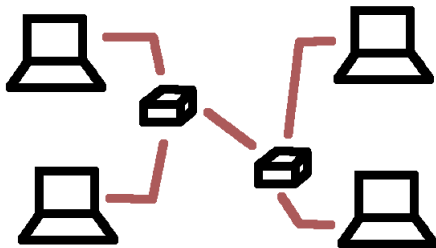


NetSim: um simulador de redes simplificado.

Carlos Eduardo Leão Elmadjian Renan Fichberg

Instituto de Matemática e Estatística da Universidade de São Paulo (IME-USP)

25 de Novembro de 2014



- O simulador: como funciona?
- Testes realizados com o simulador
- Dificuldades encontradas

O simulador: como funciona?

- Estrutura do NetSim
- Entradas
- Saídas
- As classes e as suas funções

O simulador: como funciona?

Estrutura do NetSim

O NetSim é dividido em módulos. Isso foi pensado desta maneira pois, logo de inicio, foi imaginado que haveriam muitas classes. Como o programa foi desenvolvido em Java, ao compilá-lo, há um boom na quantidade de arquivos com o surgimento dos de extensão .class. Com isto, para diminuir o caos no diretório que contém o código-fonte, há outros dois sub-diretórios:

- **/inputs** - Contém arquivos .txt com entradas para alimentar o simulador. É aqui que o usuário deve deixar suas entradas, **obrigatoriamente**.
- **/logs** - Contém arquivos .log com as saídas. As saídas são os pacotes capturados pelos *Sniffers* definidos na entrada que alimentou o programa.

O simulador: como funciona?

- Estrutura do NetSim
- Entradas
- Saídas
- As classes e as suas funções

O simulador: como funciona?

Entradas

Há no total apenas **12** tipos de entrada esperadas pelo programa.
São elas:

- Entrada para criação de computadores (*hosts*)
- Entrada para criação de roteadores (*routers*)
- Entrada para criação de enlaces do tipo duplex-link
- Entrada para configuração dos *hosts* com relação aos endereços de IP do próprio computador, do roteador padrão e do servidor DNS

O simulador: como funciona?

Entradas

- Entrada para configuração dos *routers* com relação às portas e aos endereços de IP
- Entrada para a configuração dos *routers* com relação às rotas
- Entrada para a configuração dos *routers* com relação aos seus dados de *performance*
- Entrada para a configuração dos agentes da camada de aplicação: declaração do agente e da sua natureza

O simulador: como funciona?

Entradas

- Entrada para a configuração dos agentes da camada de aplicação: associação do agente declarado a um *host*
- Entrada para a configuração dos agentes da camada de aplicação: declaração dos *sniffers*
- Entrada para a configuração dos agentes da camada de aplicação: locais da rede onde os *sniffers* agirão.
- Entrada para a Configuração das comunicações entre os agentes.

O simulador: como funciona?

- Estrutura do NetSim
- Entradas
- Saídas
- As classes e as suas funções

O simulador: como funciona?

Saídas

As saídas, como já mencionado anteriormente, são os resultados das capturas dos pacotes pelo *sniffers*, e estas ficam armazenadas por default no sub-diretório de logs. Cada *sniffer* tem seu próprio log, cujo o nome do arquivo tem o formato **nome_do_sniffer.log**. Este é o comportamento do programa para caso o usuário não passe uma saída da sua escolha.

O simulador: como funciona?

Saídas

O formato da saída é o seguinte:

- Identificador do pacote
- Instante de tempo em que o pacote foi visto (a partir da execução do programa)
- Identificador do *sniffer*
- Informações da camada de rede (IP):
 - IP de origem
 - IP de destino
 - Identificação do protocolo da camada acima
 - Tamanho cabeçalho IP + tamanho das camadas superiores
 - TTL

O simulador: como funciona?

Saídas

- Informações da camada de transporte (se for TCP):
 - Porta origem
 - Porta destino
 - Tamanho cabeçalho TCP + tamanho da camada superior
 - Número de Seqüência
 - Número de Reconhecimento
 - Bit ACK
 - Bit FIN
 - Bit SYN
- Informações da camada de transporte (se for UDP):
 - Porta origem
 - Porta destino
 - Tamanho cabeçalho UDP + tamanho da camada superior

O simulador: como funciona?

Saídas

- Informações da camada de aplicação (DNS/FTP/HTTP):
 - Pergunta ou resposta contida no pacote.

Ainda, além das informações estarem presentes no log, a cada captura os resultados daquele *sniffer* são imprimidos no *prompt* em tempo de execução.

O simulador: como funciona?

- Estrutura do NetSim
- Entradas
- Saídas
- As classes e as seus papeis

O simulador: como funciona?

As classes e as suas funções

- Agent: classe abstrata que representa um agente da rede
- ApplicationLayer: classe que representa uma camada de aplicação
- Clock: classe responsável pelo tempo de execução
- DNSServer: classe que representa um servidor DNS
- DuplexLink: classe que representa um enlace do tipo *duplex-link*

O simulador: como funciona?

As classes e as suas funções

- FTPClient: classe que representa um cliente FTP
- FTPServer: classe que representa um servidor FTP
- Host: classe que representa um *host* da rede
- HTTPClient: classe que representa um cliente HTTP
- HTTPServer: classe que representa um servidor HTTP

O simulador: como funciona?

As classes e as suas funções

- InputReader: classe responsável por manipular os arquivos de entrada do simulador
- NetSim: classe que representa o simulador de redes
- Node: classe que representa um nó. Um nó pode ser tanto um *host* quanto um *router*
- Packet: classe que representa um pacote
- Router: classe que representa um *router* da rede

O simulador: como funciona?

As classes e as suas funções

- RouterBuffer: classe que representa um buffer do *router*.
Funciona em FIFO
- SimulatorLogger: classe responsável por escrever os arquivos de saída
- Sniffer: classe que representa um *sniffer*
- TCP: classe que armazena os dados do TCP na camada de transporte
- TransportLayer: classe abstrata que representa uma camada de transporte
- UDP: classe que armazena os dados do UDP na camada de transporte

O simulador: como funciona?

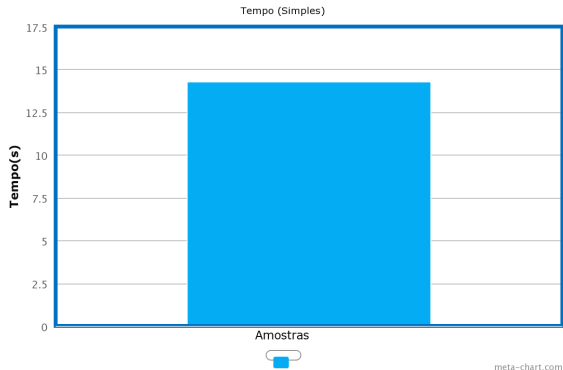
- Estrutura do NetSim
- Entradas
- Saídas
- As classes e as seus papeis

- O simulador: como funciona?
- Testes realizados com o simulador
- Dificuldades encontradas

Aqui serão apresentados gráficos dos testes. O valor da barra representado nas imagens é a **média**. Mais informações podem ser encontradas no arquivo **testes.txt**. Valores medidos para um intervalo de confiança de 95%.

Testes realizados com o simulador

Cenário simples - Tempo

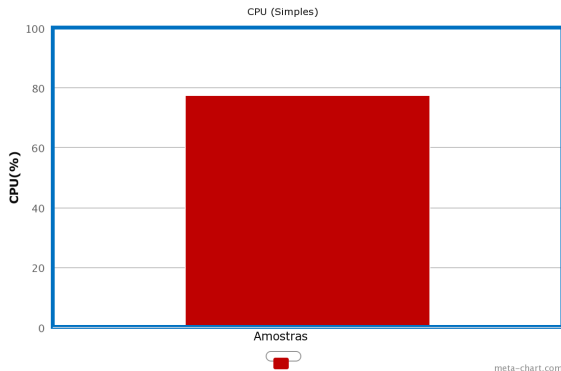


Média = 14.3107

IC = [14.294149138 ; 14.327250862]

Testes realizados com o simulador

Cenário simples - CPU

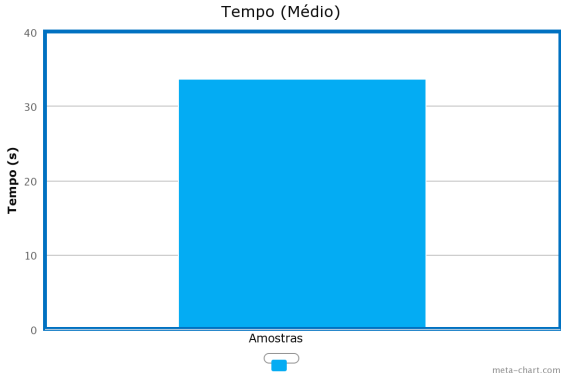


Média = 77.3667

IC = [77.18370188 ; 77.54969812]

Testes realizados com o simulador

Cenário intermediário - Tempo

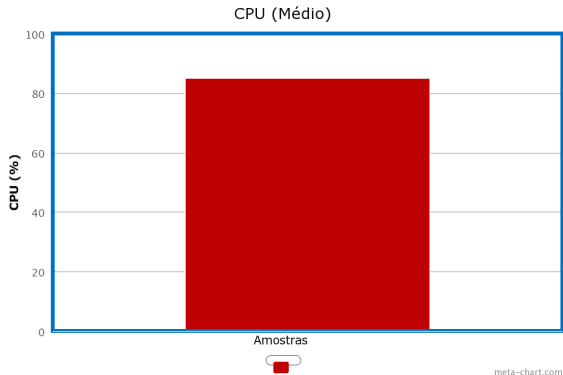


Média = 33.615

IC = [33.570252673 ; 33.659747327]

Testes realizados com o simulador

Cenário intermediário - CPU



Média = 85

IC = [85 ; 85]

Testes realizados com o simulador

Informações das máquinas do experimento

- Máquina real:
 - Intel i5 2500K 3.3GHz, 8GB de RAM
 - Placa de rede Realtek RTL8111/8168/8411 – 1Gbit/s
 - HD Seagate 7200RPM 1TB
 - Sistema Operacional: Arch Linux x86_64 (kernel 3.16.3-1)

Testes realizados com o simulador

Informações das máquinas do experimento

- Máquina virtual:
 - 1 processador da máquina hospedeira, 2GB de RAM
 - Placa de rede em modo bridge
 - Sistema Operacional: Elementary OS x86_32 (kernel 3.2.0-7)

Testes realizados com o simulador

Análise



- O simulador: como funciona?
- Testes realizados com o simulador
- Dificuldades encontradas

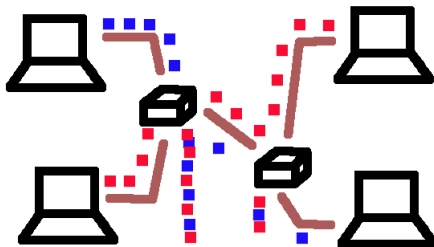
Dificuldades:

- Planejamento: encontrar a estrutura ideal que comporte os seis protocolos requisitados
- Concorrência: encontrar a forma ideal para lidar com todos os pacotes circulando pela rede simultaneamente
- Integração entre todas as classes do programa

Pontos positivos:

- Aprendizado: mais conhecimento sobre o funcionamento das camadas e como elas interagem
- Menos problemas para planejar a estrutura do programa graças a bagagem do EP2

- O simulador: como funciona?
- Testes realizados com o simulador
- Dificuldades encontradas e pontos positivos



Obrigado! :)