

Tutorial do Simulador de Redes GNS3

Marcelo Samsoniuk

Professor Eduardo Parente Ribeiro
Rede de Comunicação de Dados - TE815
Programa de Pós-graduação em Engenharia Elétrica
Departamento de Engenharia Elétrica
UFPR – 2010

Introdução

O GNS3 é um front-end gráfico que permite gerenciar a simulação de redes complexas através de simuladores de processadores MIPS, PowerPC e x86. Pode ser instalado no Windows, Linux ou OSX, à partir do endereço:

<http://www.gns3.net/download>

Além do front-end gráfico, é necessário instalar os simuladores de processadores, sendo os processadores MIPS, PowerPC e x86 atualmente suportados.

O Dynamips simula os processadores MIPS e PowerPC, que por sua vez executam imagens do IOS destinadas aos roteadores Cisco da série 1700, 2600, 3600, 3700 e 7200. Já o QEMU simula o processador x86 e executa imagens do IOS destinadas ao firewall Cisco PIX e imagens do JunOS destinadas aos roteadores Juniper.

O site abaixo possui uma relação dos processadores utilizados nos diversos modelos de roteadores Cisco já produzidos, de modo que apenas os modelos com processadores MIPS ou PowerPC podem ser utilizados no GNS3:

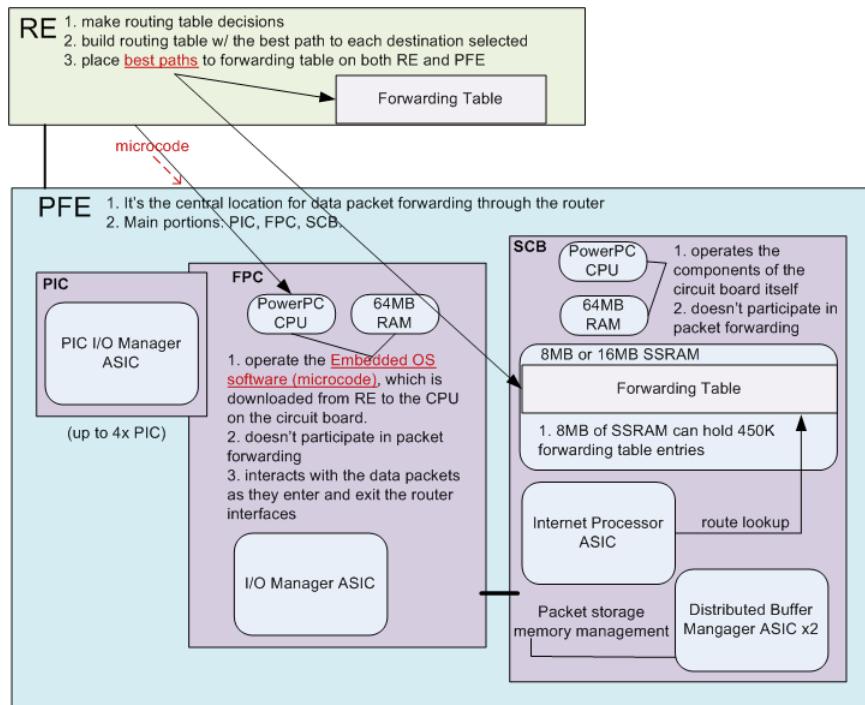
<http://www.linux-mips.org/wiki/Cisco>

O porte dos roteadores mais ou menos compatíveis com os modelos ilustrados na figura abaixo:



Além dos roteadores Cisco com processador MIPS e PowerPC, existe ainda a possibilidade de simular o firewall Cisco PIX, utilizando o simulador x86 QEMU.

No caso dos roteadores Juniper, a arquitetura é mais padronizada entre os diferentes modelos, utilizando processadores x86 para as REs (routing engines) e processadores PowerPC para as PFEs (packet routing engines), sendo normalmente uma relação de uma RE para diversas PFEs em um chassis grande. A figura abaixo ilustra como as REs e PFEs se organizam em um típico chassis da Juniper:



Vale a pena observar que, enquanto os roteadores da Cisco utilizam o conceito de Network Processor (um processador projetado para centralizar as operações de comunicação), os roteadores da Juniper dividem o processamento entre sub-sistemas de controle geral e sub-sistemas especializados para comunicação. Os sub-sistemas são todos independentes, de modo que uma falha em um sub-sistema não afeta os outros (como ocorre no conceito centralizado da Cisco).

Por outro lado, o GNS3 até o momento simula apenas as REs da Juniper, o que limita bastante a funcionalidade, na medida que não é possível simular o tráfego pesado das PFEs. Por este motivo, vamos nos concentrar neste tutorial apenas nos equipamentos da Cisco.

Instalação

O processo de instalação varia de sistema operacional para sistema operacional, mas binários disponíveis no site oficial para Windows e OSX facilitam a instalação nestes dois sistemas. No caso do Linux, existe a opção de compilar o fonte disponível no site oficial ou então utilizar um pacote pré-compilado de acordo com a distribuição utilizada.

No caso do OSX, por exemplo, o pacote pré-compilado já inclui as bibliotecas do Python, QT, PyQt e SIP, bem como os binários do GNS3 e do Dynamips. Já no pacote pré-compilado para o Windows, além destas bibliotecas e binários está incluso o QEMU, Putty e WinPCAP. Finalmente, no Linux os requisitos irão variar bastante, sendo similar ao OSX nas distribuições menores chegando a não precisar de absolutamente nada adicional no caso das distribuições mais completas.

Independente do processo, o passo mais importante é verificar se o Dynamips está funcionando corretamente primeiro e então partir para a utilização do GNS3. Nossso tutorial não inclui a simulação de equipamentos x86, como o PIX, portanto não vamos nos preocupar com o QEMU.

Obviamente é necessário possuir uma ou mais imagens do IOS da Cisco!

Estas imagens não são normalmente distribuídas e tratam-se de software proprietário. Por outro lado, as imagens normalmente estão disponíveis nos próprios roteadores, sendo relativamente fácil fazer uma cópia de um roteador para outro, como ilustrado no próprio site da Cisco:

<http://www.cisco.com/application/pdf/paws/15092/copyimage.pdf>

Assumindo que temos acesso a um roteador e que podemos obter a imagem legalmente, podemos testar a imagem no Dynamips diretamente:

Pelo nome do arquivo, inferiu-se que a imagem pertencia a um Cisco 2600, porém aparentemente após a descompressão da imagem, não houve sucesso rodando a simulação. O Dynamips simplesmente fez um shutdown de forma silenciosa, sem muitas explicações ou indicações de erro.

No fundo, isto ocorre em função de uma particularidade: o modelo 2600 é baseado no PowerPC 860 e um bug no código do simulador previne que ele consiga descomprimir com sucesso as imagens do IOS para esse processador, o que não é necessário quando temos uma imagem do IOS para os modelos baseados nos processadores MIPS.

Assim, descompactando a imagem:

```
# unzip -p c2600-is-mz.122-2.T.bin > c2600-is-mz.122-2.T.bin.raw
warning [c2600-is-mz.122-2.T.bin]: 16624 extra bytes at beginning or within zipfile
(attempting to process anyway)
```

E rodando a simulação novamente, obtemos uma execução com sucesso:

```
# dynamips-0.2.8-RC2-OSX-Leopard.intel.bin -P 2600 ./c2600-is-mz.122-2.T.bin.raw
Cisco Router Simulation Platform (version 0.2.8-RC2-x86)
Copyright (c) 2005-2007 Christophe Fillot.
Build date: Nov 11 2007 11:11:35

IOS image file: ./c2600-is-mz.122-2.T.bin.raw

ILT: loaded table "mips64j" from cache.
ILT: loaded table "mips64e" from cache.
ILT: loaded table "ppc32j" from cache.
ILT: loaded table "ppc32e" from cache.
CPU0: carved JIT exec zone of 64 Mb into 2048 pages of 32 Kb.
NVRAM is empty, setting config register to 0x2142
C2600 instance 'default' (id 0):
  VM Status : 0
  RAM size : 64 Mb
  NVRAM size : 128 Kb
  IOS image : ./c2600-is-mz.122-2.T.bin.raw

Loading BAT registers
Loading ELF file './c2600-is-mz.122-2.T.bin.raw'...
ELF entry point: 0x80008000

C2600 'default': starting simulation (CPU0 IA=0xffff00100), JIT enabled.
ROMMON emulation microcode.

Launching IOS image at 0x80008000...

Smart Init is disabled. IOMEM set to: 15

Using iomem percentage: 15

      Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

  cisco Systems, Inc.
  170 West Tasman Drive
  San Jose, California 95134-1706

Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-IS-M), Version 12.2(2)T, RELEASE SOFTWARE (fc1)
TAC Support: http://www.cisco.com/cgi-bin/ibld/view.pl?i=support
Copyright (c) 1986-2001 by cisco Systems, Inc.
Compiled Sat 02-Jun-01 21:34 by ccai
Image text-base: 0x80008088, data-base: 0x8118A888

cisco 2610 (MPC860) processor (revision 0x202) with 56320K/9216K bytes of memory.
Processor board ID 000000000000 (1880125456)
M860 processor: part number 0, mask 0
Bridging software.
X.25 software, Version 3.0.0.
1 Ethernet/IEEE 802.3 interface(s)
128K bytes of non-volatile configuration memory.
8192K bytes of processor board System flash (Read/Write)

--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]:
```

O que indica que o roteador bootou corretamente e entrou no dialogo de configuração.

Para finalizar a simulação no prompt de comando, basta digitar ^]q (control+colchetes+q).

Fazendo o mesmo teste com uma imagem MIPS compactada temos:

Restricted Rights Legend

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) of the Commercial Computer Software - Restricted Rights clause at FAR sec. 52.227-19 and subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS sec. 252.227-7013.

cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134-1706

```

Cisco Internetwork Operating System Software
IOS (tm) 3600 Software (C3640-IS-M), Version 12.2(2)T, RELEASE SOFTWARE (fc1)
TAC Support: http://www.cisco.com/cgi-bin/ibld/view.pl?i=support
Copyright (c) 1986-2001 by cisco Systems, Inc.
Compiled Sat 02-Jun-01 16:15 by ccrai
Image text-base: 0x600089A8, data-base: 0x612F8000

cisco 3640 (R4700) processor (revision 0xFF) with 126976K/4096K bytes of memory.
Processor board ID 00000000
R4700 CPU at 100Mhz, Implementation 33, Rev 1.2
Bridging software.
X.25 software, Version 3.0.0.
SuperLAT software (copyright 1990 by Meridian Technology Corp).
DRAM configuration is 64 bits wide with parity enabled.
125K bytes of non-volatile configuration memory.
8192K bytes of processor board System flash (Read/Write)

```

--- System Configuration Dialog ---

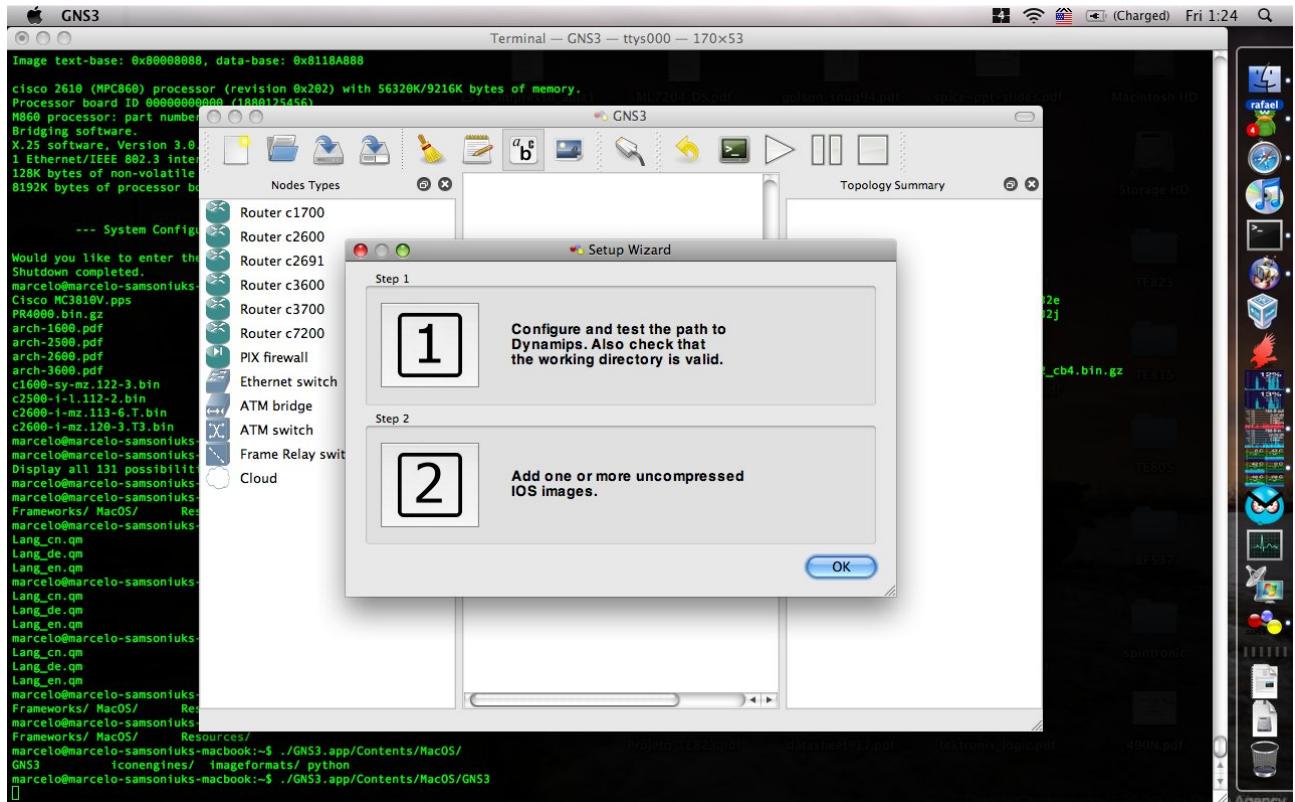
Would you like to enter the initial configuration dialog? [yes/no]:

Obtendo assim o mesmo resultado tanto para o modelo 2610 com PowerPC quanto 3640 com MIPS. Novamente, podemos finalizar a simulação com ^]q.

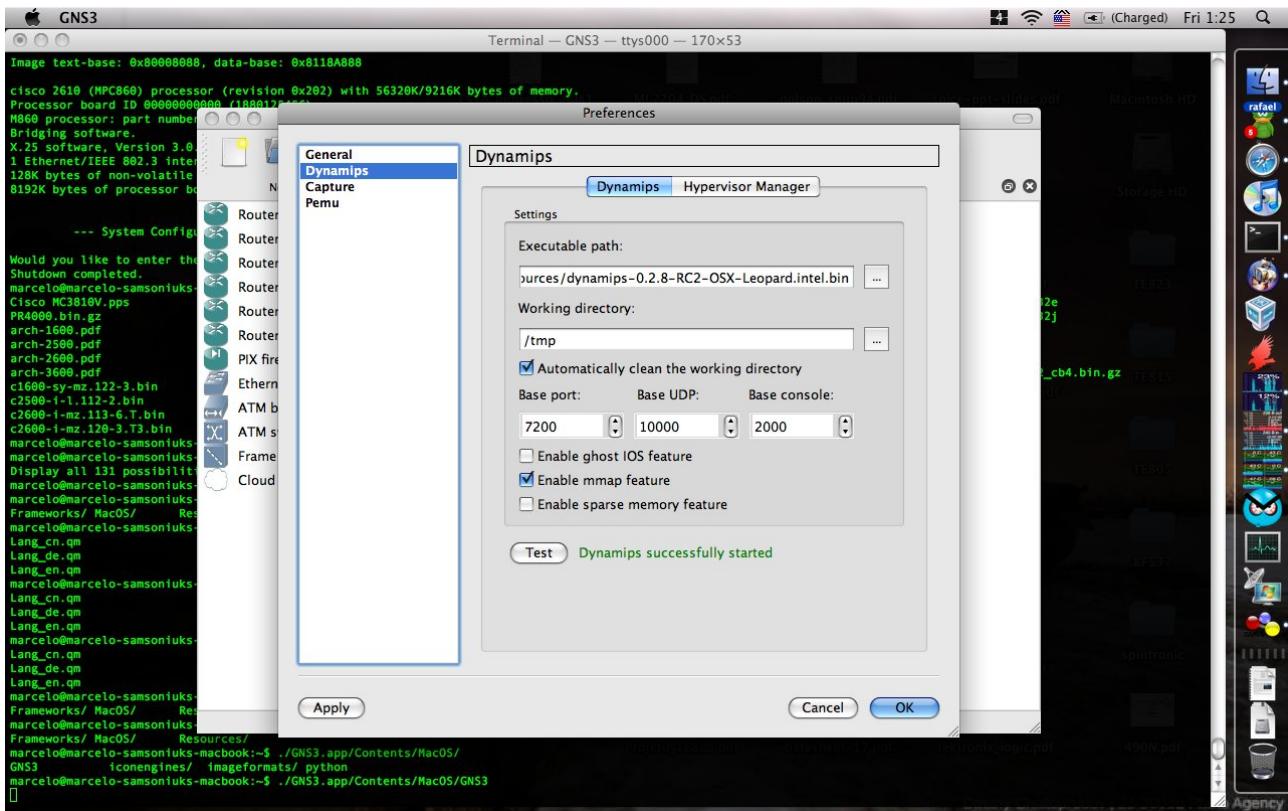
Estando devidamente testado o Dynamips e as imagens do IOS, podemos passar para a configuração do GNS3.

Configuração

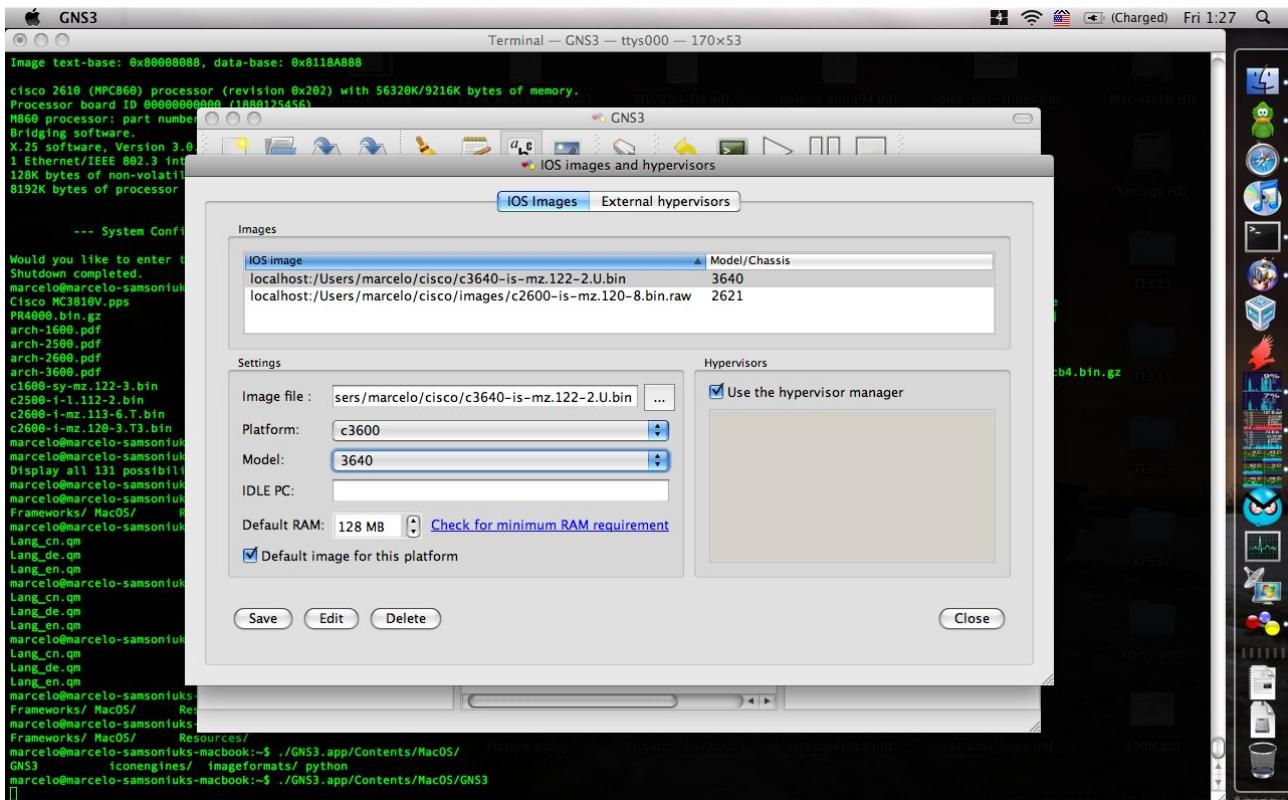
O GNS3 é o front-end gráfico que permite gerenciar as simulações do Dynamips, criando topologias mais complexas com vários tipos de roteadores. Quando o GNS3 é executado pela primeira vez, ele solicita uma configuração inicial com o caminho do simulador e o caminho das imagens do IOS para os simuladores serem executados:



Na configuração do Dynamips, basta configurar o caminho correto do simulador e testar:



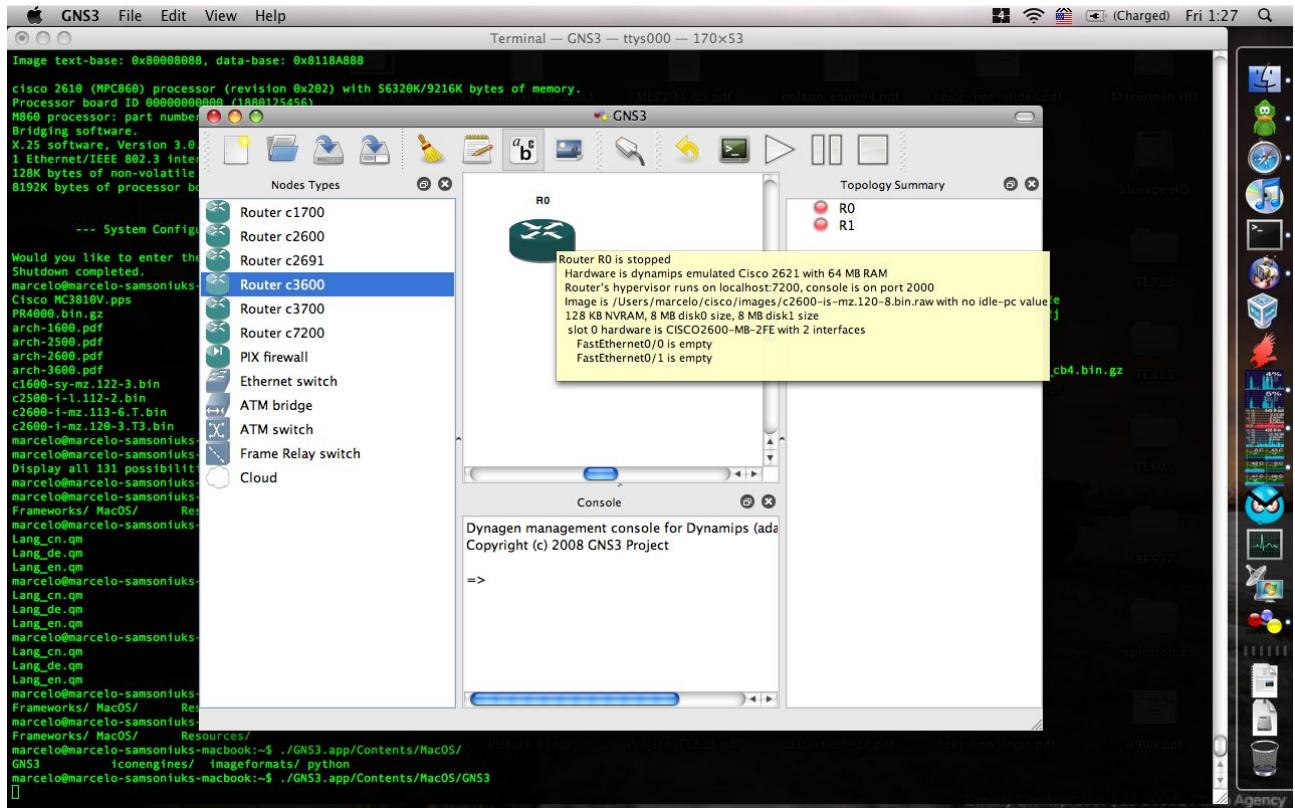
Na configuração das imagens, basta escolher as respectivas imagens para cada roteador:



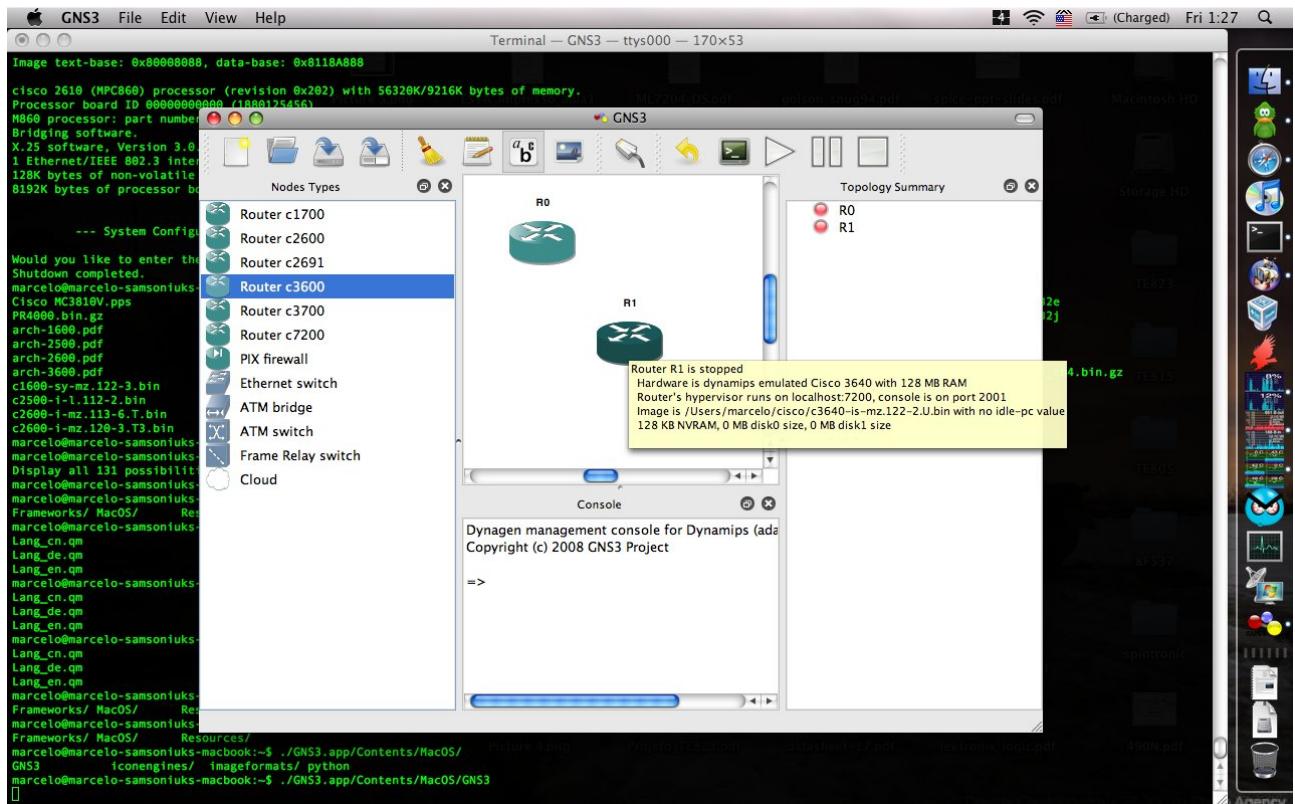
Neste caso, temos dois diferentes equipamentos, o modelos 2610 com processador PowerPC e o modelo 3640 com processador MIPS, cada um utilizando sua respectiva imagem do IOS.

Simulação

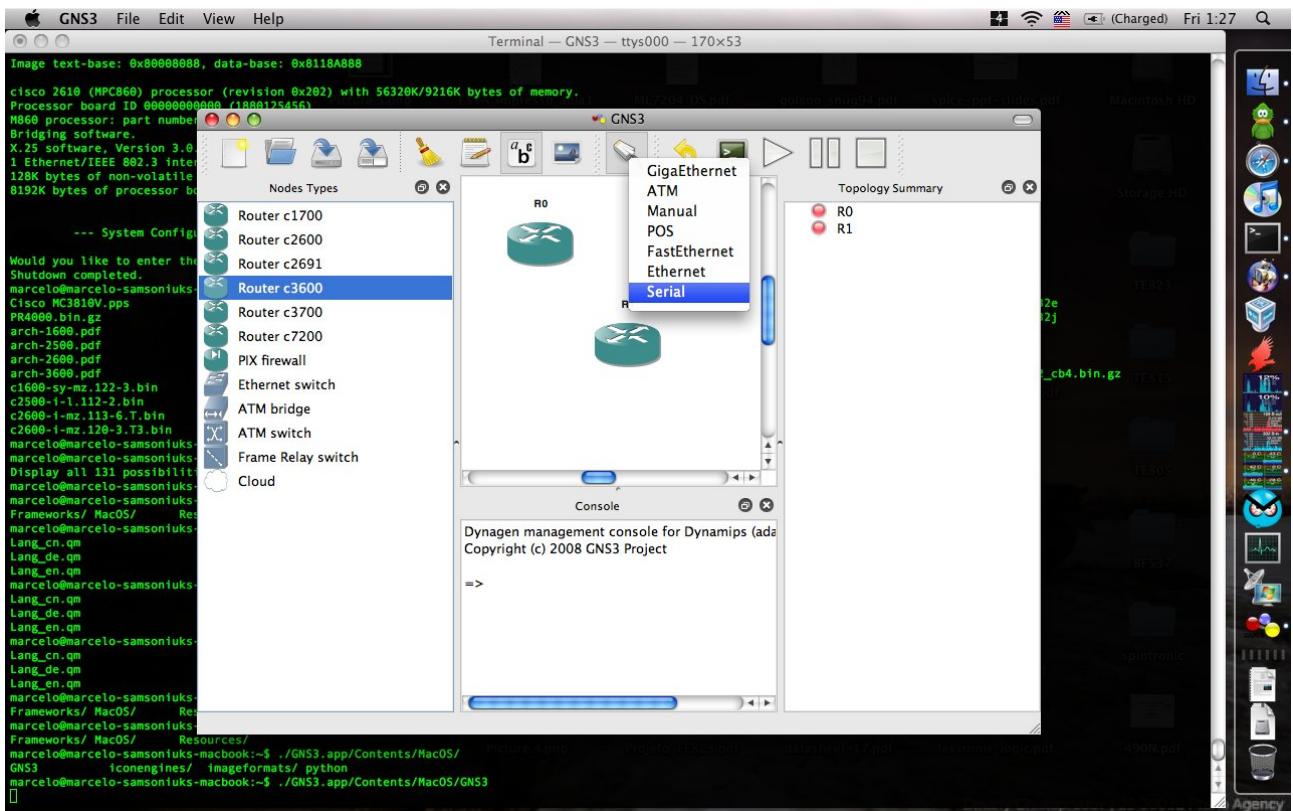
Para simular uma rede simples de dois equipamentos, adicionamos os equipamentos clicando e arrastando, começando com o modelo 2610:



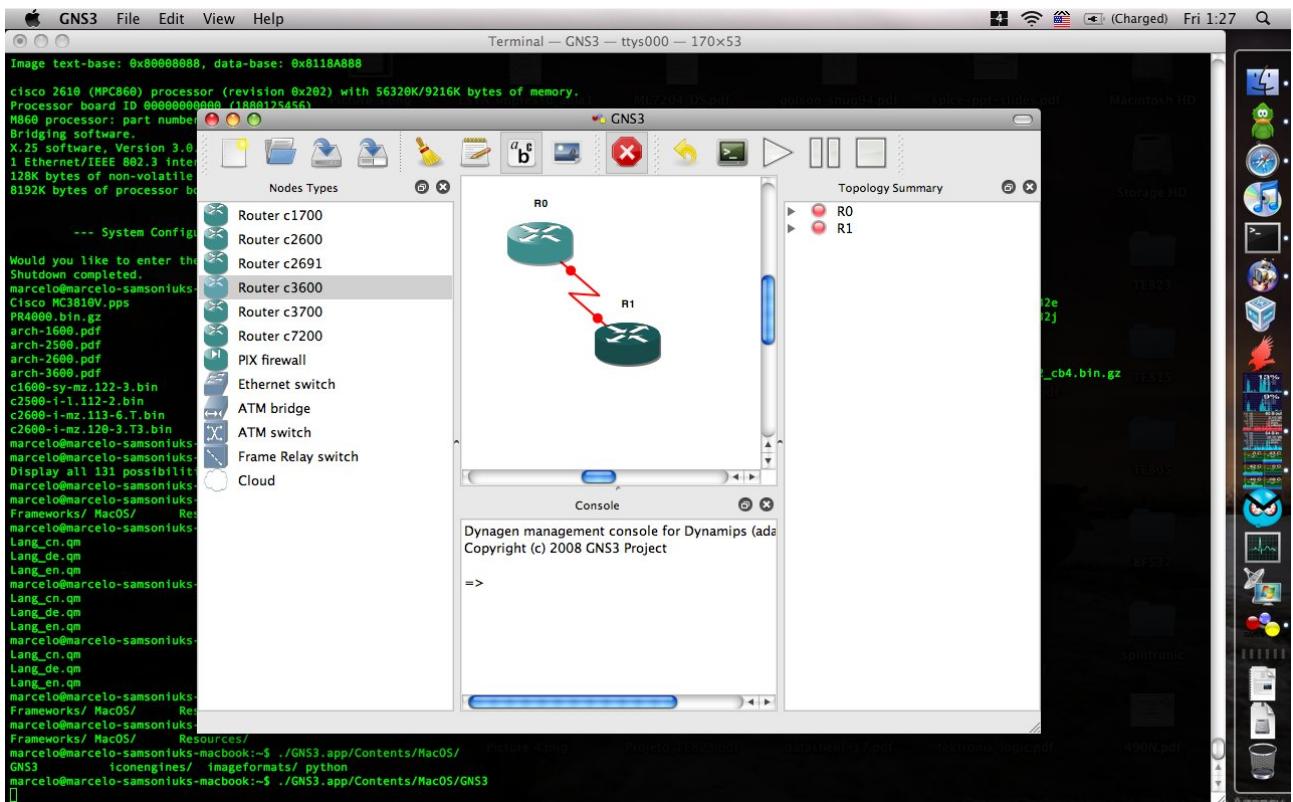
E adicionando o modelo 3640:



Utilizando a ferramenta de conexão é possível escolher uma série de tipos diferentes de conexões físicas para os equipamentos:

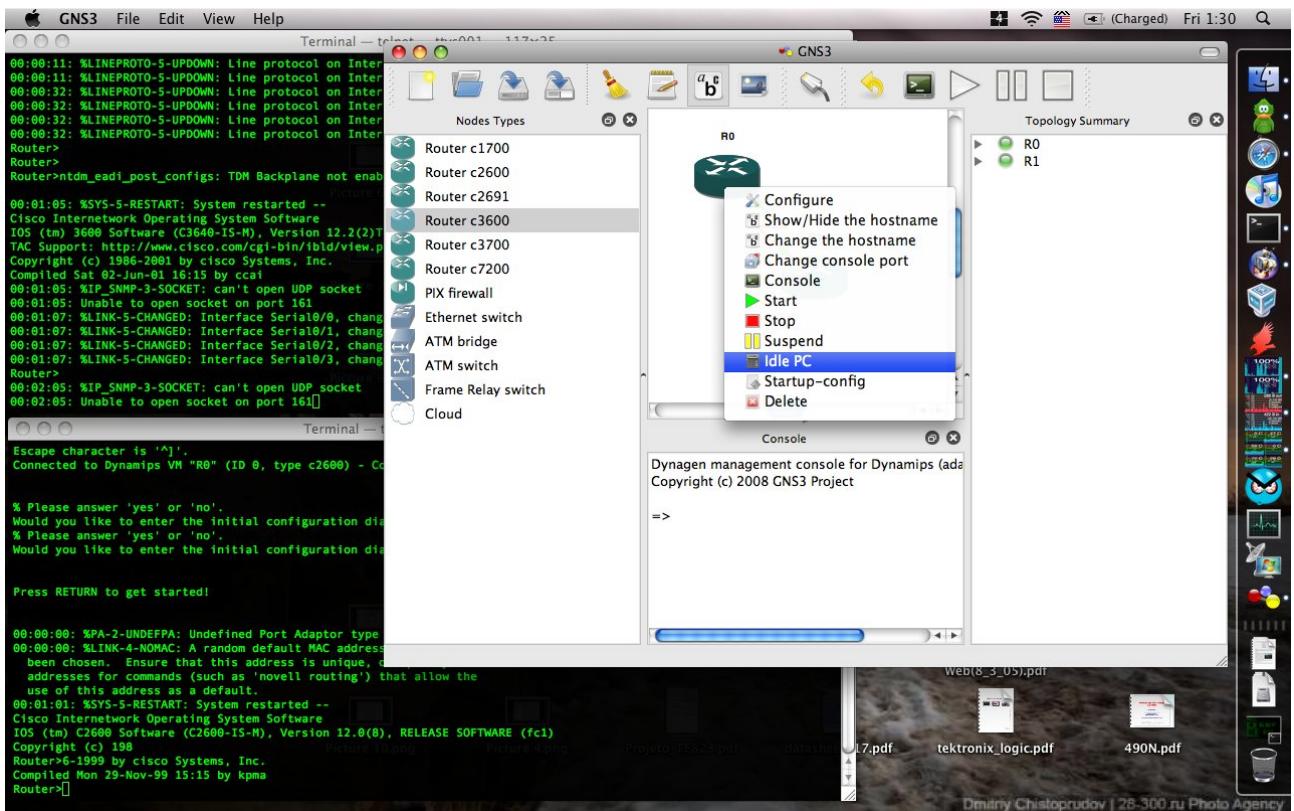


Neste caso, escolhemos uma conexão serial:

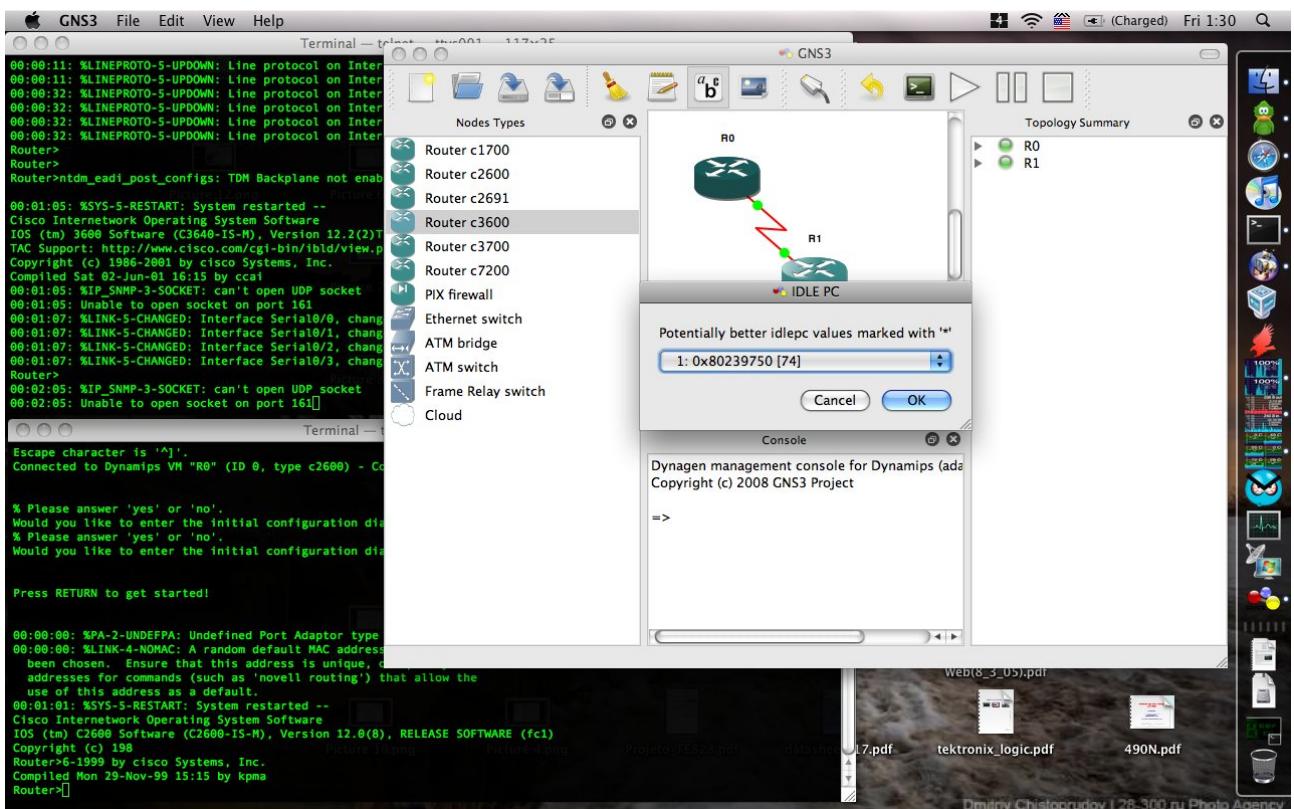


Clicando na ferramenta **start** e então em **console**, é possível iniciar a simulação e visualizar os dois roteadores bootando, porém ainda é necessário um ajuste adicional.

Para otimizar a performance da simulação e evitar o consumo excessivo do processador hospedeiro, é necessário selecionar em cada roteador a opção “Idle PC”, :



Neste caso, o Dynamips determinou que o endereço indicado abaixo é um dos diversos pontos de idle do roteador, ou seja, um ponto onde o simulador vai escalar para outra thread, maximizando a performance da simulação:



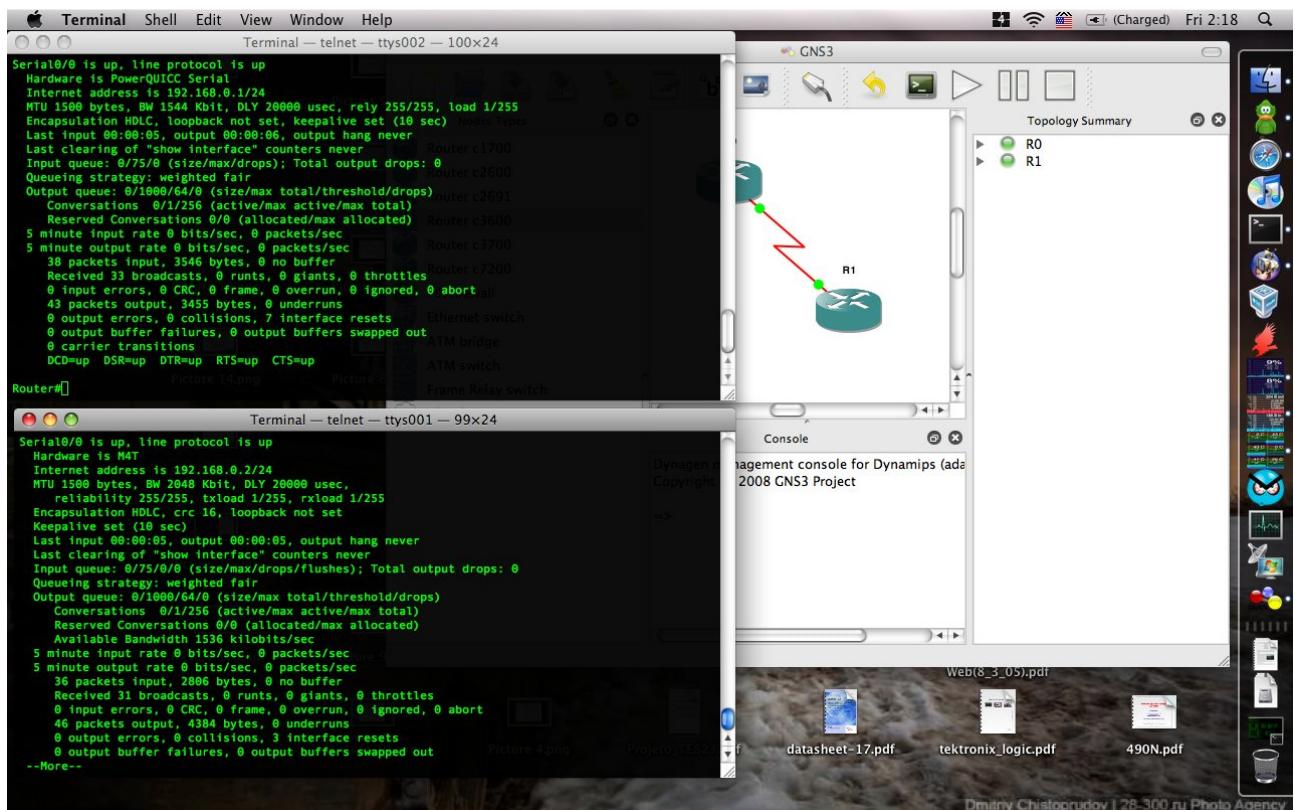
Neste momento os roteadores podem ser configurados, bastante digitar “no” para o menu de configuração inicial e teclar enter para cair no prompt de comando. Para configurar as interfaces, é possível digitar no primeiro roteador:

```
enable
configure terminal
interface serial 0/0
ip address 192.168.0.1 255.255.255.0
clock rate 64000
no shutdown
^Z
wr
show interface serial 0/0
```

E no segundo roteador:

```
enable
configure terminal
interface serial 0/0
ip address 192.168.0.2 255.255.255.0
no shutdown
^Z
wr
show interface serial 0/0
```

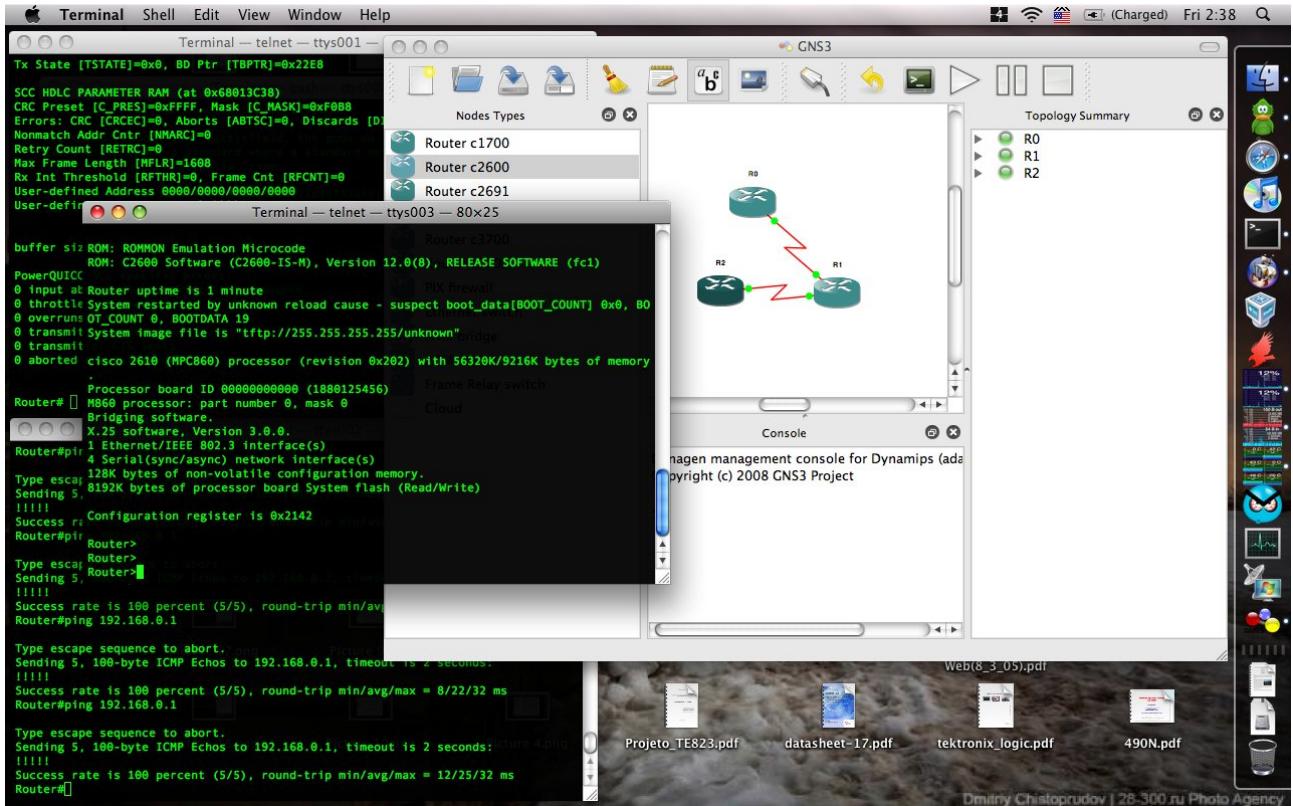
Obtendo o resultado abaixo:



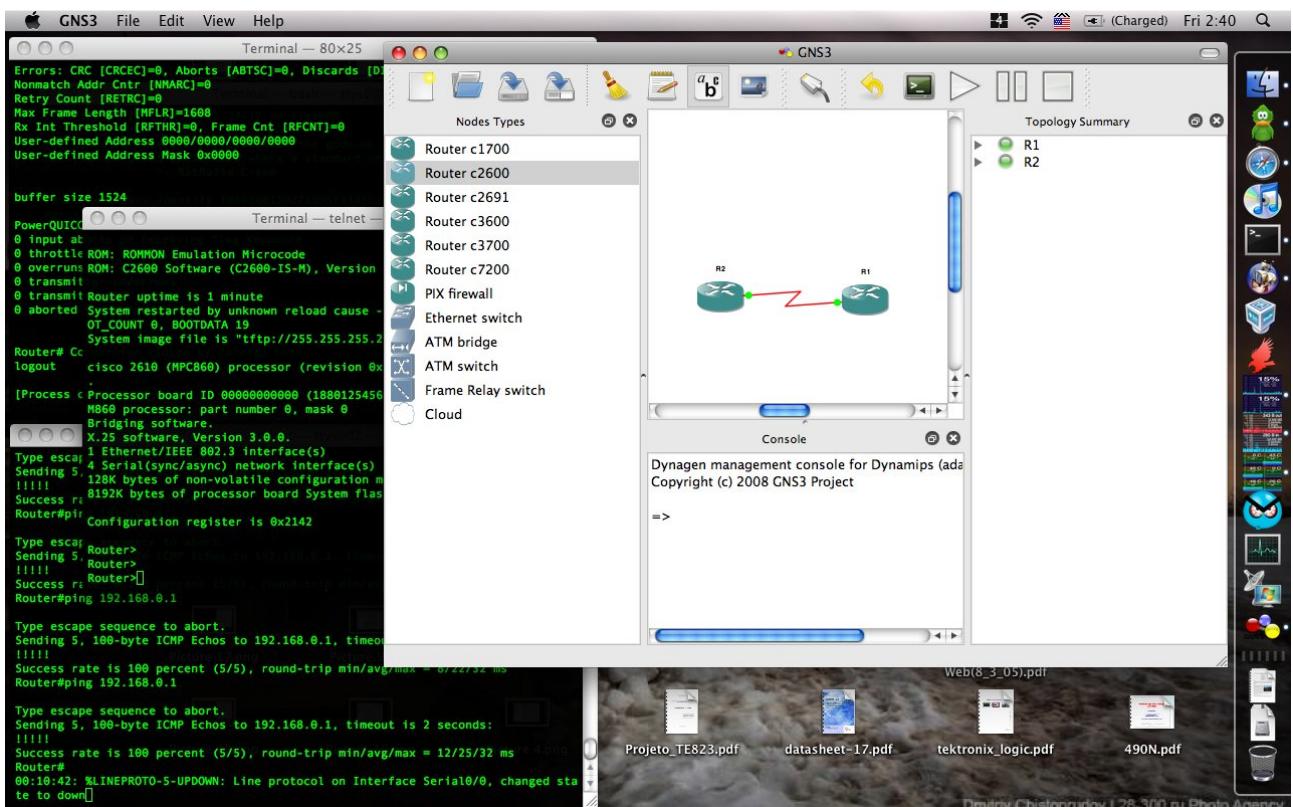
Quando observamos a serial up e o protocolo up, temos indicação que as interfaces seriais síncronas estão devidamente conectadas e operando com protocolo HDLC (default para interfaces seriais da Cisco) com um clock de 64kHz fornecido pelo 2610. Em condições reais ambos os roteadores estarão conectados à uma operadora de longa distância, de modo que o clock será fornecido pela operadora e será síncrono com a rede TDM da operadora. Como veremos adiante, acertar a velocidade das seriais para obter simulações perfeitas não é realmente tão simples!

Uma vez configurado, o roteador irá manter sua configuração no GNS3, podendo ser desligado e ligado quantas vezes for necessário.

Assim, é possível parar a simulação em qualquer instante e fazer alterações na topologia. Por outro lado, o GNS3 é flexível o suficiente para permitir a adição de roteadores com a simulação em execução:

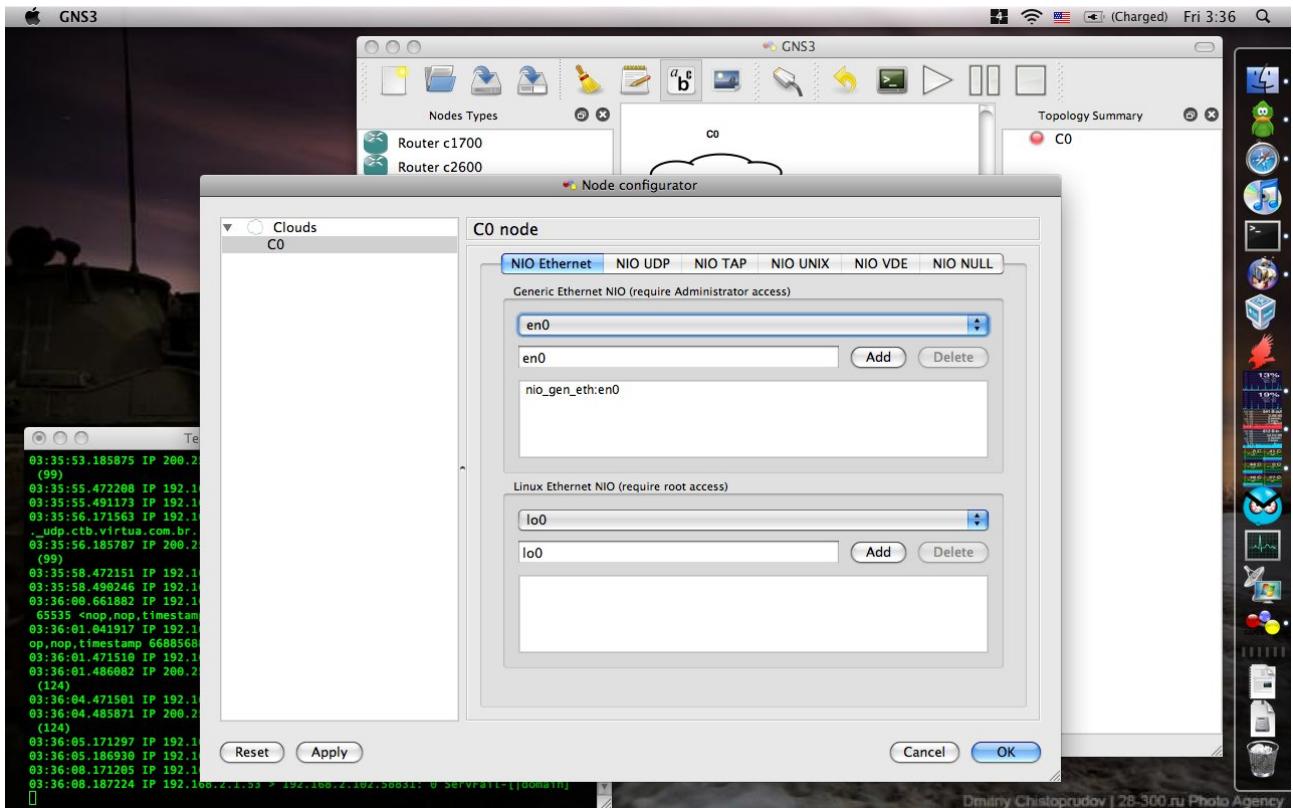


Da mesma forma, é possível parar um roteador em particular e removê-lo da topologia, sem no entanto afetar a execução dos outros:



Simulações Avançadas

Com o GNS3 é possível misturar sistemas simulados com sistemas reais, na medida que qualquer interface de rede presente na máquina hospedeira pode ser mapeada. Para isso basta adicionar uma nuvem e então escolher a interface que será utilizada, no caso a **en0**:



Adicionamos então um 2610 conectado à núvem e outro 2610 conectado via interface serial ao primeiro. Configurando os roteadores temos para o 2620 conectado à núvem:

```
enable
configure terminal
interface ethernet 0/0
ip address 192.168.2.10 255.255.255.0
no shutdown
^Z
configure terminal
interface serial 0/0
ip address 192.168.3.1 255.255.255.0
no shutdown
^Z
configure terminal
ip route 0.0.0.0 0.0.0.0 192.168.2.1
ip name-server 192.168.2.1
ip routing
^Z
wr
```

Neste caso, supomos que o computador hospedeiro possui conexão com a internet através do endereço 192.168.2.1. Assim seria possível acessar a internet e testar um ping:

```
Router#ping www.uol.com.br
Translating "www.uol.com.br"...domain server (192.168.2.1) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.221.2.45, timeout is 2 seconds:
!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 8/32/60 ms
```

Configurando então o segundo roteador:

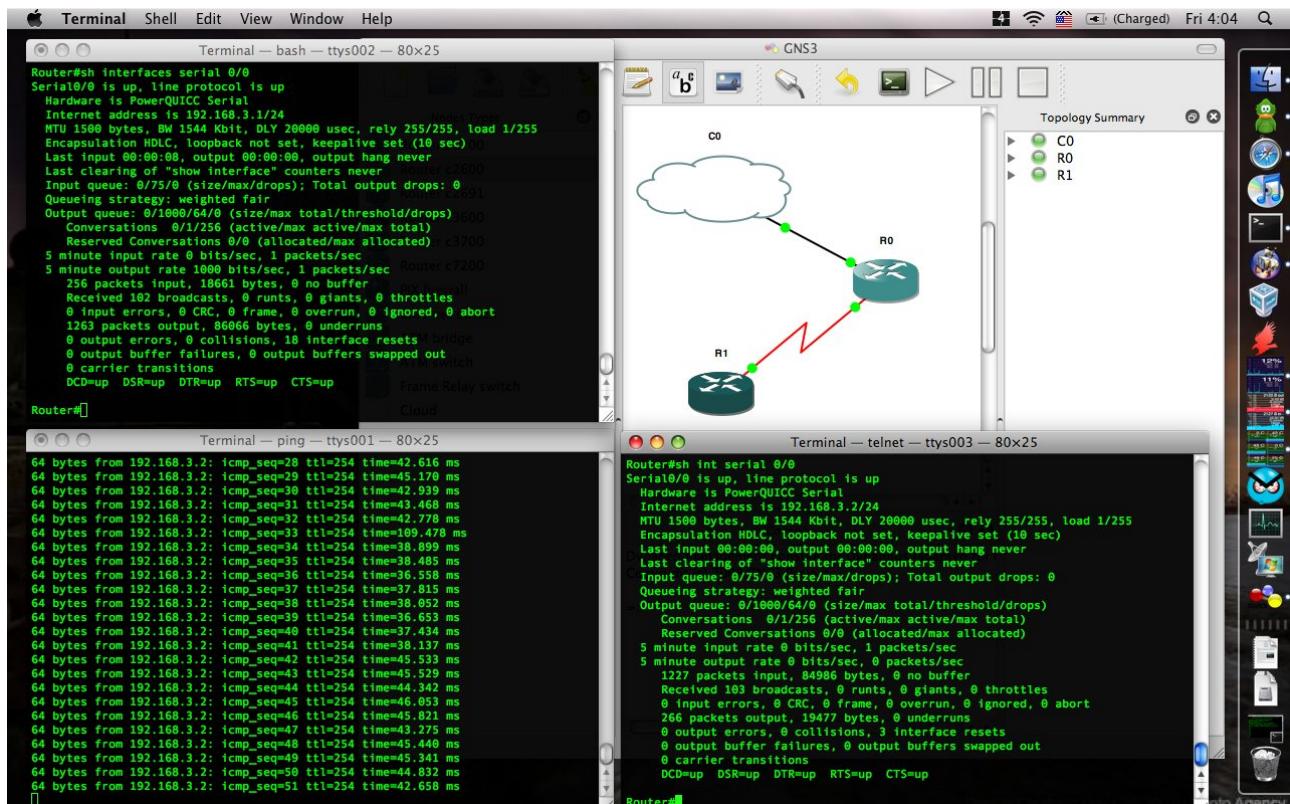
```
enable
configure terminal
interface serial 0/0
ip address 182.168.3.2 255.255.255.0
no shutdown
^ Z
configure terminal
ip route 0.0.0.0 0.0.0.0 192.168.3.1
ip routing
^ Z
wr
```

Temos conectividade entre os dois roteadores, porém não vamos muito longe, pois o roteador 192.168.2.1 não conhece as rotas para a rede 192.168.3.0/24. Assim vamos restringir nossa conectividade às redes locais.

Adicionando rotas no OSX, que é o computador hospedeiro:

```
route add 192.168.3.0/24 192.168.2.10
```

É possível interconectar os três equipamentos:

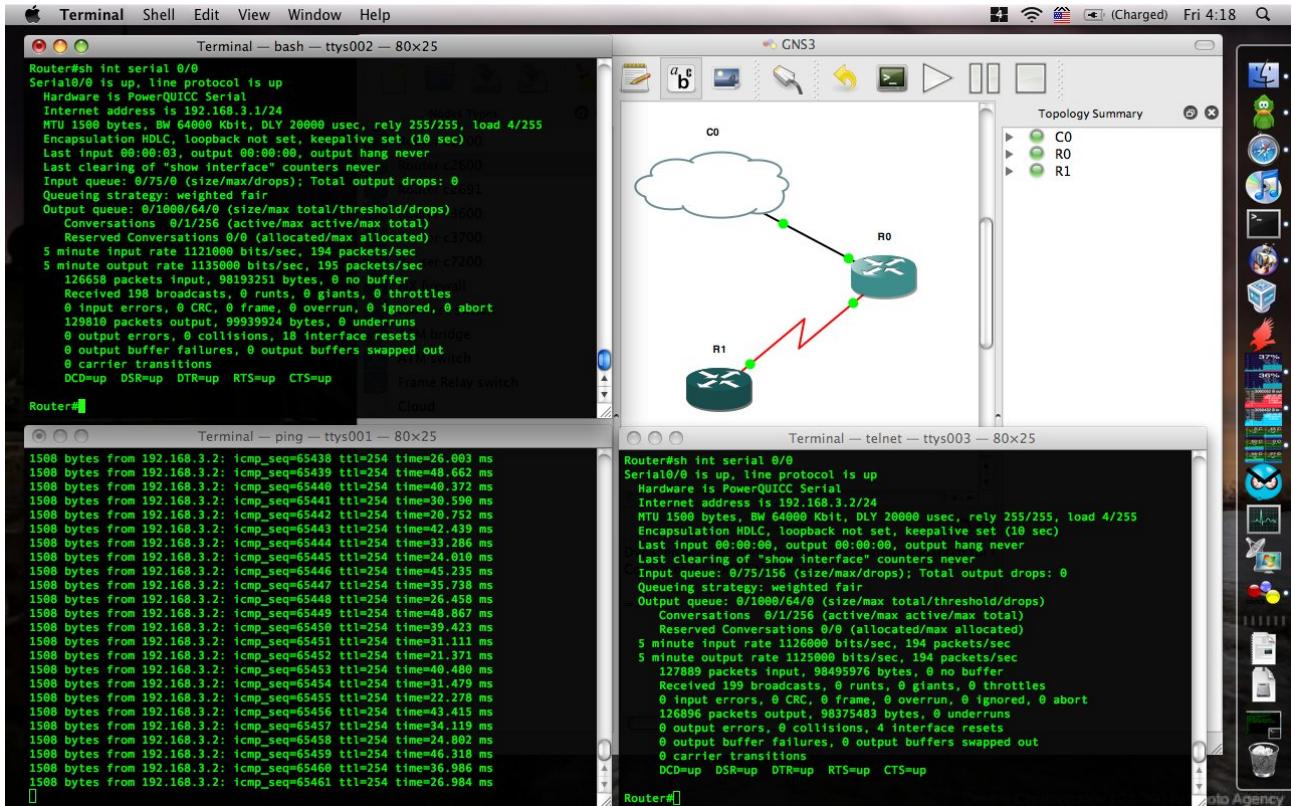


Assim, o OSX consegue se comunicar com o primeiro 2610 via interface ethernet e este roteador se comunica com o segundo 2610 via interface serial síncrona simulada. Com as devidas rotas, o OSX consegue falar com o segundo 2610, um equipamento totalmente virtual!

Com isto passa a ser possível utilizar o computador hospedeiro para gerar tráfego e mensurar resultados. Por exemplo, utilizando no OSX o comando:

```
ping -i 0.01 -s 1500 192.168.3.2
```

Estamos indicando para o sistema gerar um tráfego da ordem de 100 pacotes/segundo com protocolo ICMP e tamanho de 1500 bytes, o que totaliza aproximadamente 1.2Mbit/s, o que é confirmado nas estatísticas das interfaces dos roteadores:



Infelizmente, também é possível verificar nos exemplos de geração de tráfego que o simulador não atende corretamente as configurações de clock rate ou bandwidth no IOS simulado, pois era desejado limitar a performance da interface em apenas 64Kbit/s. Apesar das configurações, a velocidade das interfaces ultrapassou a taxa de 1Mbit/s.

Para obter precisão neste caso, a solução seria mapear as interfaces seriais de ambos os roteadores para uma nuvem apontando para um socket em loopback com controle de banda pelo firewall do kernel (no caso do OSX isto pode ser feito com o ipfw, enquanto que no Linux pode ser feito com o iptables). Esta solução será vista em detalhes adiante.

Uma observação muito importante para o OSX e Linux é que para o GNS3 ter acesso às interfaces físicas é necessário rodar o GNS3 como usuário **root**. No caso do Linux costuma existir restrição de acesso em relação ao X, o que no OSX é feito de forma transparente. No caso do Windows, normalmente não existe nenhuma restrição em relação ao acesso às interfaces.

Outra observação interessante é que o GNS3 tenta manter a execução em tempo real. Mas embora a diferença de performance entre o PowerPC 860 rodando a 50MHz e Core2 Duo de 2.1GHz pareça grande, a simulação do PowerPC no Core2 Duo nem sempre sem mantém em tempo real, podendo levar a alterações no caso de uma medição mais importante.

Mesmo assim, a qualidade da simulação não é necessariamente afetada, pois mesmo em sistemas Cisco reais existem fatores capazes de causar a perda de performance, por exemplo, no caso de falhas onde as interfaces seriais sobem e descem continuamente, de modo que o roteador não transmite pacotes quando a interface caí. Porém, como os pacotes não são transmitidos e portanto não são perdidos, este tipo de falha acaba sendo visualizado como perda de performance.

Interfaces Virtuais

O GNS3 pode utilizar quantas interfaces ethernet estiverem disponíveis no hospedeiro, porém muitas vezes podemos não ter interfaces suficientes. As interfaces disponíveis podem ser visualizadas no Dynamips com o comando:

```
# sudo ./GNS3.app/Contents/Resources/dynamips-0.2.8-RC2-OSX-Leopard.intel.bin -e
Cisco Router Simulation Platform (version 0.2.8-RC2-x86)
Copyright (c) 2005-2007 Christophe Fillot.
Build date: Nov 11 2007 11:11:35
```

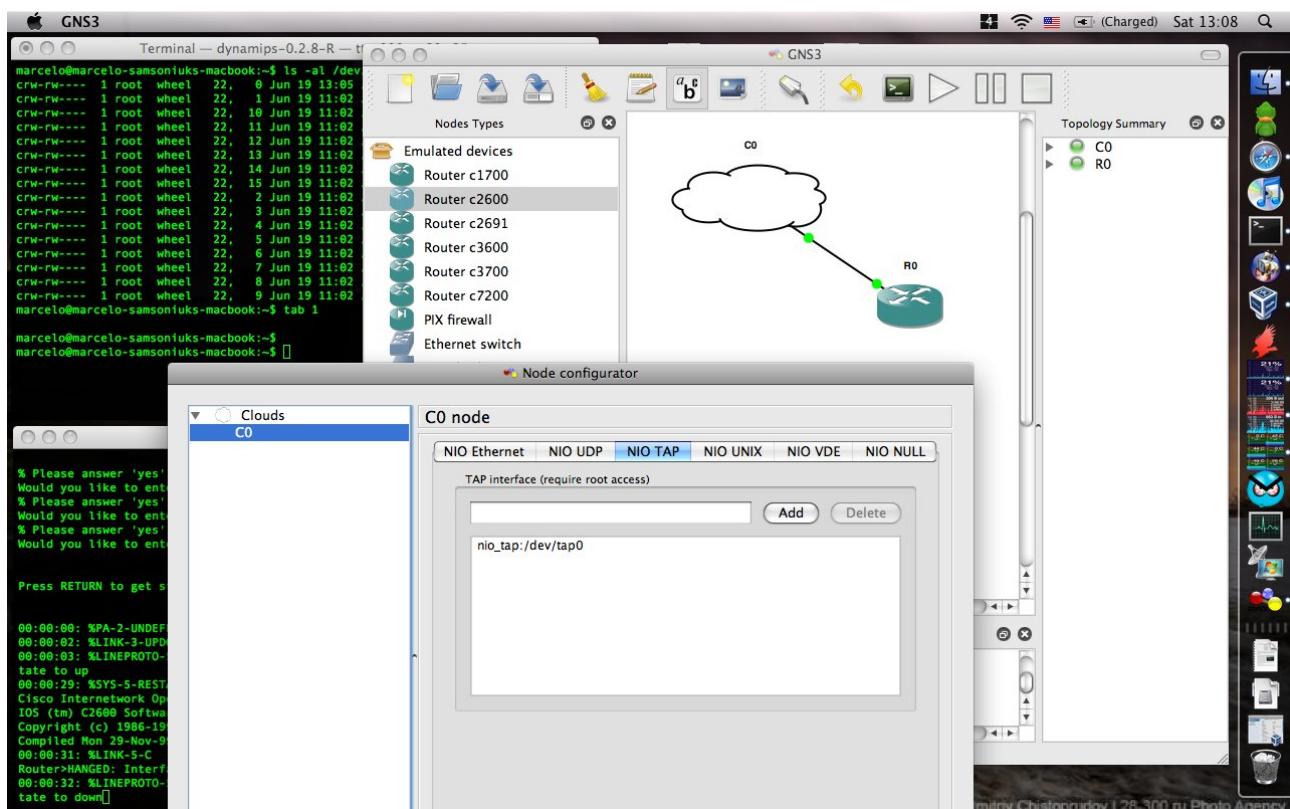
Network device list:

```
en0 : no info provided
fw0 : no info provided
en1 : no info provided
lo0 : no info provided
```

Neste caso temos 4 interfaces e isso pode não ser suficiente para topologias mais complexas!

Uma solução simples para isso é criar interfaces TAP, que são interfaces virtuais que funcionam da mesma forma que as interfaces ethernet, porém sem necessidade do hardware físico.

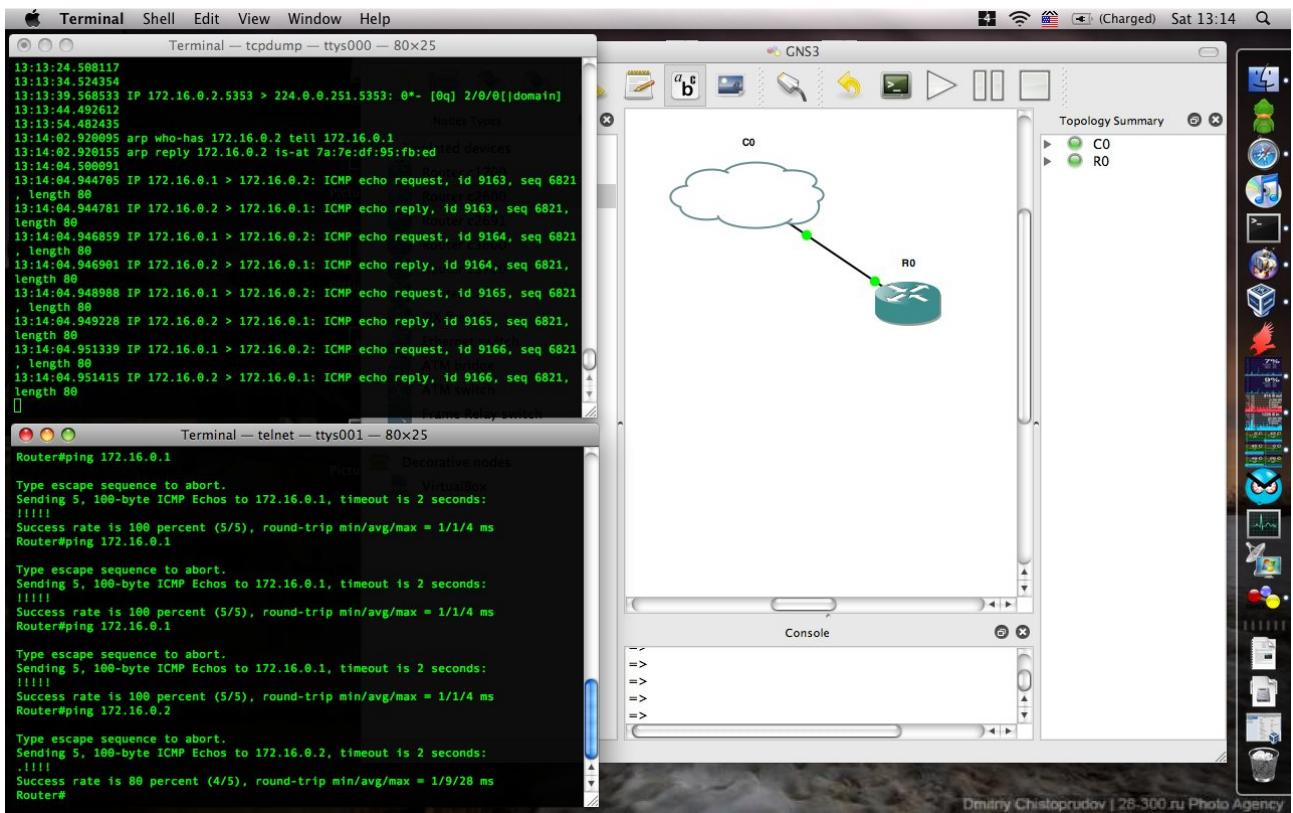
Para testar isso, criamos uma topologia simples com um 2610 conectado pela ethernet em uma nuvem que está configurada para se conectar em uma interface TAP, como ilustrado abaixo:



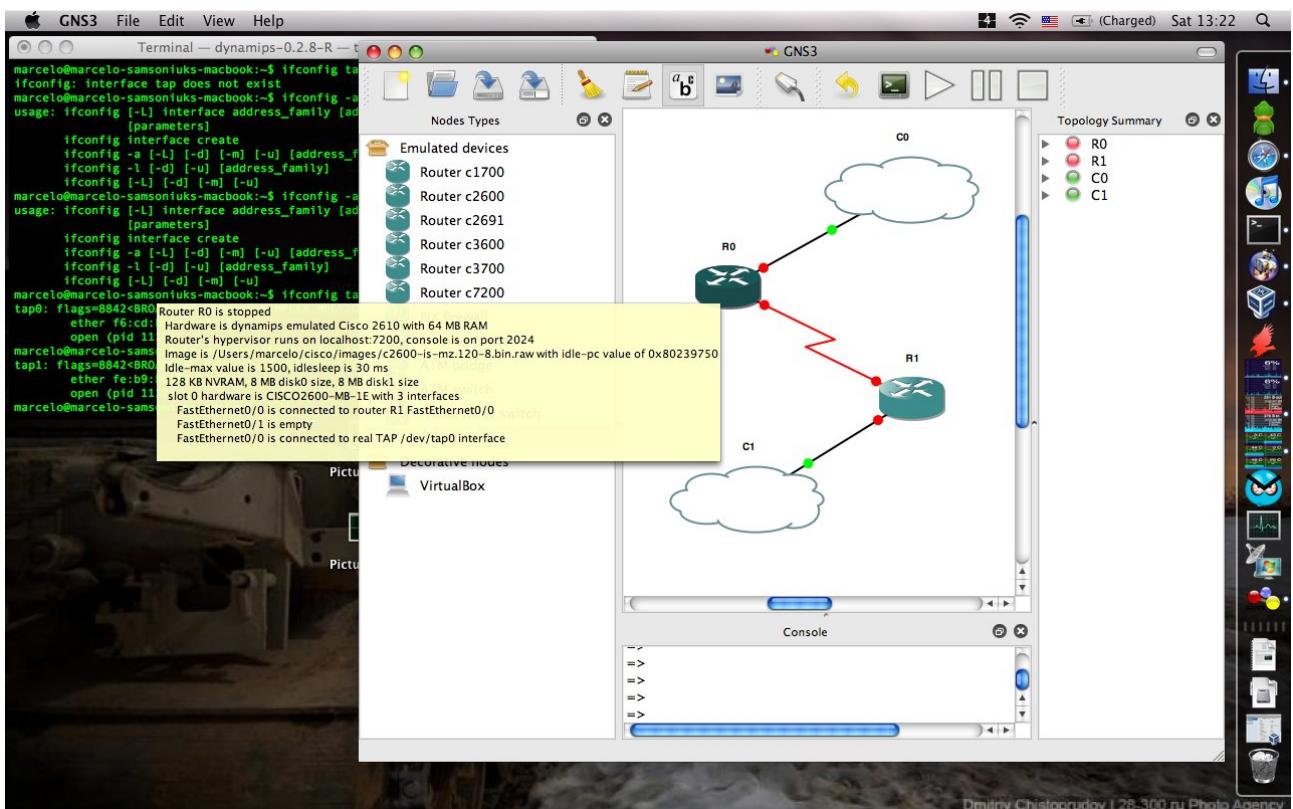
Configurando a ethernet do 2610 com:

```
enable
configure terminal
interface ethernet 0/0
ip address 172.16.0.1 255.255.255.0
no shutdown
^Z
wr
```

Configurando a interface tap0 do OSX com o endereço 172.16.0.2 e monitorando com o tcpdump nesta interface, podemos verificar que existe comunicação entre os equipamentos diretamente:



No caso do OSX podemos utilizar 16 interfaces virtuais separadamente. Estendendo o exemplo para dois 2610 conectados pela interface serial e cada um conectado a uma interface TAP diferente, temos algo como:



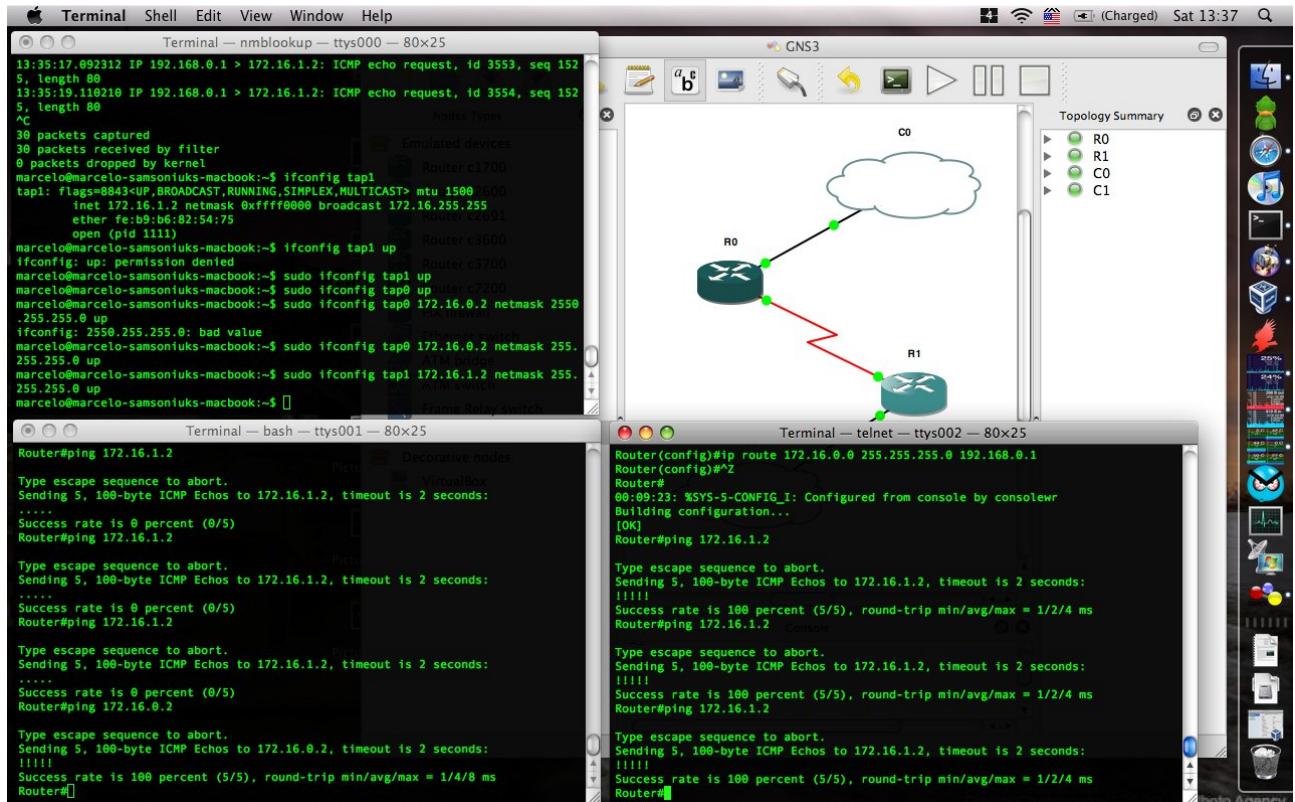
No primeiro 2610 vamos configurar:

```
enable
configure terminal
ip routing
ip 172.16.1.0 255.255.255.0 192.168.0.2
interface ethernet 0/0
ip address 172.16.0.1 255.255.255.0
no shutdown
interface serial 0/0
ip address 192.168.0.1 255.255.255.0
no shutdown
^Z
wr
```

Enquanto que no segundo 2610 temos:

```
enable
configure terminal
ip routing
ip 172.16.0.0 255.255.255.0 192.168.0.1
interface ethernet 0/0
ip address 172.16.1.1 255.255.255.0
no shutdown
interface serial 0/0
ip address 192.168.0.2 255.255.255.0
no shutdown
^Z
wr
```

Configurando as interfaces TAP com os IPs de cada rede, cada roteador deve conseguir pingar para cada interface em sua respectiva rede:



Temos agora então duas interfaces totalmente independentes funcionando no computador hospedeiro, sem utilizar, no entanto, nenhuma interface física. Isto nos permite aumentar o nível de complexidade de nossa simulação, conectando nas interfaces TAP outros simuladores. No nosso caso, vamos criar máquinas virtuais com o VirtualBox da Sun Microsystems.

VirtualBox

O VirtualBox permite criar máquinas virtuais x86 em um computador hospedeiro x86, com qualquer combinação de sistemas operacionais no hospedeiro e nas máquinas virtuais. O VirtualBox pode ser obtido no site:

<http://www.virtualbox.org/wiki/Downloads>

Sendo que a instalação e configuração fogem do escopo de nosso tutorial. Normalmente a virtualização utiliza interfaces reais ou virtuais. Em nosso caso, para podermos conectar as virtualizações ao GNS3, vamos utilizar as interfaces TAP. Neste caso, configuraremos um OpenBSD para utilizar a interface tap0 e um Linux para utilizar a interface tap1. Infelizmente a interface gráfica de gerenciamento do VirtualBox não permite o uso de interfaces TAP, pelo que é necessário configurar e iniciar pelo prompt e comando:

```
VBoxManage modifyvm OpenBSD --nic1 bridged --bridgeadapter1 'tap0: ethernet'  
--nictype1 Am79C973  
VBoxManage modifyvm Slackware12 --nic1 bridged --bridgeadapter1 'tap1: ethernet'  
--nictype1 Am79C973  
VBoxManage startvm OpenBSD  
VBoxManage startvm Slackware12
```

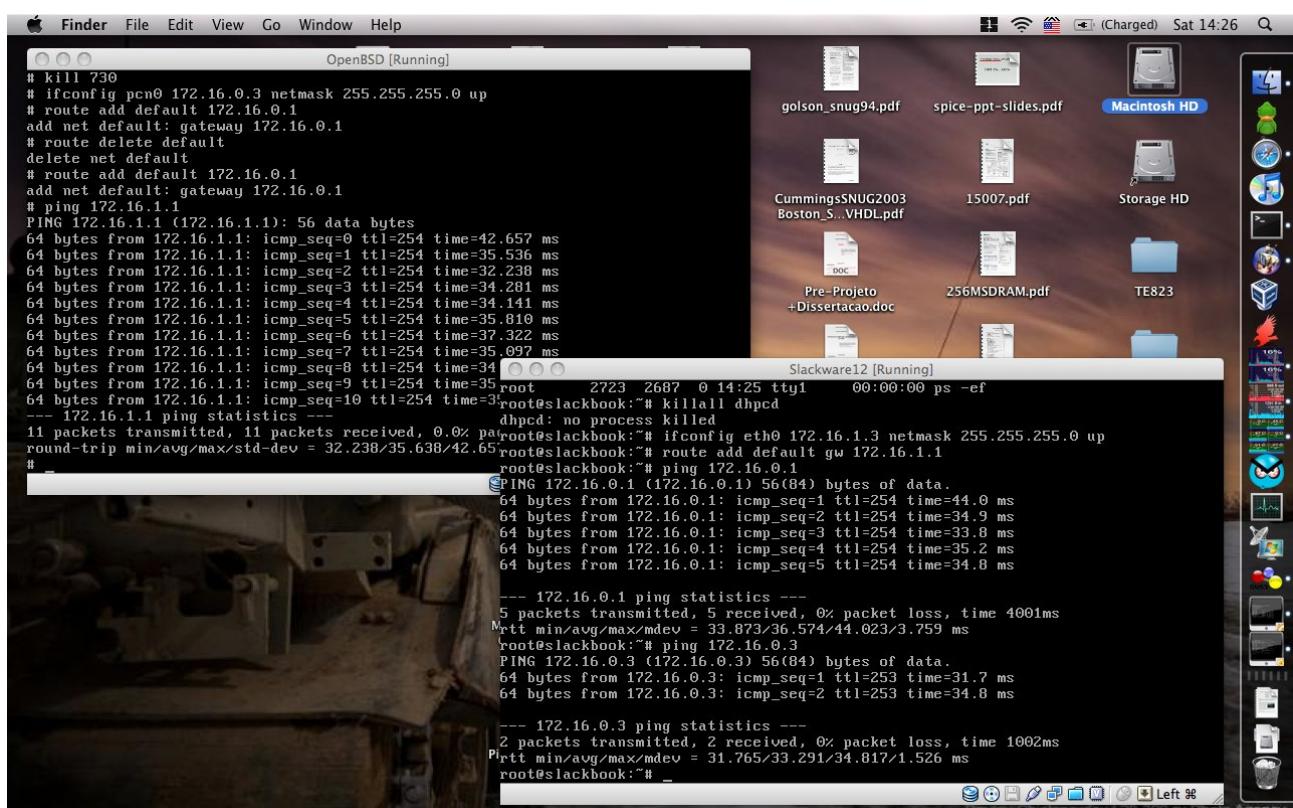
Com as duas máquinas virtuais rodando, basta configurar a rede do OpenBSD:

```
ifconfig pcn0 172.16.0.3 netmask 255.255.255.0 up  
route add default 172.16.0.1
```

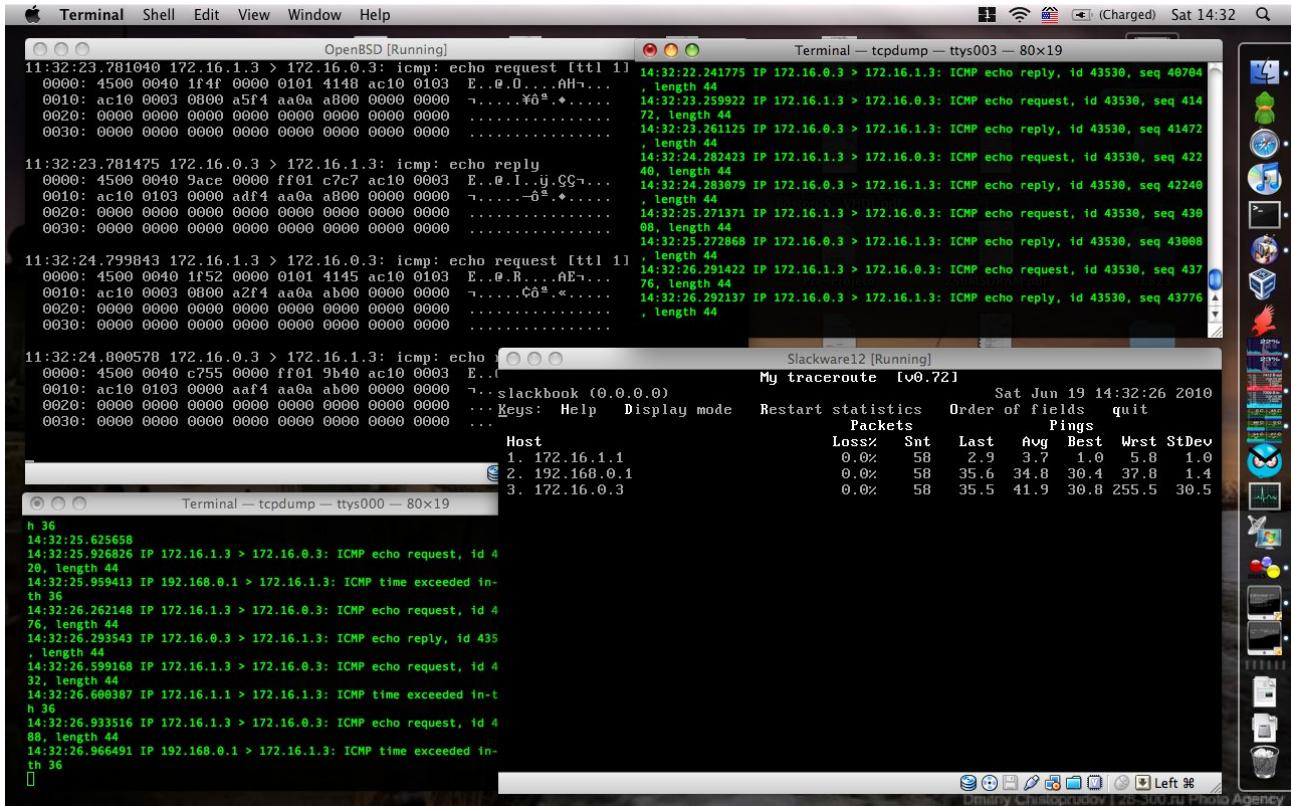
E a rede do Linux:

```
ifconfig eth0 172.16.1.3 netmask 255.255.255.0 up  
route add default gw 172.16.1.1
```

E então as duas máquinas ficam conectadas em rede através dos respectivos roteadores:

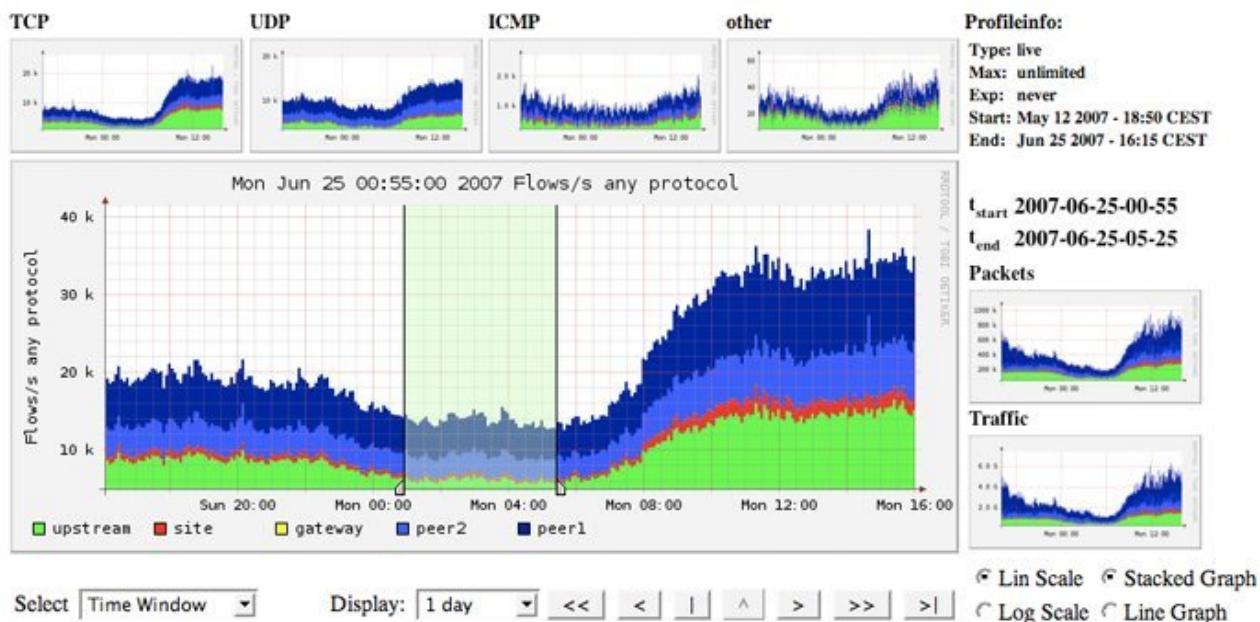


Ou seja, uma máquina OpenBSD conectada via nuvem C0 ao primeiro 2610, conectado por sua vez via serial ao segundo 2610, que esta conectado à uma máquina Linux via nuvem C1, sendo que o computador hospedeiro pode examinar tráfego em qualquer uma destas interfaces:

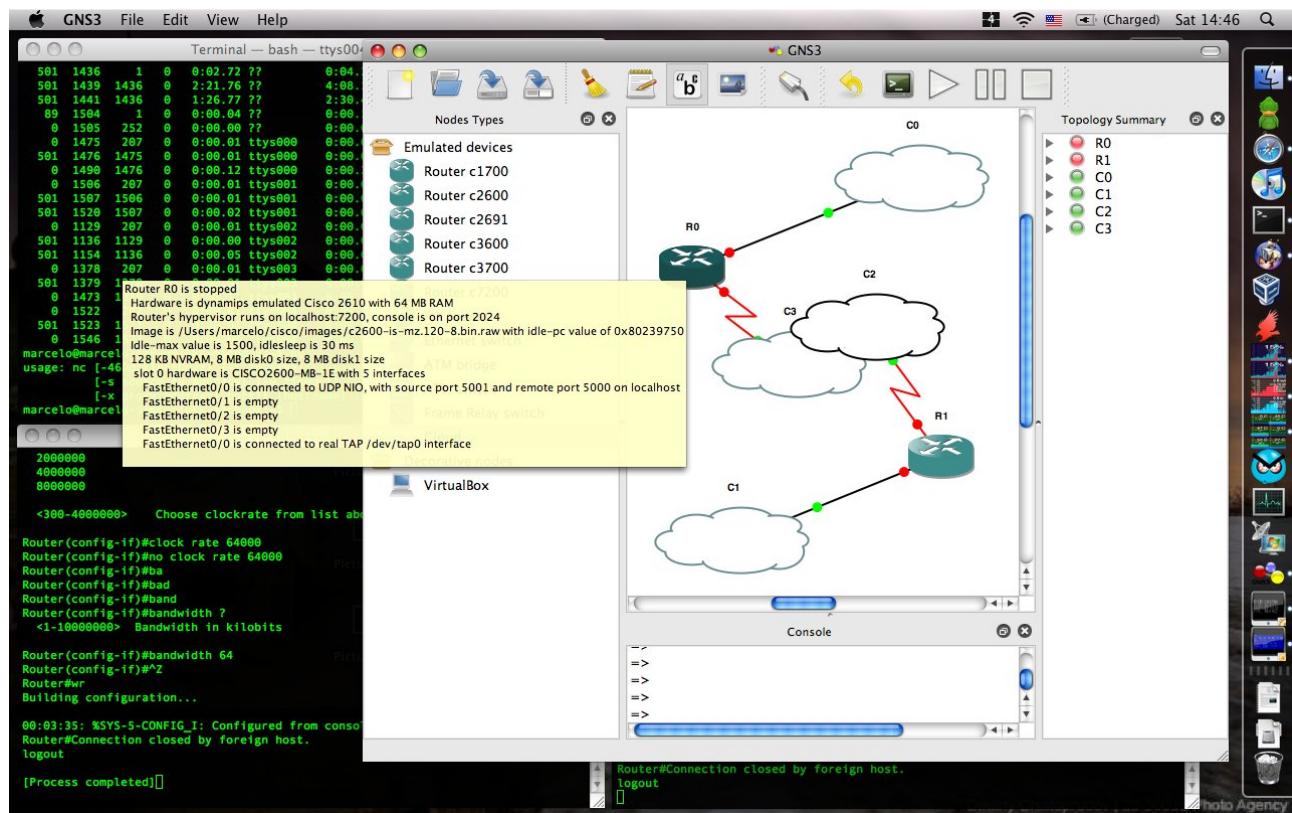


O utilitário MTR confirma que o trajeto passa pelos dois roteadores Cisco, enquanto o computador hospedeiro monitora suas interfaces TAP.

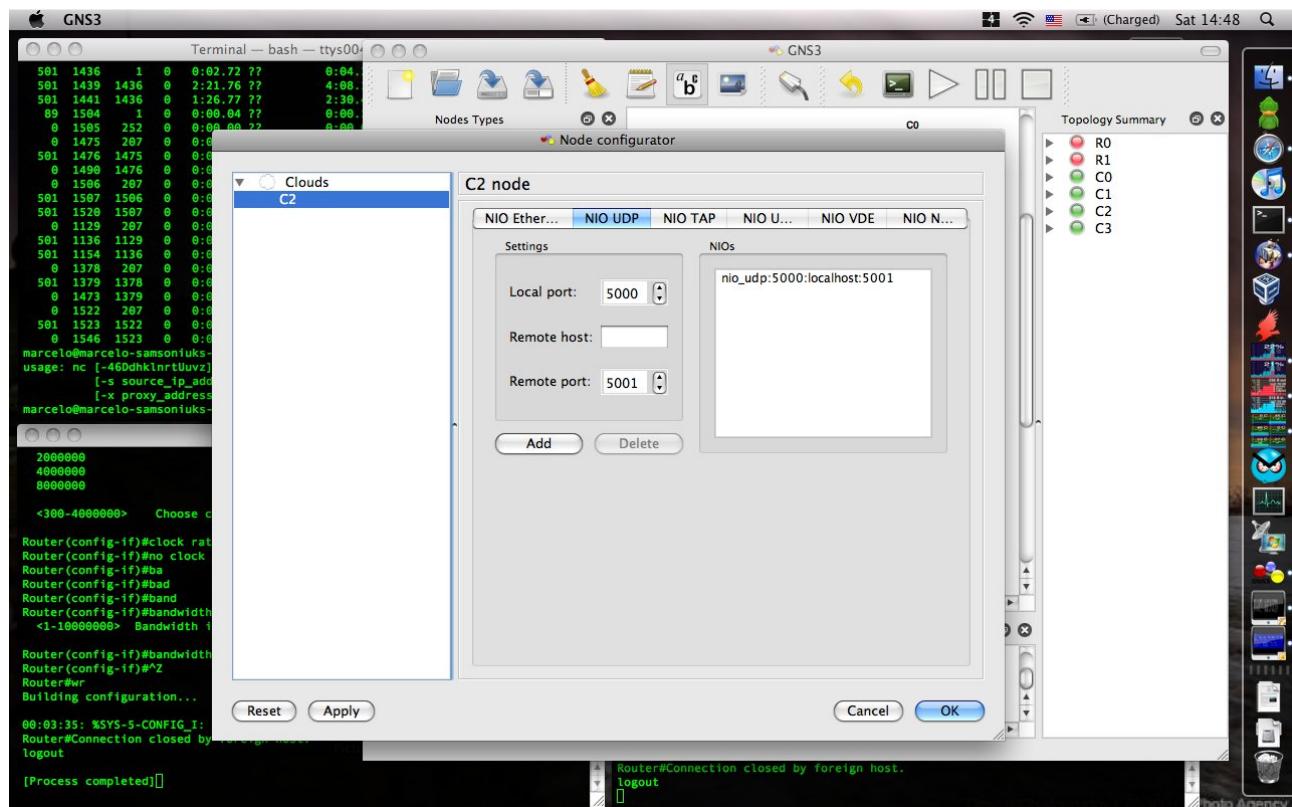
Além de poder monitorar diretamente do computador hospedeiro com o tcpdump, ainda é possível monitorar tráfego em qualquer roteador via SNMP ou NetFlow, sendo possível processar e visualizar as informações em diversas ferramentas de análise de tráfego:



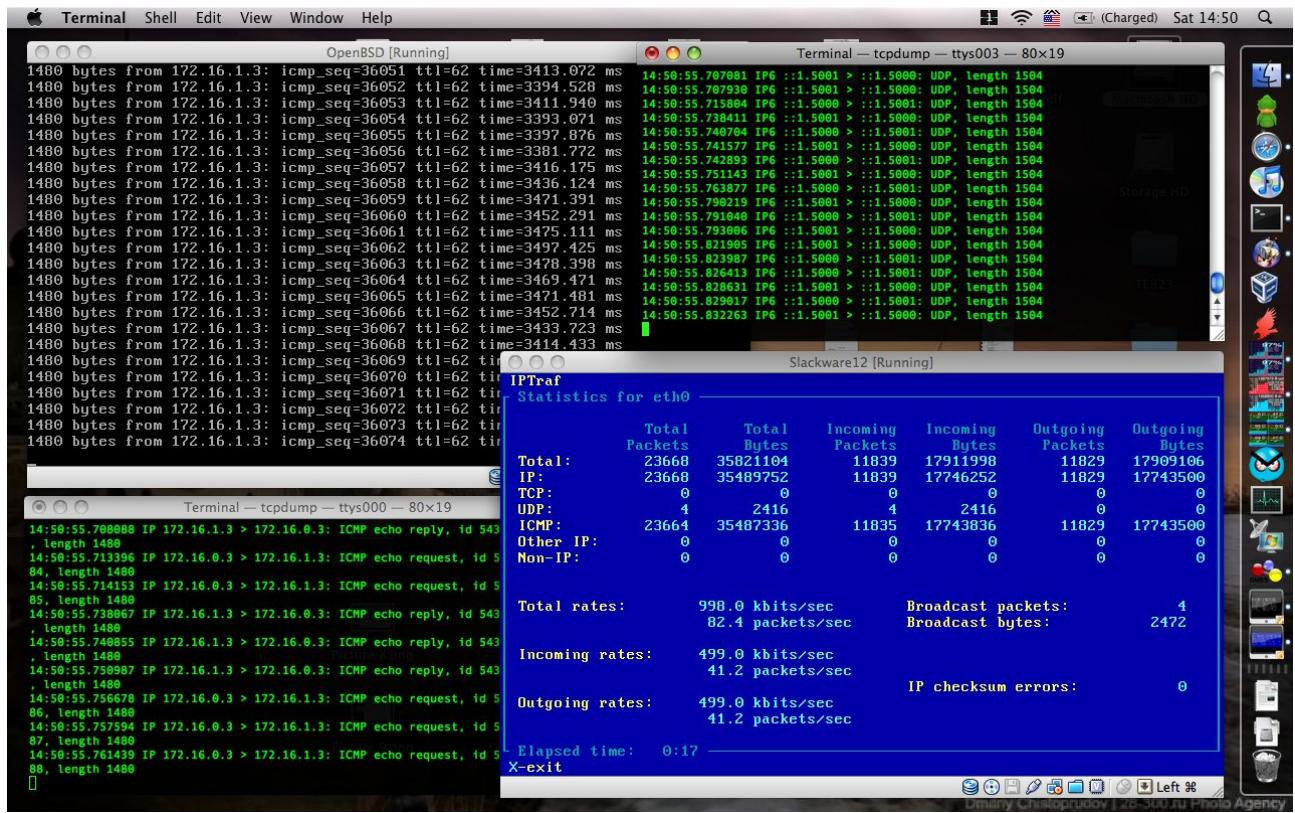
Resta agora resolver a questão do controle de banda, utilizando sockets UDP:



Neste caso, configuramos a nuvem C2 para se conectar à porta UDP de C3 e vice-versa, roteando assim o tráfego das interfaces seriais dos 2610. A figura abaixo detalha a configuração de C2, sendo que C3 é a mesma configuração exceto pelas portas estarem invertidas:



Um teste rápido confirma que o tráfego está realmente passando pelo loopback do hospedeiro:



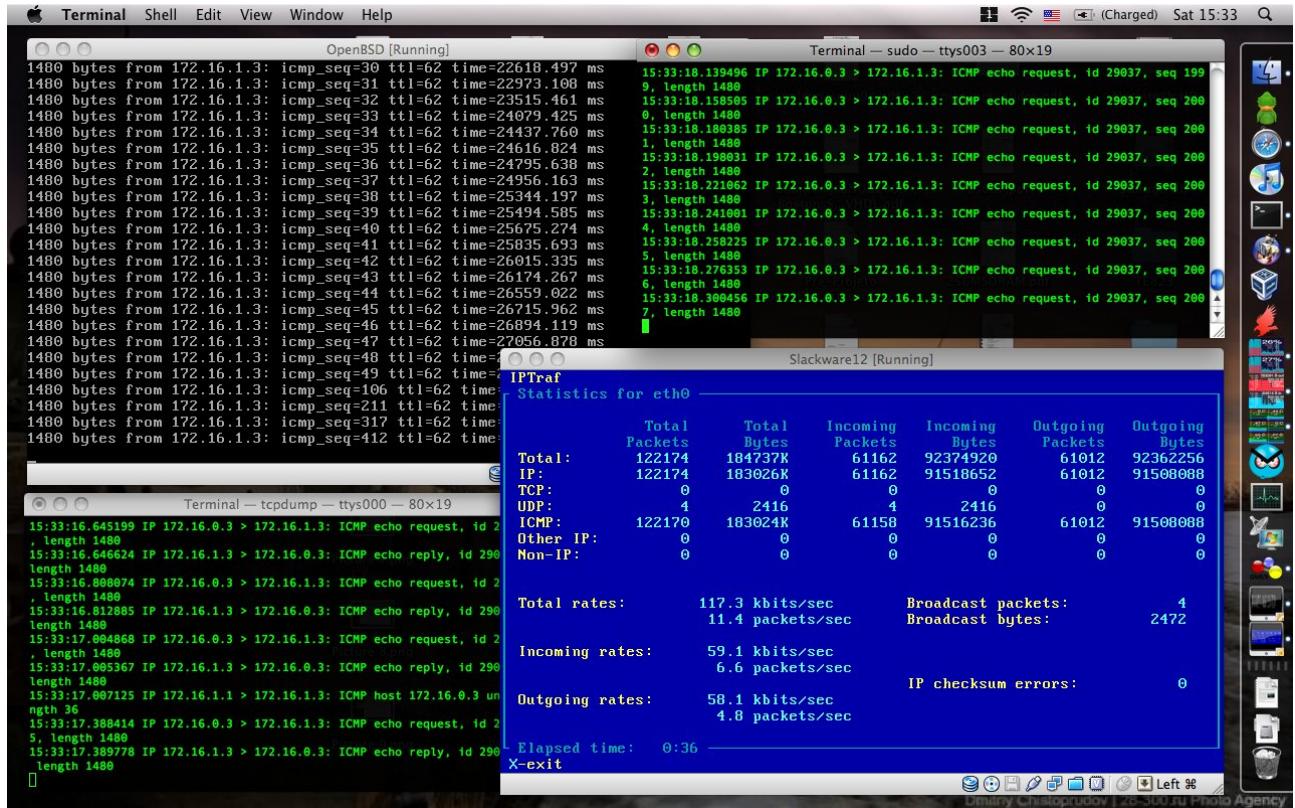
Assim, podemos utilizar os próprios recursos de firewall do OSX para controlar a banda:

```
ipfw pipe 1 config bw 64kbit/s
ipfw pipe 2 config bw 64kbit/s
ipfw add 1 pipe 1 udp from any to any 5000 via lo0
ipfw add 2 pipe 2 udp from any to any 5001 via lo0
```

Uma dica muito importante é tentar não misturar IPv4 com IPv6!

Na figura acima, podemos ver que o tráfego configurado para localhost:5000 e localhost:5001 está na verdade passando pelo loopback como IPv6, porém o ipfw na verdade está esperando por um tráfego UDP com IPv4. Uma solução é descobrir a correta sintaxe do ipfw para IPv6, outra solução é desligar o suporte a IPv4 e utilizar a configuração 127.0.0.1:5000 e 127.0.0.1:5001. Em função do manual do ipfw não ser muito claro sobre o uso de IPv6, ficamos com a última opção, ou seja, passamos toda a configuração para IPv4.

Com isso teremos o tráfego da serial limitado em 64kbit/s:



O que significa que podemos agora controlar os enlaces também: para derrubar os caminhos de TX e RX, basta utilizar as regras:

```
ipfw add 3 drop udp from any to any 5001 via lo0
ipfw add 3 drop udp from any to any 5000 via lo0
```

Bastando remover a regra 3 para que o enlace volte a funcionar.

Como os roteadores possuem capacidade para multiplas interfaces seriais, é possível adicionar caminhos extras, por exemplo, com um novo par de nuvens passando pelas portas UDP 5002 e 5003, de modo que os dois roteadores 2610 estejam conectados pelas duas interfaces seriais. Adicionando um switch para as redes 172.16.0.0/24 e 172.16.1.0/24, é possível também detalhar melhor a topologia de rede, mostrando as máquinas virtuais nas respectivas redes.

Para a configuração adicional do primeiro 2610 temos:

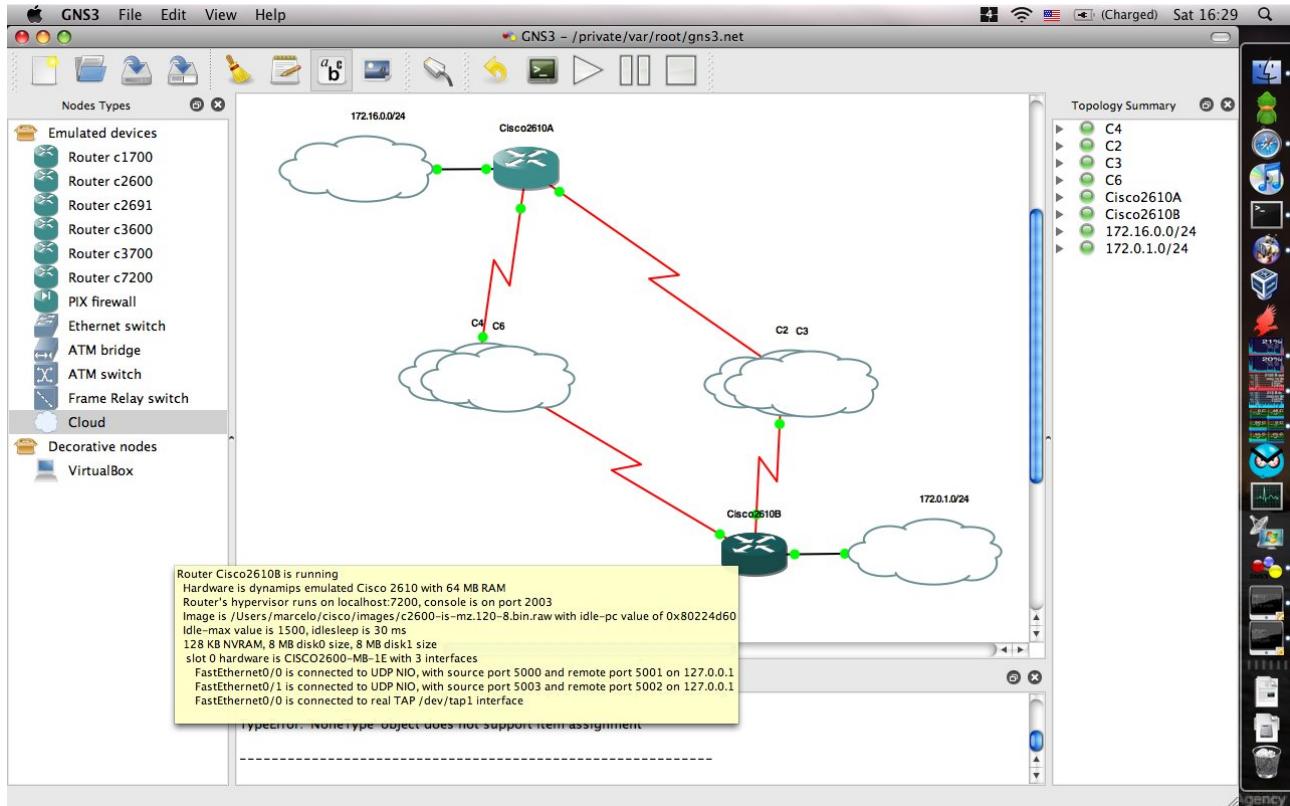
```
configure terminal
ip route add 172.16.1.0 255.255.25.0 192.168.1.2
interface serial 0/1
ip address 192.168.1.1 255.255.255.0
no shutdown
^Z
wr
```

Enquanto que para o segundo 2610 temos o adicional de:

```
configure terminal
ip route add 172.16.0.0 255.255.25.0 192.168.1.1
interface serial 0/1
ip address 192.168.1.2 255.255.255.0
no shutdown
^Z
wr
```

Assim, temos uma seguinte topologia com dois enlaces, que normalmente distribuem a carga e, em caso de falha, assumem o tráfego um do outro, pois cada roteador irá ficar com duas rotas iguais com a mesma métrica para interfaces diferentes.

A figura abaixo ilustra como ficou a configuração final:



Em configuração normal, o tráfego é balanceado por ambas as interfaces, porém quando uma interface caí, o tráfego é desviado para a interface que se mantém em operação. Um efeito colateral do socket UDP é que as interfaces não detectam realmente as condições operacionais, sendo necessário mais estudos no sentido de criar condições mais favoráveis de simulação:

- Uma solução seria rotear cada serial por uma interface TAP e criar uma bridge de tráfego TAP, podendo assim subir e descer a interface e com isso passando informação adicional para o IOS.
- Outra solução seria criar interfaces seriais virtuais aos pares, utilizando os device drivers ethernet como base e encadeando os ring-buffers de TX e RX de modo que as interfaces se comuniquem. Esta solução teria a vantagem adicional de ser possível controlar a cadência dos ring-buffers, limitando também a banda.

Por outro lado, isto pode simular quedas de tráfego não visíveis por parte dos roteadores, por exemplo, quando as interfaces da rede pública se mantém operacionais, mas o núcleo da rede falha por algum motivo.

O GNS3 é uma ferramenta ainda em desenvolvimento e possivelmente muitas melhorias ainda podem ser feitas. Por outro lado, muito ainda pode ser explorado no aspecto de geração de tráfego nas máquinas virtuais e roteamento nas máquinas simuladas.

Conclusão

O GNS3 é um simulador com grande potencial para desenvolvimento futuro: não apenas pode ser estendido no sentido de suportar outros sistemas operacionais além do IOS (por exemplo, o JunOS), como pode incluir outros simuladores (por exemplo, o QEMU) e também máquinas virtuais (através do mapeamento de uma interface virtual em uma nuvem). A interface gráfica possui grande qualidade e o conjunto como um todo é incrivelmente robusto e sólido.

Para desenvolvedores em geral, a simulação de roteadores Cisco representa a oportunidade ideal para validar produtos, na medida que a Cisco é considerada referência no mercado IP. Ao mesmo tempo, é interessante observar que o GNS3 é um simulador desenhado para operar sempre em tempo real. Isto pode dificultar simulações casadas de HW e eventualmente mascarar detalhes de temporização em casos mais críticos.