UNIVERSIDADE DE COIMBRA
DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

INTRODUÇÃO ÀS REDES DE COMUNICAÇÃO

RELATÓRIO TRABALHO 1

Novembro, 2016

Autores:
Teresa Salazar, 2015234237
Gonçalo Amaral, 2015249122

INTRODUÇÃO

Com este trabalho pretende-se analisar e comparar a transmissão de dados usando os protocolos UDP e TCP utilizando o NS2.

Usando a rede especificada, constituída por PCs e routers, em que os PCs também fazem o routing de pacotes de dados, o "PC A" vai enviar ao "PC E" um bloco de dados de 2MB, que começa a ser transmitido no instante 0.5 segundos.

Dependendo do cenário, poderá haver um envio adicional de informação por UDP.

O Protocolo UDP (User Datagram Protocol) é um protocolo de envio de informação que, entre os dois protocolos, é o mais rápido e mais simples pois não fornece garantia na entrega dos pacotes.

O Protocolo TCP (Transmission Control Protocol) é mais seguro porque, ao contrário do Protocolo UDP, existe um checksum que vai actualizando o receptor da informação que já recebeu.

NOTAS

Argv0: Cenário – Permite escolher o cenário.

Argv1: Protocolo – Permite alternar o protocolo usado - TCP ou UDP.

Argv2: Janela – Permite definir a janela de transmissão para envios para o protocolo do tipo TCP.

Argv3: Quebra – Permite escolher se queremos activar a quebra de liagação entre "PC C" e o "PC D" – 0 ou 1.

Argv4: Velocidade – Permite escolher a velocidade de ligação entre o "PC A" e o "PC B". Para a todos os exercícios, exceto o 4.3, deverá ter o valor de 10Mb.

Para correr o projecto é necessário correr o comando:

ns project.tcl <cenario> <protocol> <window> <break> <velocidade>

Para chamar o trace_analyzer fomos modificando os valores de type - cbr ou tcp caso a ligação seja udp ou tcp. O float, source e destination são sempre 1, 0 e 7.

EXERCÍCIO 2

Tamanho por omissão das filas nos nós	50
Tamanho por omissão dos pacotes TCP	1000
Tamanho por omissão dos pacotes UDP	1000
Tamanho por omissão da janela do TCP	20

EXERCÍCIO 3.1

ns project.tcl 1 udp 0 0 10Mb
awk -f trace_analyzer.awk type=cbr src=0 dest=7 flow=1 out.tr

Statistics for cbr from node 0 to 7 in flow 1
Total sent: 2098
Total received: 2085
Lost packets: 13
Average delay: 000.891840
Total transmission time: 001.736162

No caso do TCP, a janela influencia o resultado. Para janelas abaixo de 34, o número total de pacotes enviados não era enviado. No entanto, à medida que a janela aumenta, o Average delay e o total transmission time variam.

Após várias tentativas, a janela que minimiza o tempo total de transmissão do bloco de dados entre o "PC A" e o "PC E" sem perda de pacotes é 88.

ns project.tcl 1 tcp 101 0 10Mb
awk -f trace_analyzer.awk type=tcp src=0 dest=7 flow=1 out.tr

Statistics for tcp from node 0 to 7 in flow 1
Total sent: 2099
Total received: 2099
Lost packets: 0
Average delay: 000.045320
Total transmission time: 002.264192

TCP			UDP	
Tempo min	Janela min	Nº pacotes perdidos	Tempo min	Nº pacotes perdidos
2.264192	101	0	1.736162	13

EXERCÍCIO 3.2

ns project.tcl 1 tcp 218 1 10Mb
awk -f trace_analyzer.awk type=tcp src=0 dest=7 flow=1 out.tr

Statistics for tcp from node 0 to 7 in flow 1
Total sent: 2124
Total received: 2111
Lost packets: 13
Average delay: 000.064469
Total transmission time: 002.763752

ns project.tcl 1 udp 0 1 10Mb
awk -f trace_analyzer.awk type=cbr src=0 dest=7 flow=1 out.tr

Statistics for cbr from node 0 to 7 in flow 1
Total sent: 2098
Total received: 2085
Lost packets: 13
Average delay: 000.891840
Total transmission time: 001.736162

TCP			UDP	
Tempo min	Janela min	Nº pacotes perdidos	Tempo min	Nº pacotes perdidos
2.763752	218	13	1.736162	13

EXERCÍCIO 4.1

ns project.tcl 2 udp 0 0 10Mb
awk -f trace_analyzer.awk type=cbr src=0 dest=7 flow=1 out.tr

Statistics for cbr from node 0 to 7 in flow 1
Total sent: 2098
Total received: 1318
Lost packets: 780
Average delay: 000.893865
Total transmission time: 001.736888

ns project.tcl 2 tcp 20 0 10Mb
awk -f trace_analyzer.awk type=tcp src=0 dest=7 flow=1 out.tr

Statistics for tcp from node 0 to 7 in flow 1
Total sent: 1251
Total received: 1245
Lost packets: 6
Average delay: 000.043693
Total transmission time: 005.499968

TCP		UDP	
Tempo	Nº pacotes perdidos	Tempo	Nº pacotes perdidos
5.499968	6	001.736888	780

EXERCÍCIO 4.2

ns project.tcl 2 udp 0 1 10Mb
awk -f trace_analyzer.awk type=cbr src=0 dest=7 flow=1 out.tr

Statistics for cbr from node 0 to 7 in flow 1
Total sent: 2098
Total received: 1302
Lost packets: 796
Average delay: 000.894443
Total transmission time: 001.733210

ns project.tcl 2 tcp 20 1 10Mb
awk -f trace_analyzer.awk type=tcp src=0 dest=7 flow=1 out.tr

Statistics for tcp from node 0 to 7 in flow 1
Total sent: 1128
Total received: 1111
Lost packets: 17
Average delay: 000.043887
Total transmission time: 005.499472

TCP		UDP	
Tempo	Nº pacotes perdidos	Tempo	Nº pacotes perdidos
5.499472	17	1.733210	796

EXERCÍCIO 4.3

ns project.tcl 2 udp 0 0 4Mb
awk -f trace_analyzer.awk type=cbr src=0 dest=7 flow=1 out.tr

Statistics for cbr from node 0 to 7 in flow 1
Total sent: 2098
Total received: 2098
Lost packets: 0
Average delay: 002.141477
Total transmission time: 004.236642

ns project.tcl 2 tcp 46 1 10Mb
awk -f trace_analyzer.awk type=tcp src=0 dest=7 flow=1 out.tr

Statistics for tcp from node 0 to 7 in flow 1
Total sent: 1438
Total received: 1431
Lost packets: 7
Average delay: 000.046817
Total transmission time: 005.456096

TCP			UDP		
Tempo min	Janela min	Nº pacotes enviados/recebidos	Tempo min	Nº pacotes perdidos	Velocidade
5.456096	46	1438 / 1431	4.236642	0	4Mb

EXERCÍCIO 5

Tal como se esperava, confirma-se que no protocolo UDP existe mais perda de pacotes. Prova-se então a importância da presença do checksum no protocolo TCP para garantir o envio da informação.

Como consequência, o protocolo UDP consegue reduzir o seu tempo de execução, enquanto que o TCP tem que confirmar qual a informação que já foi recebida e assim perde mais tempo.

EXERCÍCIO 6

O problema presente será que nas ligações PC B - PC D e PC D - PC C, a bandwidth disponivel será menor pois já existe uma stream de dados nestas ligações.

Cada ligação tem uma bandwidth de 10Mb/s mas como na ligaçao PC B a PC D temos uma stream de 6Mb/s, a bandwidth disponivel nesse canal será de 4Mb/s para os restantes dados. Na ligação de PC D a PC C existe uma stream de 5Mb/s tendo uma bandwidth disponível de 5Mb/s. No entanto, como é feita no sentido oposto, não irá interferir com a stream proveniente de PC B.

A bandwidth disponivel irá afetar a quantidade de pacotes que poderão ser enviados por segundo.

Como PC B irá receber pacotes a 10Mb/s e apenas pode enviar para PC C a 4Mb/s, a fila de PC B irá comecar a encher progressivamente, havendo desta forma perda de pacotes.

Ao efetuarmos o corte, o routing será mudado passando os dados de PC A a serem transmitidos por canais que não estão cheios.

Para resolvermos este problema poderemos diminuir a velocidade da ligaçao de PC A a PC B de maneira a a fila de PC B não crescer substancialmente e não existir uma maior perda de pacotes. Também se pode aumentar a fila de PC B apesar de esta solucao não ser a mais conveniente. Finalmente, é possível aumentar a bandwidth das ligações entre PC B e PC D de maneira a haver bandwidth disponível suficiente.