Resultados da Simulação

As 3 (três) simulações foram implementadas no mesmo arquivo e simulada através de apenas uma animação, porém esta animação foi dividida através do tempo de simulação em 3 (três) etapas. Cada etapa corresponde a uma simulação. A primeira parte corresponde ao intervalo de 0.5s (meio segundo) a 7s (sete segundos) da animação e simula o envio de pacotes para o *host* móvel em sua rede local (*home link*), a segunda parte corresponde ao intervalo de 7s (sete segundos) a 24s (vinte e quatro segundos) da animação e simula o envio de pacotes para o host móvel durante a transição da sua rede local (*home link*) para uma rede externa (*foreign link*) e a última parte corresponde ao intervalo de 24s (vinte e quatro segundos) a 30s (trinta segundos) da animação, simula o envio de pacotes para o host móvel em uma rede externa (*foreign link*).

O NS gera após interpretar o código implementado em OTcl dois arquivos de traço, que podem ser analisados pelo usuário. Estas duas formas de saída utilizada pelo NS durante esta simulação serão descritas a seguir.

4.3.1 Nam

Através desta ferramenta de animação gráfica (*Nam*) pode-se visualizar perfeitamente toda a simulação. A figura abaixo ilustra a transferência de dados entre o *host* WD(0) e o *host* móvel depois de estabelecerem uma conexão TCP. A Figura 4.3 representa a primeira simulação proposta por este trabalho.

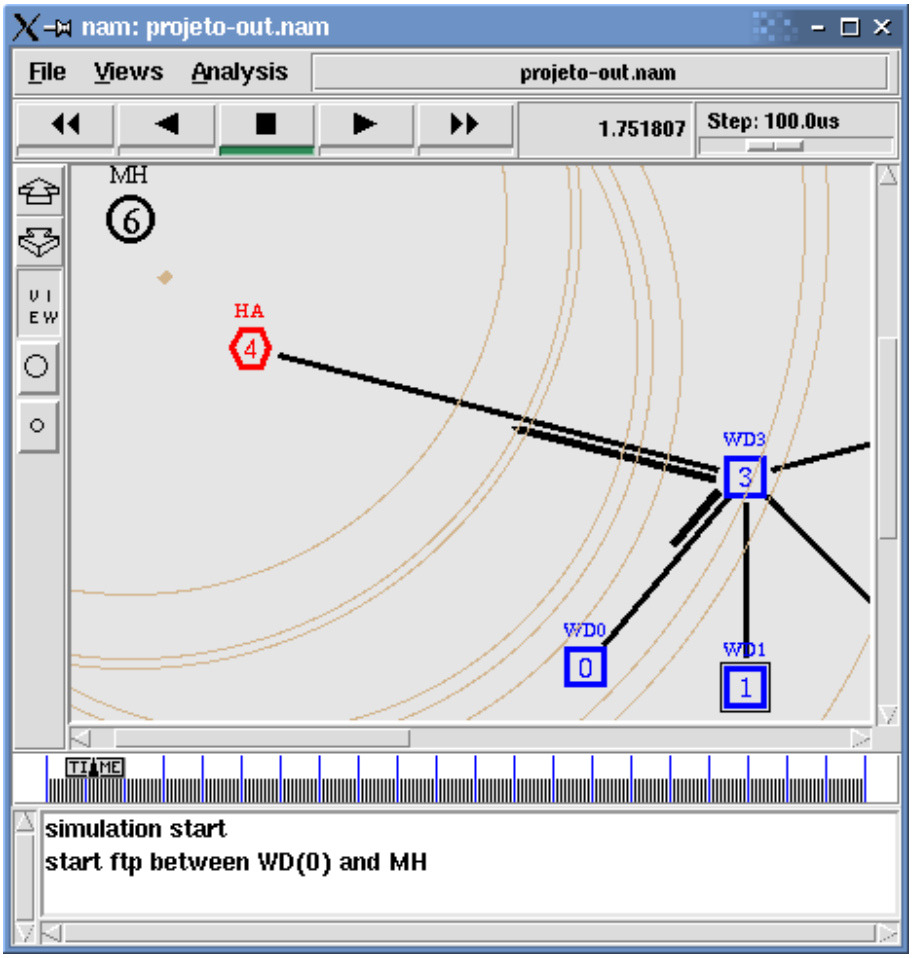

Figura 4.3 – *Host* móvel em sua rede local
Fonte: Dados da pesquisa

Na Figura 4.4 vemos que o *host* móvel já começou a se movimentar para fora de sua rede local (*home link*).

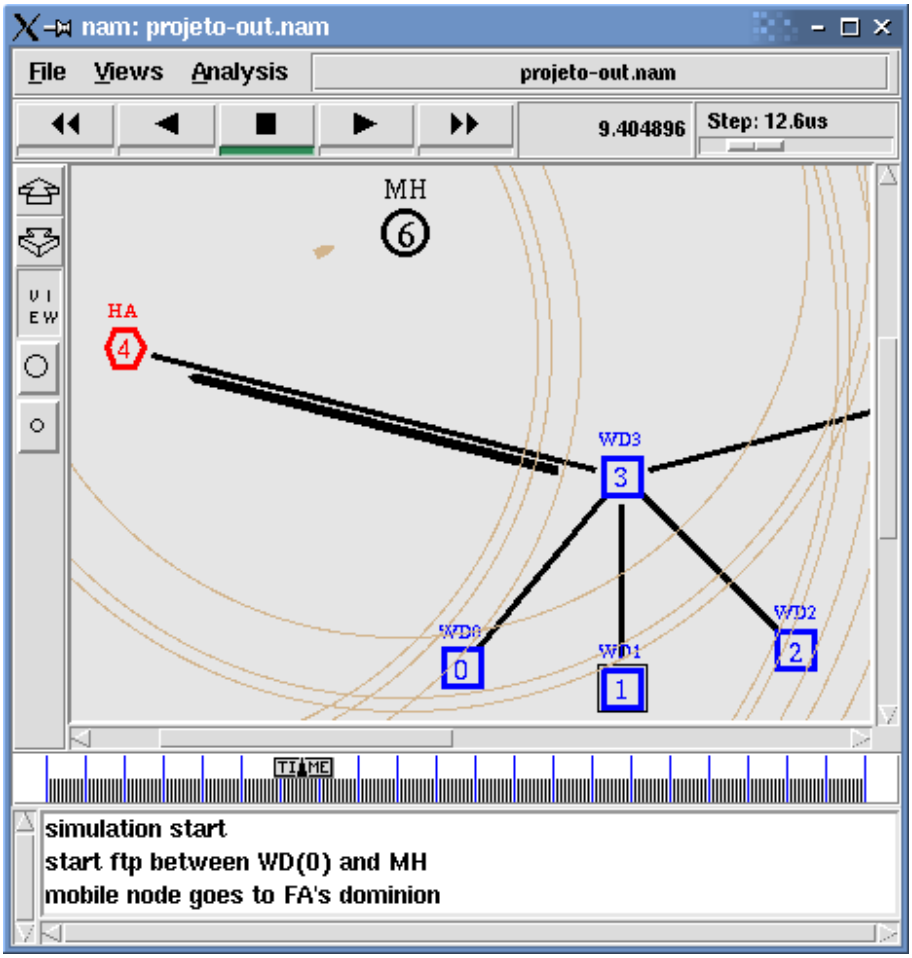

Figura 4.4 – *Host* móvel saindo da sua rede local
Fonte: Dados da pesquisa

Quando o *host* móvel atinge a fronteira de sua rede local, não conseguindo mais se comunicar com o seu *home agent*, a transmissão de dados é interrompida, ocorrendo algumas perdas de pacotes. Este fato é ilustrado pela Figura 4.5.

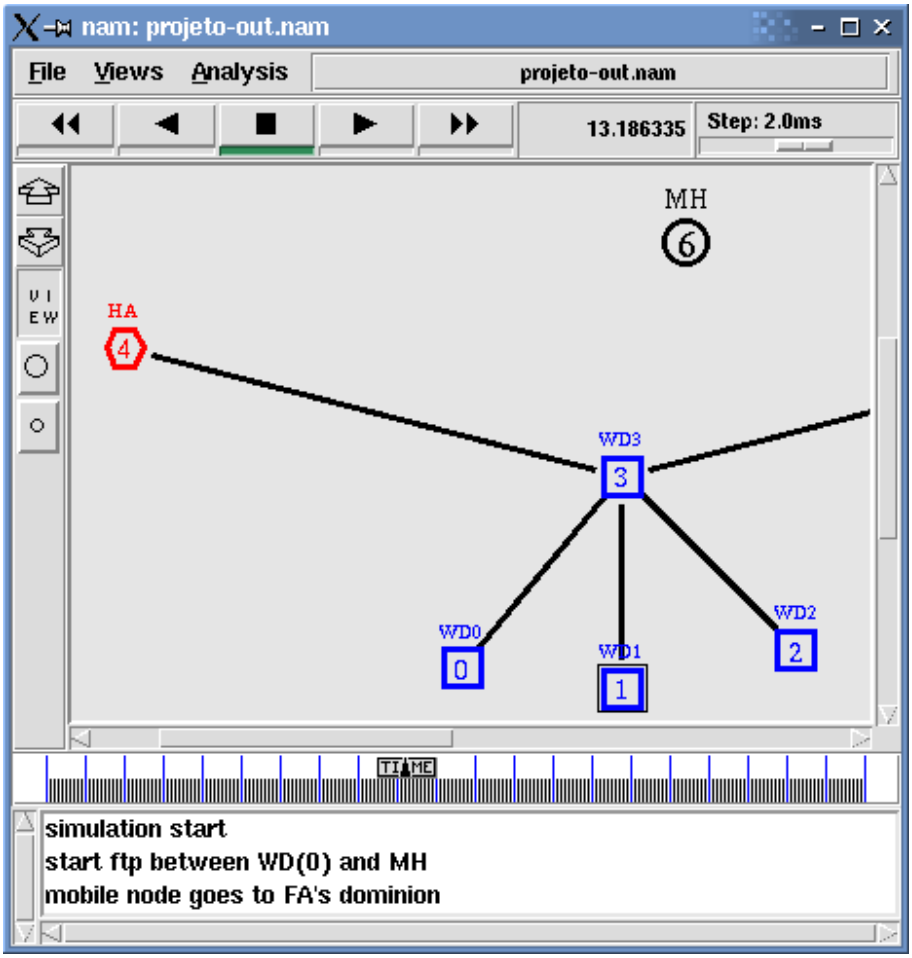

Figura 4.5 – *Host* móvel fora re área de cobertura
Fonte: Dados da pesquisa

Após certo tempo o *host* móvel atinge uma rede externa e inicia-se a comunicação entre o *host* móvel e o *foreign agent* (FA) e entre o *home agent* (HA) do *host* móvel e o *foreign agent* (Figura 4.6).

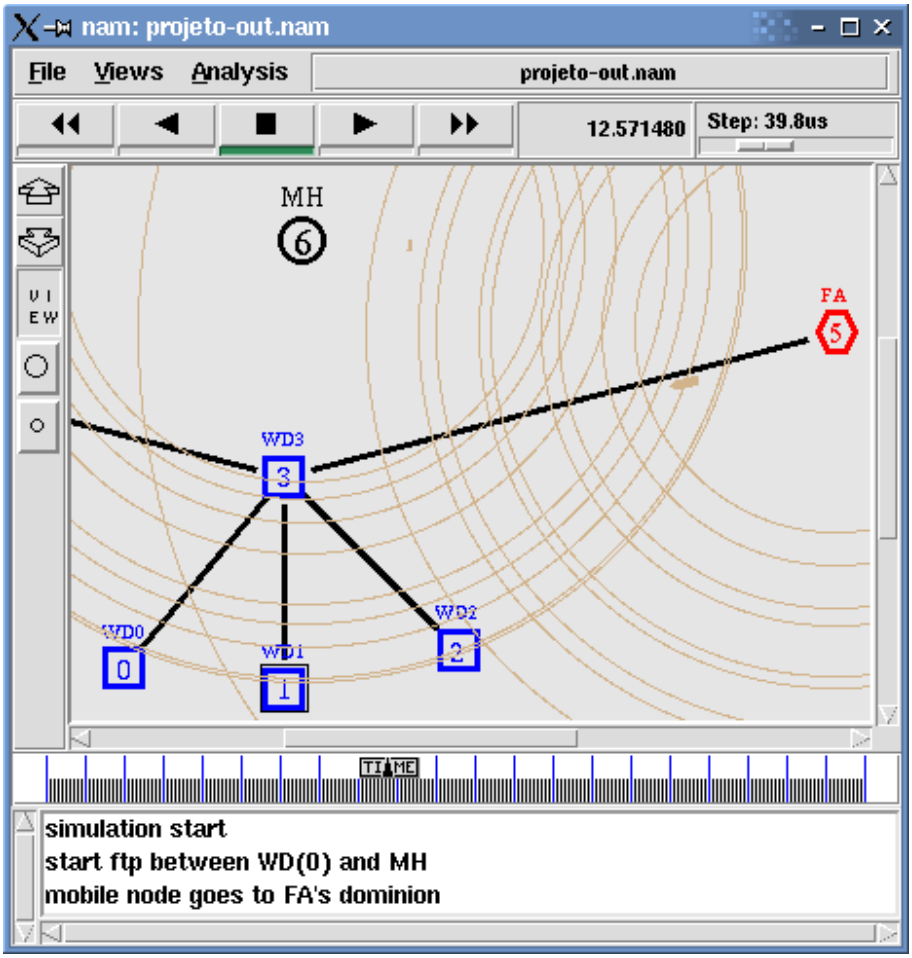

Figura 4.6 – *Host* móvel restabelecendo a conexão em uma rede externa
Fonte: Dados da pesquisa

Com a Figura 4.6 encerra-se representação da terceira simulação proposta por este trabalho.

Restabelecida a conexão entre o *host* móvel e o HA através do FA, retoma-se o envio de dados entre o *host* WD(0) e o *host* móvel através do encapsulamento do pacote feito pelo HA e o desencapsulamento e entrega do

pacote feita pelo FA (Figura 4.7 e Figura 4.8). Estas figuras representam a segunda simulação proposta por este trabalho.

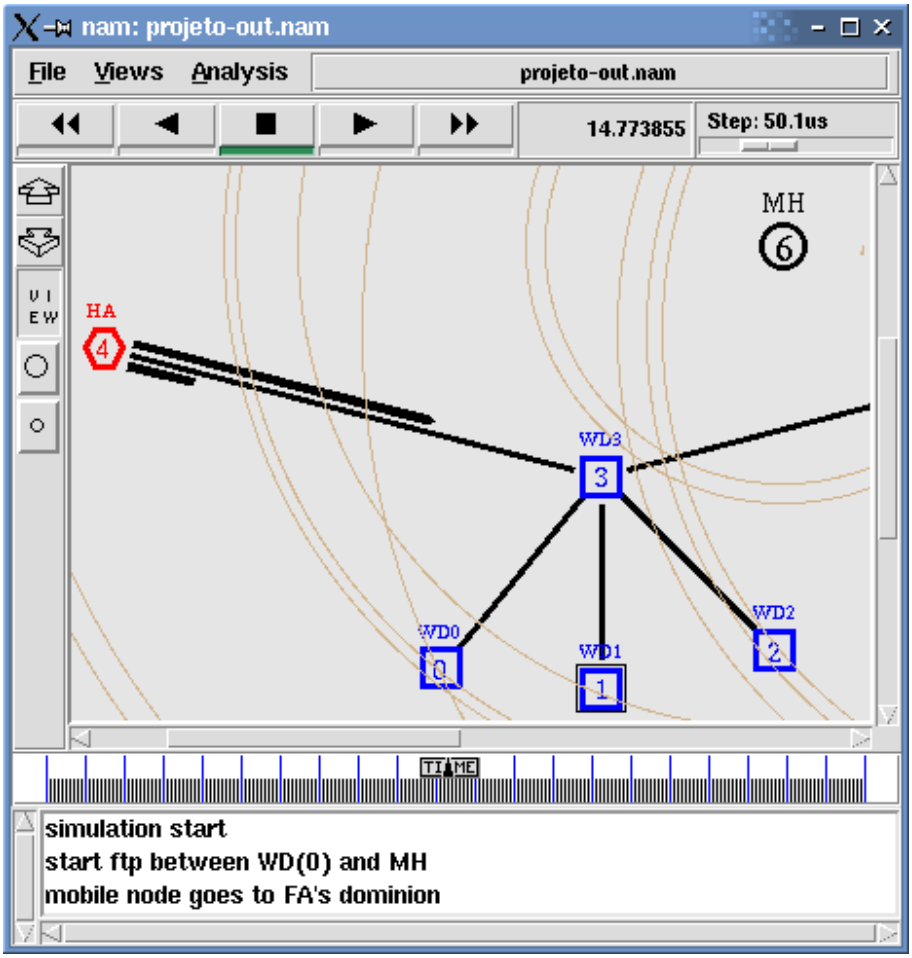

Figura 4.7 – HA enviando pacotes para o FA (encapsulamento)
Fonte: Dados da pesquisa

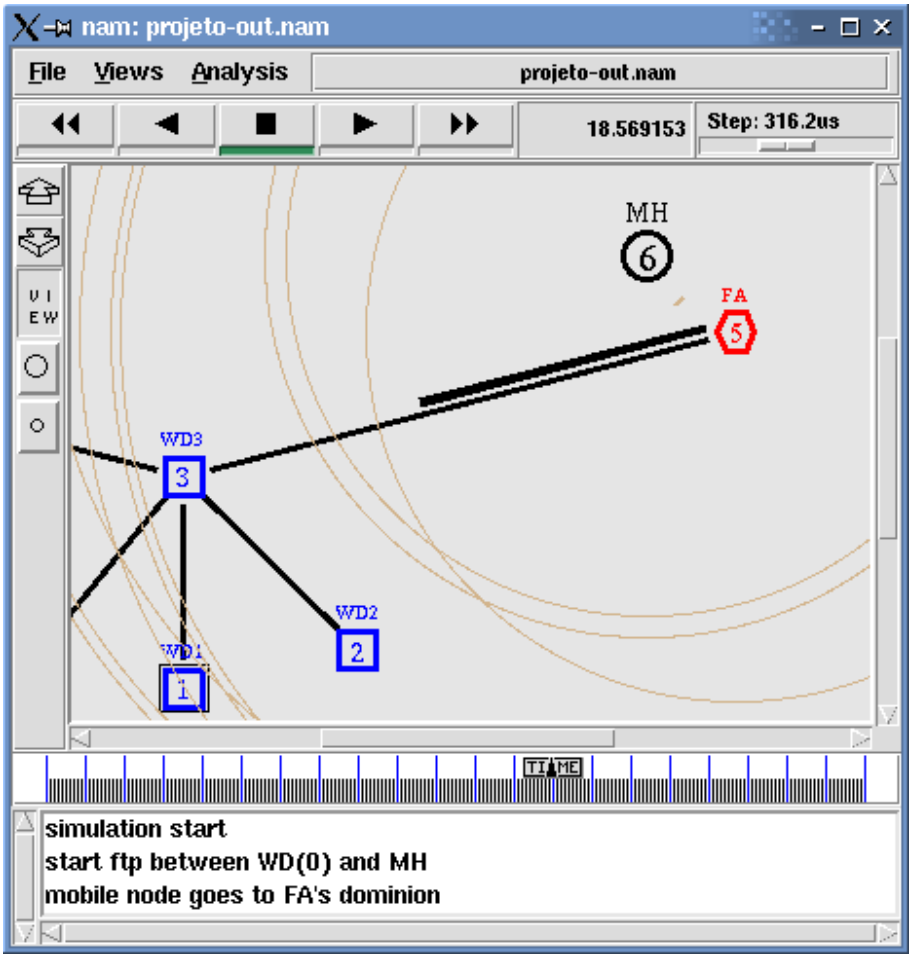

Figura 4.8 – FA enviando pacotes ao *host* móvel(desencapsulamento)
Fonte: Dados da pesquisa

4.3.2 Arquivo de traço

Este arquivo de traço construído pelo NS descreve toda troca de informação realizada pelos *hosts*, gerando com isso um arquivo muito extenso para ser inserido por completo neste documento, portanto será mostrado apenas alguns trechos de maior relevância.

As informações contidas no arquivo de traço referente a troca de dados em uma rede *wireless* é mais complexa comparado a troca de dados em uma rede wired. Estas informações estão dividas em campos que descrevem as propriedades do host, informações sobre o pacote no nível IP/ MAC/ aplicação, etc.

1. `r 0.502064 0 3 tcp 40 ------- 2 0.0.0.0 1.0.1.2 0 2`
2. `r 0.504128 3 4 tcp 40 ------- 2 0.0.0.0 1.0.1.2 0 2`
3. `s -t 1.000339577 -Hs 4 -Hd -1 -Ni 4 -Nx 70.00 -Ny 200.00 -Nz 0.00 -Ne -1.000000 -Nl AGT -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 4194304.0 -Id -1.0 -It udp -Il 48 -If 0 -Ii 6 -Iv 32`

Os 3 (três) primeiros itens descrevem o pedido para o estabelecimento de uma conexão TCP partindo do host 0.0.0 (WD(0)) para o host 1.0.1 (host móvel), seguindo o caminho WD(0) para o WD(3), WD(3) para o HA (4) e por fim do HA (4) para o MH (host móvel identificado pelo número de criação 6).

No pacote enviado pelo HA para o MH as siglas significam: s (envio), t (instante em que ocorreu a ação de envio/ recebimento/ perda/... do pacote) Hs e Hd significam o identificador do host fonte e o número de hops para alcançar o host destino respectivamente, Ni (identificador do host), Nx, Ny e Nz representam as coordenadas do host, Ne (o nível de energia do host), N1 (nível de traço agente/roteador/MAC) Nw (caso o pacote tenha sido perdido este campo explicaria o motivo), Ma (duração do pacote no nível MAC), Md e Ms (endereço ethernet do host destino e fonte), Mt (tipo de ethernet), Is (endereço e número da porta do host fonte separados pelo ponto), Id (endereço e número da porta do host destino separados pelo ponto), It (tipo de pacote), Il (tamanho do pacote), If (identificador de fluxo), Ii (identificador único) e por fim a sigla Iv (valor do tempo de vida do pacote).

4. `r -t 1.003462852 -Hs 6 -Hd -1 -Ni 6 -Nx 5.00 -Ny 215.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 0 -Md ffffffff -Ms 0 -Mt 800 -Is 4194304.0 -Id -1.0 -It udp -Il 68 -If 0 -Ii 6 -Iv 32`
5. `r 1.644708 4 3 ack 60 ----- 2 1.0.1.2 0.0.0.0 0 12`
6. `r 1.646804 3 0 ack 60 ----- 2 1.0.1.2 0.0.0.0 0 12`

Os itens 4 (quatro) a 6 (seis) descrevem envio da confirmação de recebimento do pedido de estabelecimento de conexão pelo host móvel ao host WD(0).

7. `r 1.650468 0 3 tcp 1040 ---- 2 0.0.0.0 1.0.1.2 1 13`
8. `r 1.654132 3 4 tcp 1040 ---- 2 0.0.0.0 1.0.1.2 1 13`
9. `s -t 1.654546735 -Hs 4 -Hd -2 -Ni 4 -Nx 70.00 -Ny 200.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 249e -Md 2 -Ms 0 -Mt 0`

Os itens 7 (sete) a 9 (nove) descrevem o início da troca de dados entre o host WD(0) e o host móvel.

10. `r 4.220853 0 3 tcp 1040 2 0.0.0.0 1.0.1.2 219 450`
11. `r 4.224517 3 4 tcp 1040 2 0.0.0.0 1.0.1.2 219 450`
12. `s -t 4.233651595 -Hs 4 -Hd -2 -Ni 4 -Nx 70.00 -Ny 200.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 4c4 -Md 2 -Ms 0 -Mt 0`
13. `s -t 4.233965802 -Hs 6 -Hd 4194304 -Ni 6 -Nx 9.68 -Ny 214.83 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 13a -Md 0 -Ms 2 -Mt 800 -Is 4194305.2 -Id 0.0 -It ack -Il 112 -If 2 -Ii 437 -Iv 32`

Os itens 10 (dez) a 13(treze) descrevem a transferência de dados para o host móvel já em movimento, podem ser percebido através dos campos que

indicam as coordenadas do host, no item 4 ele estava na posição (5.00, 215.00, 0.0) e agora está na posição (9.68, 214.83, 0.0).

14. `r 28.399947 0 3 tcp 1040 ----- 2 0.0.0.0 1.0.1.2 1961 4024`
15. `r 28.403611 3 4 tcp 1040 ----- 2 0.0.0.0 1.0.1.2 1961 4024`
16. `r 28.407307 4 3 tcp 1060 ----- 2 0.0.0.1 2.0.0.1 1961 4024`
17. `+ 28.407307 3 5 tcp 1060 ----- 2 0.0.0.1 2.0.0.1 1961 4024`
18. `s -t 28.408724249 -Hs 5 -Hd -2 -Ni 5 -Nx 400.00 -Ny 200.00 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw --- -Ma 249e -Md 2 -Ms 1 -Mt 0`

Por fim, os itens 14 (quatorze) a 18 (dezoito) descrevem o envio de dados para o *host* móvel em uma rede externa (*foreign link*). O pacote é enviado pelo host WD(0) e caminha em direção ao *host* WD(3) que é enviado para o HA que encapsula o pacote e reenvia para o FA através do *host* WD(3). Este encapsulamento pode ser notado através do campo que indica o tamanho do pacote que foi enviado. Nos itens 14 (quatorze) e 15 (quinze) o pacote está com 1040 bytes, posteriormente nos itens 16 (dezesseis) e 17 (dezessete) o pacote já é reenviado com um cabeçalho a mais, tendo assim um acréscimo de 20 bytes no tamanho total do pacote que agora está com 1060 bytes.

Capítulo 5

Conclusão e Propostas Futuras

5.1 Conclusão

Foi possível através deste projeto realizar um considerável estudo e por conseqüência um enorme aprendizado na utilização e no funcionamento do IP móvel, que é um protocolo amplamente empregado nos sistemas de comunicações móveis (sistemas celulares), como também permitir o uso de uma sofisticada ferramenta de simulação (NS) utilizada por várias organizações acadêmicas e comerciais durante anos e até os dias de hoje.

Todas as simulações propostas por este projeto foram realizadas de acordo com o planejado, seguindo rigorosamente as definições do IP móvel contidas na biblioteca do NS.

5.2 Propostas Futuras

Este trabalho poderá no futuro ser utilizado para iniciar possíveis projetos relacionados à utilização do NS em qualquer tipo de simulação de redes com fio, sem fio ou até mesmo redes via satélites (suportadas pelo NS).

Referências Bibliográficas

[Derfler (1993)], F. J. Jr; Fred L. "Como Funcionam as Redes". Editora Quart, 1993.

[Ferreira (1999)] A A; Robson G. "O Paradigma Computacional da Próxima Década". I Escola de Informática da SBC, Edição Norte, Pará, 1999.

[Tanembaum (1997)], Andrew S. "Redes de Computadores". Tradução da 3ª edição. Editora Campus. 1997.

[Lough (1997)], Daniel L; Blankenship, T. Keith; Krizman, Kevin J. "A Short Tutorial on Wireless and IEEE 802.11". Virginia Polytechnic Institute and State University. Departament of Electrical and Computer Engineering. 1997

[NS-2] "THE NETWORK SIMULATOR" [online]. Avaliable from http://www.isi.edu/nsnam. Cited 20 April 2003.

[KESHAV], S., "REAL 5.0 overview" [online]. Avaliable from http://www.cs.cornell.edu/skeshav/real/overview.html Cited 20 April 2003.

[Perkins] Charles E. "Mobile Networking Through Mobile IP". Sun Microsystems[online]. Available from http://www.computer.org /internet/v2n1/perkins.htm Cited 20 April 2003.

[Perkins (1996-I)] Charles E. “IP Encapsulation Within IP,” IETF RFC 2003, May 1996.

[Perkins (1996-II)] Charles E. “Minimal Encapsulation Within I,” IETF RFC 2004, May 1996.

[Sun (2001)] Microsystems Inc. “Mobile IP Administration Guide”. Part Number 806-7600-10, April 2001.

[Solomon (1998)] J. “Mobile-IP,” Prentice Hall, New Jersey, 1998.

[Deering (1991)] S.E. “ICMP Router Discovery Messages,” IETF RFC 1256, Sept. 1991.

[Rivest (1992)] R.L. "The MD5 Message-Digest Algorithm," IETF RFC 1321, Apr. 1992

[SAMAN] (Simulation Augmented by Measurement and Analysis for Networks) [online]. Available from http://www.isi.edu/saman/index.html.

[CONSER] (Collaborative Simulation for Education and Research) [online]. Available from http://www.isi.edu/conser/index.html Cited 20 April 2003.

[ICIR] (The International Computer Science Institute Center for Internet Research) [online]. Available from http://www.icir.org/.

[OTCL] (MIT Object Tool Command Language) [online]. Available from http://otcl-tclcl.sourceforge.net/otcl/.

[Brown], Brian. “Introduction to awk” [online]. Available from http://allman.rhon.itam.mx/dcomp/awk.html.

[Ferguson (1998)], Paul; HUSTON, Geoff. “Quality of service”. 1ª. edição. John Wiley & Sons, 1998.

[Alencar (1998)] Marcelo Sampaio de Alencar. “Telefonia Digital”. São Paulo, Érica, 1998.

[Marcos], Marcos Portnoi. **“**Network Simulator – Visão Geral da Ferramenta de Simulação de Redes**”** Available from http://locksmith.orcishweb.com/articles/networksimulator-sepa.pdf.

Anexos

Anexo A

Arquivo projeto.tcl

```
#////////////////////////////////////////////////////////////////////////////////

#////////////////////////////////////////////////////////////////////////////////

#                SIMULATION OF MOBILE IP THROUGH NETWORK SIMULATOR (NS2):
#                           A PROPOSAL OF WIRELESS NETWORK


#                                  Monografy's Project
#
#    Pupil : Guilherme Stella Ravagnani 20001217
#    Teacher who orientate : Teach. Ricardo Martins de Abreu Silva


#                                        University Federal of Lavras - MG Brazil
#                                        Department of Science of Computer - DSC

#    Copyrigthy(c) 2003
#    ***************************************************************************
#    *                                                                         *
#    * This source is free; redistribution and use in source and binary forms, *
#    * with or without modification, are permitted provided that the following *
#    * conditions describe at the terms of the GNU General Public License as   *
#    * published by the Free Software Foundation; either version 2 of the      *
#    * License, or (at your option) any later version.                         *
#    *                                                                         *
#    ***************************************************************************

#////////////////////////////////////////////////////////////////////////////////

#////////////////////////////////////////////////////////////////////////////////

# ======================================================================
# Define options
# ======================================================================

set opt(chan)   Channel/WirelessChannel     ;# channel type
set opt(prop)   Propagation/TwoRayGround    ;# radio-propagation model
set opt(netif)  Phy/WirelessPhy             ;# network interface type
set opt(mac)    Mac/802_11                  ;# MAC type
set opt(ifq)    Queue/DropTail/PriQueue     ;# interface queue type
set opt(ll)     LL                          ;# link layer type
set opt(ant)    Antenna/OmniAntenna         ;# antenna model
set opt(ifqlen)         50                  ;# max packet in ifq
set opt(adhocRouting)   DSDV                ;# routing protocol

set opt(x)      500                         ;# x coordinate of topology
set opt(y)      300                         ;# y coordinate of topology
set opt(stop)   30                          ;# time to stop simulation
set opt(seed)   0.0

set opt(ftp-start)      0.5                 ;# time to start FTP aplication

# ======================================================================

set num_wired_nodes     4                   ;# number of wired nodes
set num_bs_nodes        2                   ;# number of base station
set opt(nn)             1                   ;# number of mobilenodes


# ======================================================================
# Main Program
# ======================================================================

# create simulator instance
set ns   [new Simulator]

# set up for hierarchical routing
$ns node-config -addressType hierarchical

AddrParams set domain_num_ 3             ;# number of domains
```

```
72  lappend cluster_num 4 1 1                ;# number of clusters in each domain
73  AddrParams set cluster_num_ $cluster_num
74  lappend eilastlevel 1 1 1 1 2 1          ;# number of nodes in each cluster
75  AddrParams set nodes_num_ $eilastlevel   ;# of each domain
76
77  $ns use-newtrace
78  set tracefd  [open projeto-out.tr w]
79  set namtrace [open projeto-out.nam w]
80  $ns trace-all $tracefd                   ;# All traces are written in the tracefd
81  ;# all wireless nam traces are written into the namtrace
82  $ns namtrace-all-wireless $namtrace $opt(x) $opt(y)
83
84
85  # Create topography object
86  set topo   [new Topography]
87
88  # define the topology's boundaries
89  $topo load_flatgrid $opt(x) $opt(y)
90
91  # create God
92  # God needs to know the number of all wireless interfaces
93  create-god [expr $opt(nn) + $num_bs_nodes]
94
95  #create wired nodes
96  set temp {0.0.0 0.1.0 0.2.0 0.3.0}          ;# hierarchical addresses
97
98  for {set i 0} {$i < ($num_wired_nodes)} {incr i} {
99      set WD($i) [$ns node [lindex $temp $i]]
100 }
101
102 # Create channel
103 set chan_ [new $opt(chan)]
104
105 # Configure for ForeignAgent and HomeAgent nodes
106 $ns node-config -mobileIP ON \
107                 -adhocRouting $opt(adhocRouting) \
108                 -llType $opt(ll) \
109                 -macType $opt(mac) \
110                 -ifqType $opt(ifq) \
111                 -ifqLen $opt(ifqlen) \
112                 -antType $opt(ant) \
113                 -propType $opt(prop) \
114                 -phyType $opt(netif) \
115                 -channel $chan_ \
116                 -topoInstance $topo \
117                 -wiredRouting ON \
118                 -agentTrace ON \
119                 -routerTrace OFF \
120                 -macTrace ON
121
122 # Create HA and FA
123 set HA [$ns node 1.0.0]
124 set FA [$ns node 2.0.0]
125 $HA random-motion 0
126 $FA random-motion 0
127
128 # Position (fixed) for base-station nodes (HA & FA) and wired node
129
130 $HA set X_ 70.000000000000
131 $HA set Y_ 200.000000000000
132 $HA set Z_ 0.000000000000
133
134 $FA set X_ 400.000000000000
135 $FA set Y_ 200.000000000000
136 $FA set Z_ 0.000000000000
137
138 $WD(3) set X_ 250.000000000000
139 $WD(3) set Y_ 100.000000000000
140 $WD(3) set Z_ 0.000000000000
141
142 # create a mobilenode that would be moving between HA and FA.
```

```
143 # note address of MH indicates its in the same domain as HA.
144 $ns node-config -wiredRouting OFF
145
146 set MH [$ns node 1.0.1]
147 set HAaddress [AddrParams addr2id [$HA node-addr]]
148 [$MH set regagent_] set home_agent_ $HAaddress
149
150 # movement of the MH
151 $MH set X_ 5.000000000000
152 $MH set Y_ 215.000000000000
153 $MH set Z_ 0.000000000000
154
155 # MH starts to move towards FA
156 $ns at 4.000000000000 "$MH setdest 420.000000000000 200.000000000000 20.000000000000"
157
158 #give label, shape and color for nodes
159 $HA shape "hexagon"
160 $FA shape "hexagon"
161 $HA color "red"
162 $FA color "red"
163
164 $WD(0) shape "square"
165 $WD(0) color "blue"
166 $WD(1) shape "square"
167 $WD(1) color "blue"
168 $WD(2) shape "square"
169 $WD(2) color "blue"
170 $WD(3) shape "square"
171 $WD(3) color "blue"
172
173 $HA    label HA
174 $FA    label FA
175 $WD(0) label WD0
176 $WD(1) label WD1
177 $WD(2) label WD2
178 $WD(3) label WD3
179 $MH    label MH
180
181 # create links between wired and BaseStation nodes
182
183 $ns duplex-link $WD(0) $WD(3) 5Mb 2ms DropTail
184 $ns duplex-link $WD(1) $WD(3) 5Mb 2ms DropTail
185 $ns duplex-link $WD(2) $WD(3) 5Mb 2ms DropTail
186 $ns duplex-link $WD(3) $HA    5Mb 2ms DropTail
187 $ns duplex-link $WD(3) $FA    5Mb 2ms DropTail
188
189 $ns duplex-link-op $WD(1) $WD(3) orient up
190 $ns duplex-link-op $WD(2) $WD(3) orient left-up
191 $ns duplex-link-op $WD(0) $WD(3) orient right-up
192 $ns duplex-link-op $HA    $WD(3) orient right-down
193 $ns duplex-link-op $FA    $WD(3) orient left-down
194
195 # define color index
196 $ns color 0 tan
197
198 # setup TCP connections between a wired node and the MobileHost
199 set tcp [new Agent/TCP]             ;# create sender agent
200 $tcp set class_ 2
201 set sink [new Agent/TCPSink]        ;# create receiver agent
202 $ns attach-agent $WD(0) $tcp        ;# put sender on node $WD(0)
203 $ns attach-agent $MH $sink          ;# put receiver on node $MH
204 $ns connect $tcp $sink              ;# establish TCP connection
205 set ftp [new Application/FTP]       ;# create an FTP source "application"
206 $ftp attach-agent $tcp              ;# associate FTP with the TCP sender
207 $ns at $opt(ftp-start) "$ftp start" ;#arrange for FTP to start at time $opt(ftp-start)
208
209
210 # Define initial node position in nam
211 # 12 defines the node size in nam
212 $ns initial_node_pos $MH 12
213
```

```
214 # Tell all nodes when the siulation ends
215 $ns at $opt(stop).0 "$MH reset"
216
217 $ns at $opt(stop).0 "$HA reset";
218 $ns at $opt(stop).0 "$FA reset";
219
220 $ns at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns halt"
221 $ns at $opt(stop).0001 "stop"
222
223 proc stop {} {
224     global ns_ tracefd namtrace
225     close $tracefd
226     close $namtrace
227 }
228
229 ;# inserts an annotation in nam
230 $ns at 0.0 "$ns trace-annotate \"simulation start\""
231 $ns at 0.5  "$ns trace-annotate \"start ftp between WD(0) and MH\""
232 $ns at 7.0  "$ns trace-annotate \"mobile node goes to FA's dominion\""
233 $ns at 30.0 "$ns trace-annotate \"simulation end\""
234
235 # headers for tracefile
236 puts $tracefd "M 0.0 title Mobile IP's Simulation"
237 puts $tracefd "M 0.0 mobilenode $opt(nn) x $opt(x) y $opt(y) rp \ $opt(adhocRouting)"
238 puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"
239
240 puts "Starting Simulation..."
241 $ns run                 ;# starts the simulator
```