ECEN 602 Machine Problem 5
Network Simulation

Team 11:

1. Dhanraj Murali – UIN:734003894
2. Swetha Reddy Sangem – UIN: 433003591

Implementation Roles:

1. Dhanraj – Worked on code, simulation, and testing.
2. Swetha – Worked on code, simulation, and report.

Submission:

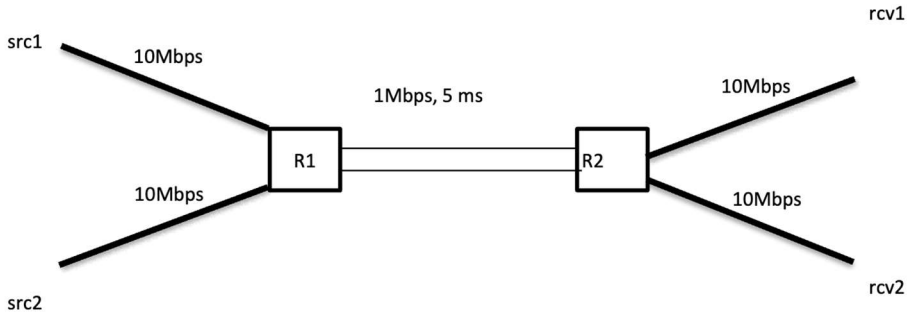
1. Submission 1 – Source Code written by the team.
2. Submission 2 – Source Code written by team and optimized by ChatGPT.
3. Submission 3 – Source Code generated by ChatGPT.
4. report.pdf

Environment Setup:
Ubuntu 22.04
nam version: nam_1.14_amd64.deb

TCP Flavors: SACK, VEGAS

Network Configuration:

The configuration of the network is as shown below:



- Two routers (R1, R2) connected with a 1 Mbps link and 5ms of latency
- Two senders (src1, src2) connected to R1 with 10 Mbps links
- Two receivers (rcv1, rcv2) connected to R2 with 10 Mbps links
- Application sender is FTP over TCP

There are 3 cases of network delays that are to be simulated are:

Case 1:
a. src1-R1 and R2-rcv1 end-2-end delay = 5 ms
b. src2-R1 and R2-rcv2 end-2-end delay = 12.5 ms
Case 2:
a. src1-R1 and R2-rcv1 end-2-end delay = 5 ms
b. src2-R1 and R2-rcv2 end-2-end delay = 20 ms
Case 3:
a. src1-R1 and R2-rcv1 end-2-end delay = 5 ms
b. src2-R1 and R2-rcv2 end-2-end delay = 27.5 ms

Execution:

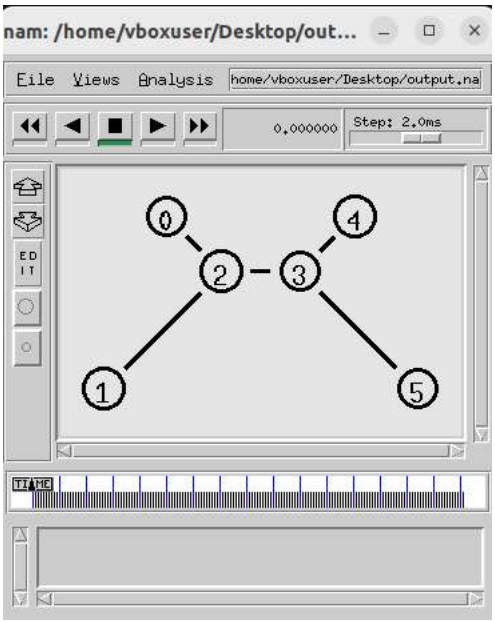
Select the TCP flavor and Case number by sending them as command line arguments.
Command for debugging and simulation:

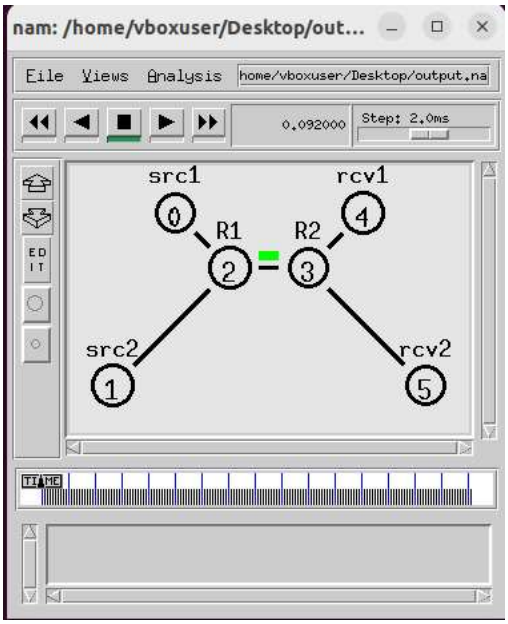
>> ns ns2.tcl <flavor> <case_number>

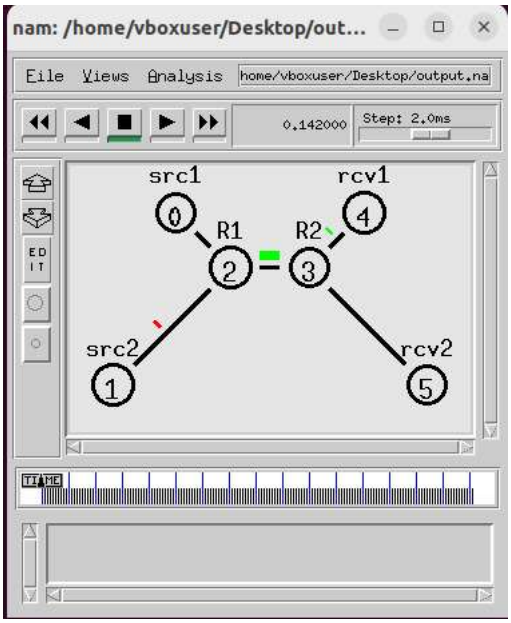
Results:

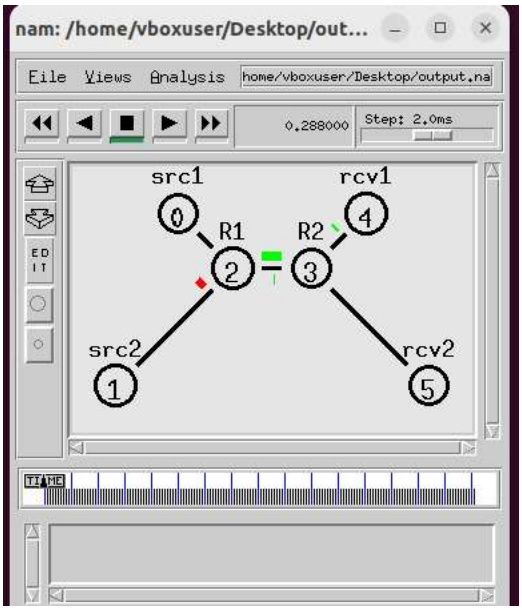
The simulations were run for 400 seconds, and the initial 100 seconds were ignored while measuring the values.

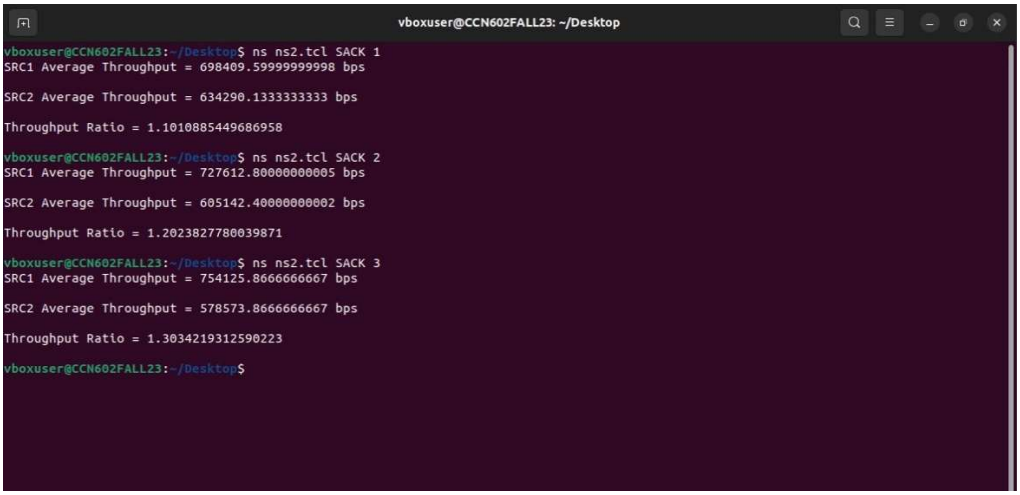
Simulation output for VEGAS 3:
The simulation can be seen through the out.nam file after clicking the play button.

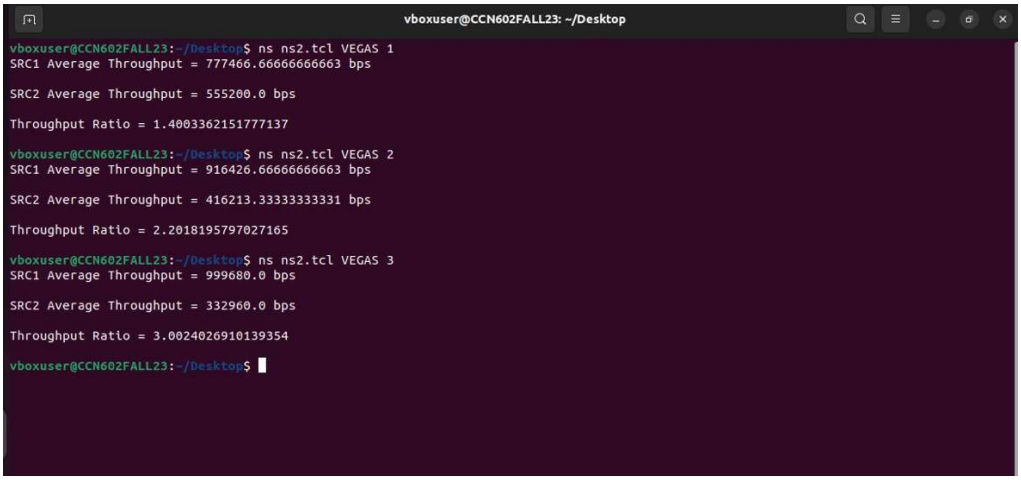





nam: /home/vboxuser/Desktop/out...
File Views Analysis home/vboxuser/Desktop/output.na
0.142000
Step: 2.0ms
src1
rcv1
R1
R2
src2
rcv2
0
1
2
3
4
5


nam: /home/vboxuser/Desktop/out...
File Views Analysis home/vboxuser/Desktop/output.na
0.288000
Step: 2.0ms
src1
rcv1
R1
R2
src2
rcv2
0
1
2
3
4
5


vboxuser@CCN602FALL23: ~/Desktop
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl SACK 1
SRC1 Average Throughput = 698409.59999999998 bps
SRC2 Average Throughput = 634290.1333333333 bps
Throughput Ratio = 1.1010885449686958
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl SACK 2
SRC1 Average Throughput = 727612.80000000005 bps
SRC2 Average Throughput = 605142.40000000002 bps
Throughput Ratio = 1.2023827780039871
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl SACK 3
SRC1 Average Throughput = 754125.8666666667 bps
SRC2 Average Throughput = 578573.8666666667 bps
Throughput Ratio = 1.3034219312590223
vboxuser@CCN602FALL23:~/Desktop$


vboxuser@CCN602FALL23: ~/Desktop
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl VEGAS 1
SRC1 Average Throughput = 777466.66666666663 bps
SRC2 Average Throughput = 555200.0 bps
Throughput Ratio = 1.4003362151777137
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl VEGAS 2
SRC1 Average Throughput = 916426.66666666663 bps
SRC2 Average Throughput = 416213.33333333331 bps
Throughput Ratio = 2.2018195797027165
vboxuser@CCN602FALL23:~/Desktop$ ns ns2.tcl VEGAS 3
SRC1 Average Throughput = 999680.0 bps
SRC2 Average Throughput = 332960.0 bps
Throughput Ratio = 3.0024026910139354
vboxuser@CCN602FALL23:~/Desktop$

Throughput for TCP Flavor SACK:

	Src1	Src2	Ratio
Case 1:	698409.59 bps	634290.13 bps	1.101088
Case 2:	727612.8 bps	605142.4 bps	1.202382
Case 3:	754125.8 bps	578573.8 bps	1.303421

Throughput for TCP Flavor VEGAS:

	Src1	Src2	Ratio
Case 1:	777466.6 bps	555200.0 bps	1.400336
Case 2:	916426.6 bps	416213.3 bps	2.201819
Case 3:	999680.0 bps	332960.0 bps	3.002402

From the throughput outputs obtained above, it can be seen that:

i) The throughput values for TCP SACK remain more consistent as the RTT increases but for TCP VEGAS, the throughput values vary quite significantly. The throughput decreases as the RTT increases, which is significant for TCP Vegas.

ii) Comparing the case 1 where the RTTs of the two sources are in the ratio 1:2, TCP Vegas throughput ratio (1.400336) is higher than TCP SACK throughout ratio (1.101088). TCP Vegas performs better than TCP SACK in all cases.

TCP SACK uses packet loss to denote the congestion in the network.
TCP Vegas focuses on measuring the RTT and uses this information to infer the network congestion. Instead of relying just on packet loss as a signal of congestion, it monitors the changes in RTT.

The reason why TCP Vegas performs better than TCP SACK:
i) TCP Vegas is more stable than TCP SACK which rely on the sudden reduction in transmission rate after packet loss. This is because in SACK, the sender will increase rate until there is congestion and then it cuts back on the rate, which it keeps on the repeating and hence is unstable.
ii) TCP Vegas adapts more quickly to network conditions without the requiring the need to wait for packet loss signals.
iii) TCP Vegas employs delay-based congestion control.
iv) TCP Vegas proactively adjusts the sending rate to avoid congestion before it leads to packet loss.

ChatGPT optimization comments:

Consistent Naming Conventions:

Maintain a consistent naming convention for variables, e.g., choose between camelCase or underscores. In your script, both styles are used. Stick to one for better readability.

Code Structure and Comments:

Break down your code into functions or procedures. This improves readability and makes it easier to understand and modify the code.
Add comments to explain the purpose of each section or function.

Avoid Global Variables:

Minimize the use of global variables. Instead, pass variables as parameters to functions or procedures.

Use Equality Operator (==):
When comparing values, use == instead of ==.

Use incr for Incrementing:
Use incr for incrementing variables instead of set variable [expr $variable + 1].

Use expr with Variables:
When using expr, use curly braces around variable names to avoid issues with spaces.

Use file exists for File Checking:
Instead of using $argc, consider using file exists to check if the file exists.

Reuse Code:
Reuse code where possible to avoid redundancy.

References:

[1] **Simple Simulation Example** http://nile.wpi.edu/NS/simple_ns.html
[2] **Tutorial for Network Simulator ns** https://www.isi.edu/nsnam/ns/tutorial/index.html