TCL script to simulate File Transfer Protocol using NS2. Demonstrate the file upload, file download and file transfer mechanism for the same.

Team Members:

Taduvai Satvik Gupta – BL.EN.U4EAC21075

Aditya Thelu – BL.EN.U4EAC21076

Thupakula Pranavi –BL.EN.U4EAC21077

Tummala Manushri – BL.EN.U4EAC21078

Batch-18

Contents:

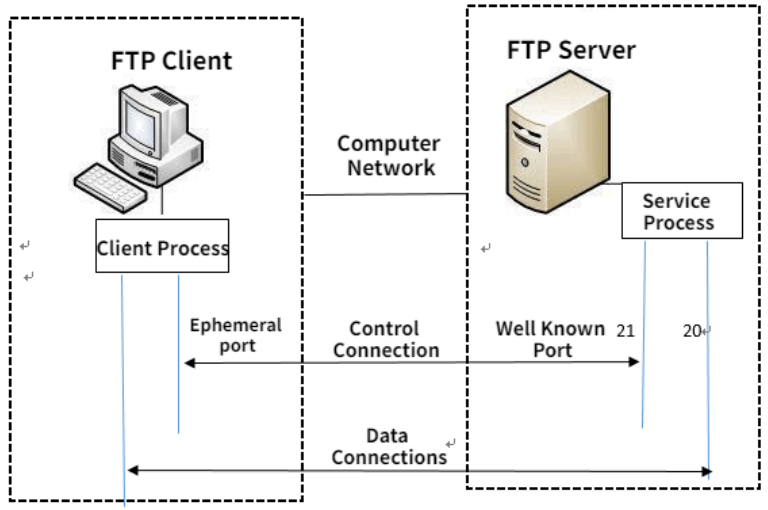
1. Aim
2. FTP
3. Different FTP Protocols
4. NS2 Stimulation
5. Nam Animation
6. Outputs
7. Trace File Performance Analysis
8. Conclusion

Aim:

- To stimulate the File Transfer Protocol (FTP) using Tool Command Language (TCL) in NS2.
- Demonstrate file upload, file download and transfer mechanisms.

FTP:

- **File transfer protocol (FTP)** is an Internet tool provided by TCP/IP.
- FTP is a standard communication protocol. There are various other protocols like HTTP which are used to transfer files between computers, but they lack clarity and focus as compared to FTP.
- FTP employs two connections: a control connection for transmitting commands and receiving responses between the client and server, and a separate data connection for transferring files, facilitating efficient upload and download operations.
- It uses port no 20 and 21 for transferring the data.
- Types of FTP Connections:
  - Active FTP Connection :In an Active FTP connection, the client establishes the command channel, and the server establishes the data channel.
  - Passive FTP Connection: In a Passive FTP connection, the client establishes both the data channel as well as the command channel.



Working Principle of FTP

Different FTP Protocols:

1. **FTP (File Transfer Protocol)**: The original standard protocol for transferring files over a network, operating over TCP/IP.
2. **SFTP (SSH File Transfer Protocol)**: A secure file transfer protocol that utilizes SSH encryption for data confidentiality and integrity.
3. **FTPS (File Transfer Protocol Secure)**: Adds SSL/TLS encryption to FTP connections, providing a secure channel between client and server.
4. **FTPES (FTP over Explicit TLS/SSL)**: Similar to FTPS, FTPES enhances FTP connections with SSL/TLS encryption, with the client explicitly requesting a secure connection.
5. **TFTP (Trivial File Transfer Protocol)**: A lightweight file transfer protocol designed for transferring small files quickly and efficiently, lacking authentication and encryption features.

NS2 Simulation:

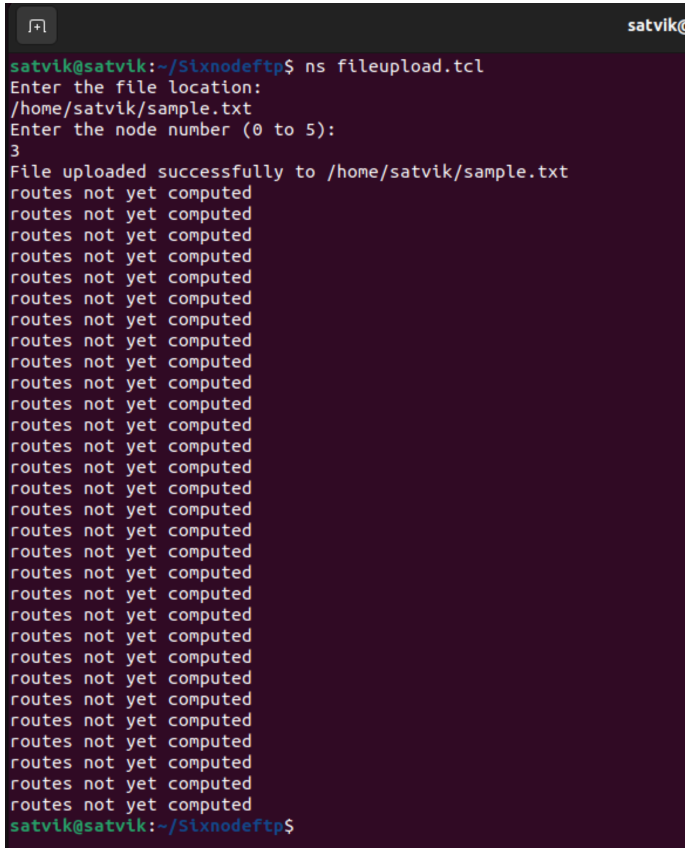
- The FTP procedure is defined to simulate a file transfer using TCP/IP. It takes source node, destination node, file name as parameters.
- The procedure reads the file from the source node, sends it over TCP to the destination node, and saves it there.
- The simulation is halted after 2 seconds.
- This FTP function is called with user inputs and supports transferring a file size of 1MB.

Nam Animation:

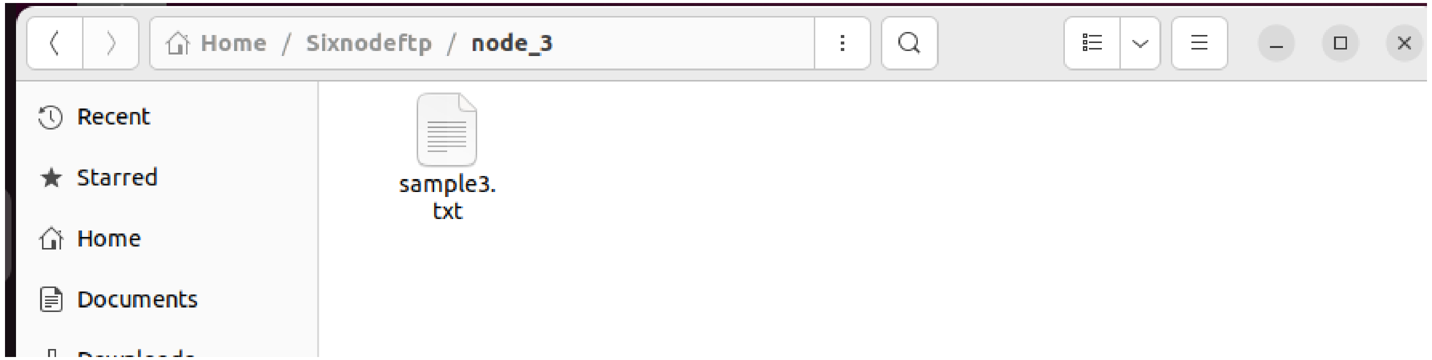
- To visualize the file transfer 6 nodes are created and duplex links are created between them, the traffic is created.
- Traffic generation is achieved using two types of applications: FTP over TCP, CBR over UDP.
- FTP traffic demonstrates bulk file transfers with focus on throughput, latency, packet loss, and congestion
- CBR(Constant Bit Rate) traffic showcases continuous data transmission emphasizing network capacity, jitter, QoS, and resource allocation.

Outputs:

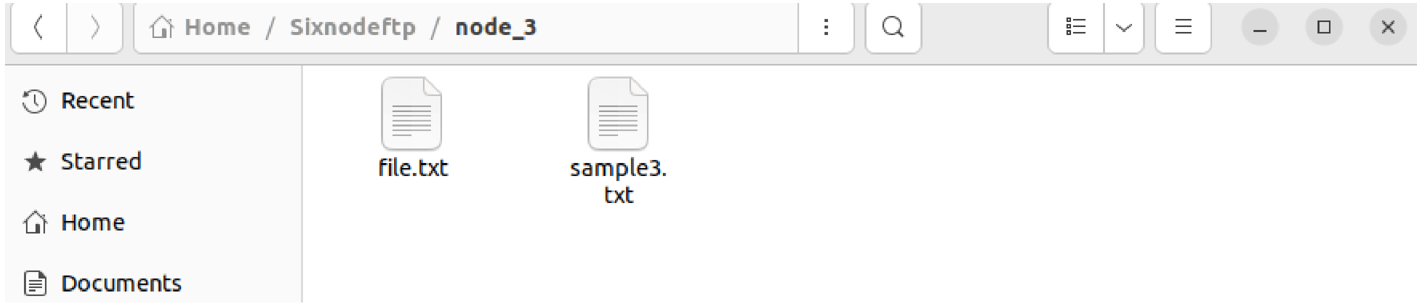
File Upload:



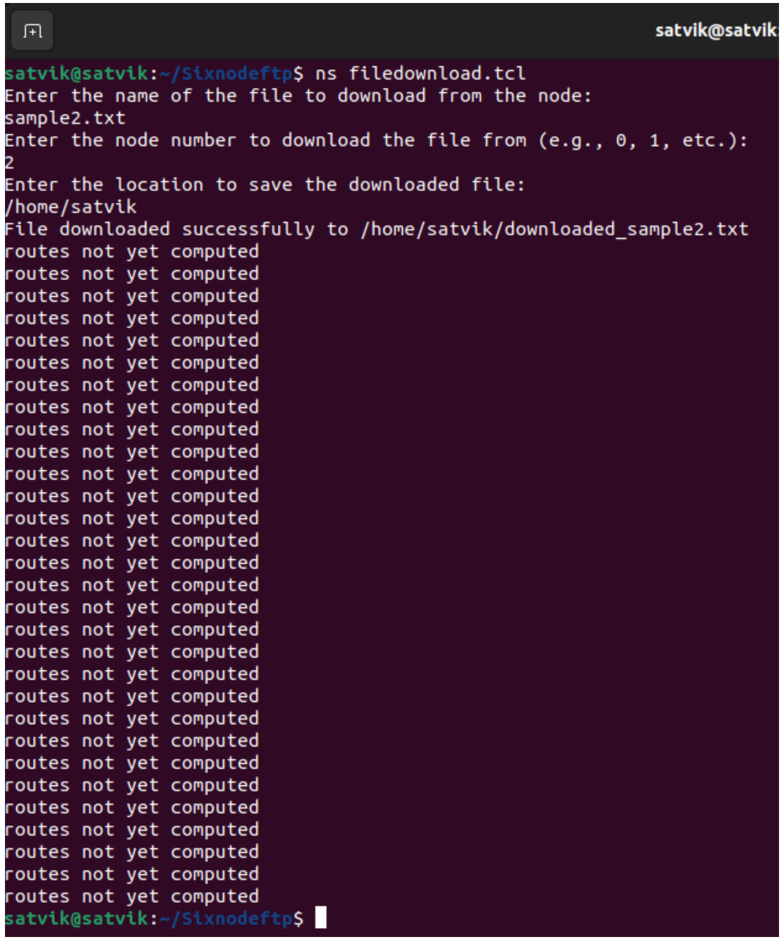
Before Transfer:



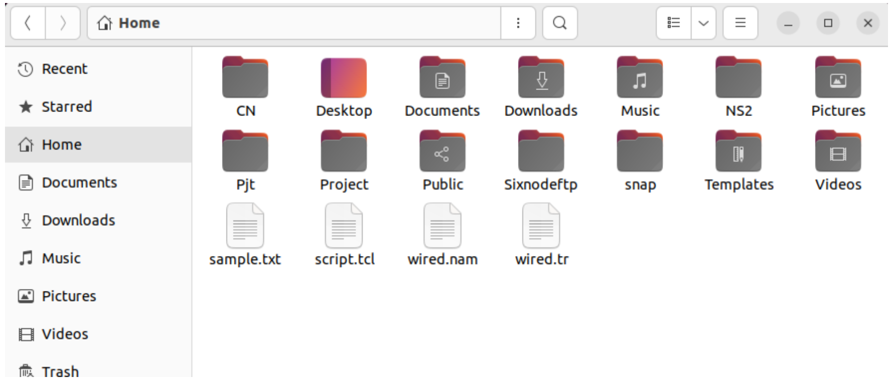
After Transfer:



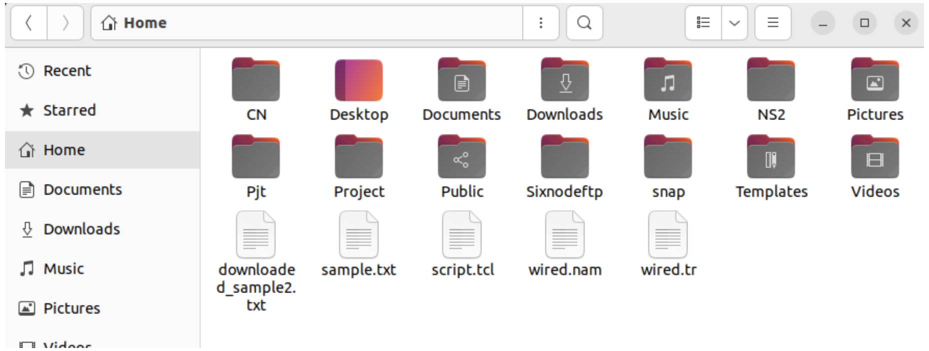
File Download:



Before Transfer:



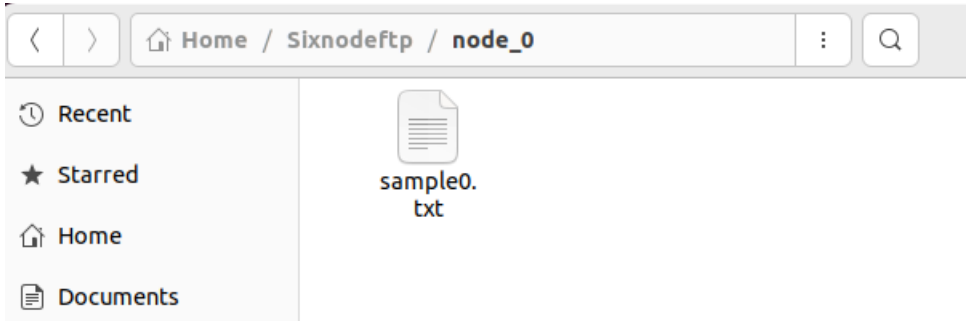
After Transfer:



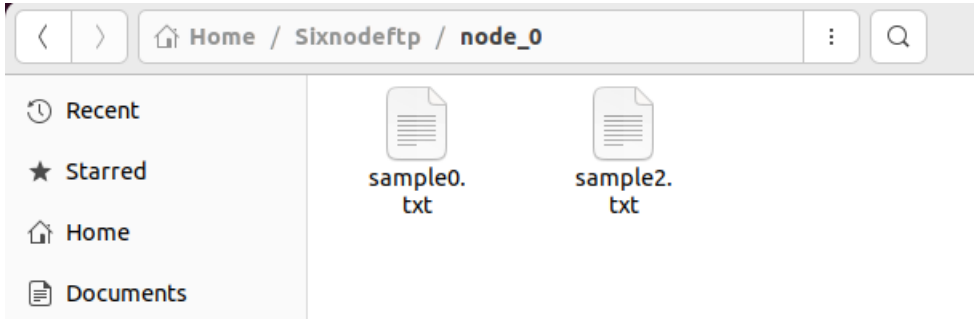
File Transfer:

```
satvik@satvik:~/Sixnodeftp$ ns filetransfer.tcl
Enter source node (0-5):
2
Enter destination node (0-5):
0
Enter file name (e.g., sample.txt):
sample2.txt
File transfered successfully
satvik@satvik:~/Sixnodeftp$
```

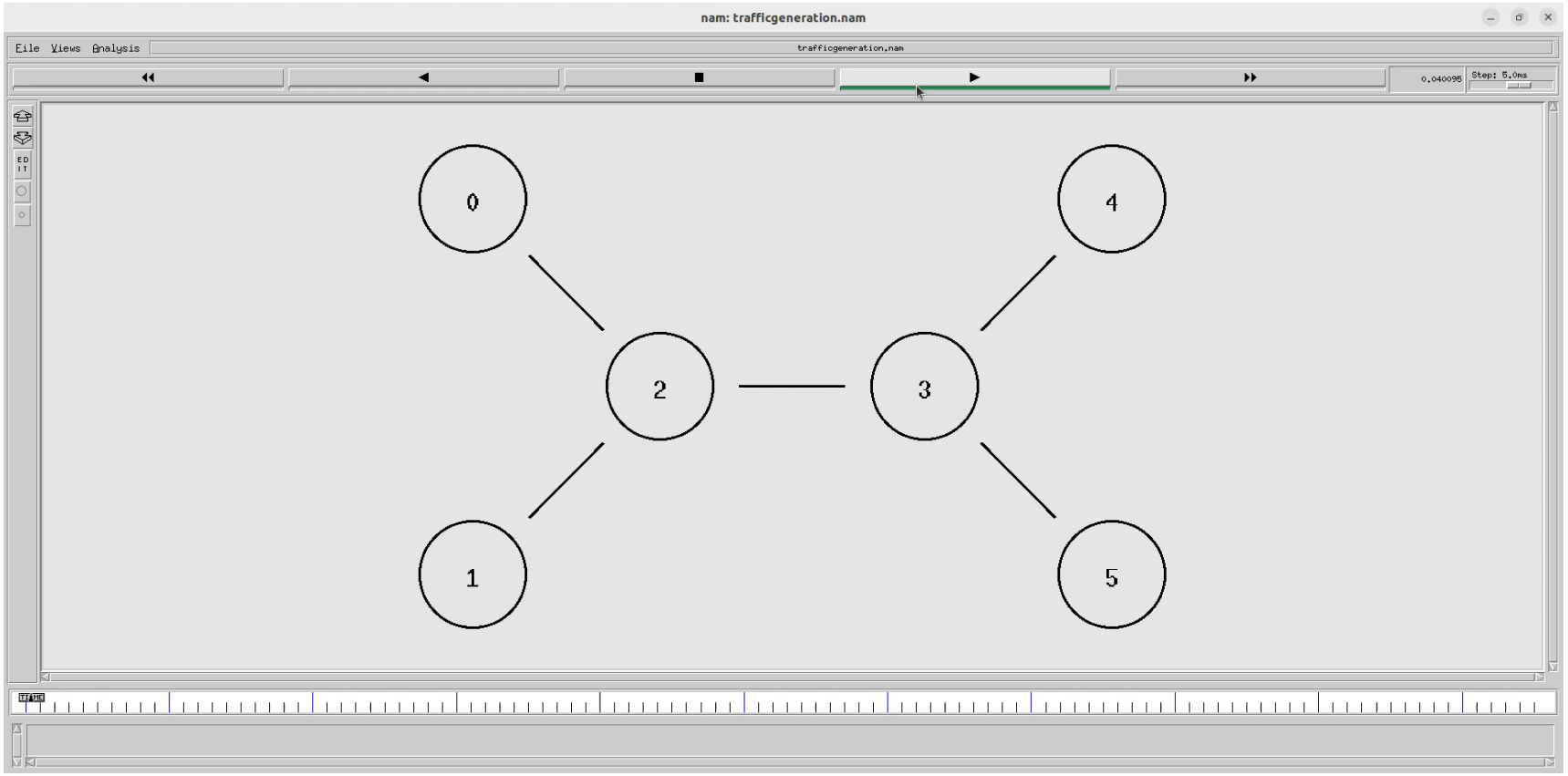
Before Transfer:



After Transfer:



Nam Animation for traffic generation:



event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

r : receive (at to_node)
+ : enqueue (at queue)
- : dequeue (at queue)
d : drop (at queue)

src_addr : node.port (3.0)
dst_addr : node.port (0.0)

```
+ 0.00017 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 0
- 0.00017 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 0
+ 0.007102 2 0 rtProtoDV 6 ------- 0 2.1 0.1 -1 1
- 0.007102 2 0 rtProtoDV 6 ------- 0 2.1 0.1 -1 1
+ 0.007102 2 1 rtProtoDV 6 ------- 0 2.1 1.1 -1 2
- 0.007102 2 1 rtProtoDV 6 ------- 0 2.1 1.1 -1 2
+ 0.007102 2 3 rtProtoDV 6 ------- 0 2.1 3.2 -1 3
- 0.007102 2 3 rtProtoDV 6 ------- 0 2.1 3.2 -1 3
r 0.01017 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 0
r 0.017103 2 0 rtProtoDV 6 ------- 0 2.1 0.1 -1 1
```

Trace file Performance Analysis:

Performance Analysis - File Transfer:

- Throughput is a measure of how many units of information a system can process in a given amount of time.
- Throughput for File transfer:
  - No. of bytes = 26x6 = 156 bytes
  - No. of bits = 156x8 = 1248 bits
  - So, throughput = 1248/2 = 624 bits/sec = 0.624 kbps
- Packet delivery ratio:
  - Dropped packets are denoted by d in the first bit
  - No packets are lost
  - So, PDR is 100%
  - Packet loss is 0%

```
r 0.01017 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 0
r 0.017103 2 0 rtProtoDV 6 ------- 0 2.1 0.1 -1 1
r 0.017103 2 1 rtProtoDV 6 ------- 0 2.1 1.1 -1 2
r 0.017103 2 3 rtProtoDV 6 ------- 0 2.1 3.2 -1 3
r 0.027103 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 4
r 0.027103 1 2 rtProtoDV 6 ------- 0 1.1 2.1 -1 5
r 0.027103 3 2 rtProtoDV 6 ------- 0 3.2 2.1 -1 6
r 0.027103 3 4 rtProtoDV 6 ------- 0 3.2 4.2 -1 7
r 0.027103 3 5 rtProtoDV 6 ------- 0 3.2 5.1 -1 8
r 0.037104 2 0 rtProtoDV 6 ------- 0 2.1 0.1 -1 9
r 0.037104 2 1 rtProtoDV 6 ------- 0 2.1 1.1 -1 10
r 0.037104 2 3 rtProtoDV 6 ------- 0 2.1 3.2 -1 11
r 0.037104 4 3 rtProtoDV 6 ------- 0 4.2 3.2 -1 12
r 0.037104 5 3 rtProtoDV 6 ------- 0 5.1 3.2 -1 13
r 0.047104 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 14
r 0.047104 1 2 rtProtoDV 6 ------- 0 1.1 2.1 -1 15
r 0.054062 3 2 rtProtoDV 6 ------- 0 3.2 2.1 -1 16
r 0.054062 3 4 rtProtoDV 6 ------- 0 3.2 4.2 -1 17
r 0.054062 3 5 rtProtoDV 6 ------- 0 3.2 5.1 -1 18
r 0.230587 4 3 rtProtoDV 6 ------- 0 4.2 3.2 -1 19
r 0.287941 1 2 rtProtoDV 6 ------- 0 1.1 2.1 -1 20
r 0.409578 5 3 rtProtoDV 6 ------- 0 5.1 3.2 -1 21
r 1.935355 0 2 rtProtoDV 6 ------- 0 0.1 2.1 -1 22
r 1.947903 3 2 rtProtoDV 6 ------- 0 3.2 2.1 -1 23
r 1.947903 3 4 rtProtoDV 6 ------- 0 3.2 4.2 -1 24
r 1.947903 3 5 rtProtoDV 6 ------- 0 3.2 5.1 -1 25
```

Performance Analysis – Traffic Generation:

- Throughput for Traffic Generation:
  - Throughput = 166.3 kbps
- Packet delivery ratio:
  - Dropped packets are denoted by d in the first bit
  - No packets are lost
  - So, PDR is 100%
  - Packet loss is 0%

```
r 1.010003 0 2 tcp 40 ------- 0 0.0 5.0 0 0
r 1.010003 4 3 tcp 40 ------- 0 4.0 1.0 0 1
r 1.010004 2 3 cbr 48 ------- 0 2.0 5.1 0 2
r 1.020006 2 3 tcp 40 ------- 0 0.0 5.0 0 0
r 1.020006 3 2 tcp 40 ------- 0 4.0 1.0 0 1
```

Conclusion:

- The scripts demonstrate the setup of simulation parameters, creation of nodes, establishment of links, definition of agents, and configuration of applications for FTP and traffic generation.
- Through these simulations, we observed the behavior of FTP protocols in file upload, download, and transfer mechanisms, as well as the impact of traffic generated by the Constant Bit Rate (CBR) application on network performance.
- NS2's visualization features, such as trace and Nam files, provided insights into network behavior and topology during the simulations, aiding in analysis and understanding.
- Key components such as the TCP Sliding Window Algorithm and FTP File Transfer Mechanism are highlighted for their pivotal roles in ensuring effective data transmission between client and server nodes.

Thankyou