Finolex Academy of Management and Technology, Ratnagiri

Department of Information Technology

Subject:	**Networking Lab (ITL401)**		
Class:	**SE IT / Semester – IV (CBCGS) / Academic year: 2017-18**		
Name of Student:	**Kazi Jawwad A Rahim**		
Roll No:	**28**	**Date of performance (DOP) :**	
Experiment No:	**07**	**Date of checking (DOC) :**	
Title: Graphical simulation of link failure in wired network using NS2.			
Marks:		**Teacher's Signature:**	

1. Aim: **Graphical simulation of link failure in wired network using NS2.**

2. Prerequisites:

Knowledge of

1. Defining the network nodes, links, queues and topology as well.
2. Defining the agents and their applications
3. Network Layers and protocols

3. Hardware Requirements:

1. PC with minimum 2GB RAM

4. Software Requirements:

1. Linux (Ubuntu 10.04)
2. ns-2.34 package
3. Text editor

5. Learning Objectives:

1. To demonstrate the topology with link failure.
2. Defining the different agents and their applications like TCP, FTP over TCP, UDP, CBR and CBR over UDP etc.
3. To observe a DV routing protocol in action.

6. Course Objectives Applicable: LO 3

7. Program Outcomes Applicable: PO2, PO4

8. Program Education Objectives Applicable: 1

9. Theory:

The link failure in the wired network leads to intolerable performance degradation. Therefore, it becomes necessary to use an efficient routing protocol that automatically recovers the link from the link failure. Distance Vector (DV) routing protocol automatically recovers the link failure by using alternative path for data transmission.

In this program the link between any two nodes can be failed using the command "rtmodel-at" along with keyword "down" with the time specification and the DV protocol recovers the link failure using dynamic route discovery mechanism.

The following code creates seven nodes and stores them in the array n().

```
for {set i 0} {$i < 7} {incr i} {
  set n($i) [$ns node]}
```

To connect the nodes to create a circular topology.

```
for {set i 0} {$i < 7} {incr i} {
  $ns duplex-link $n($i) $n([expr ($i+1)%7]) 1Mb 10ms DropTail
}
```

This 'for' loop connects all nodes with the next node in the array with the exception of the last node, which is being connected with the first node. To accomplish that, used the '%' (modulo) operator.

$ns rtmodel-at 1.0down $n1$n2

The code given above will fail the link between **Node n1** and **Node n2** at time **1.0**.

$ns rtmodel-at 2.0 up $n1$n2

The code given above will reconnect the link between **Node n1** and **Node n2** at time **2.0**.

Steps to create and execute tcl script:

Step 1: Open any text editor (vi, nano).

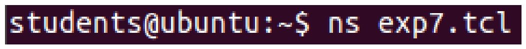
Step 2: Write the program using ns2 tcl script and save with extension as "filename.tcl"

Step 3: Execute tcl script as "ns filename.tcl"

Step 4: Press the "play button" and Observe the output.

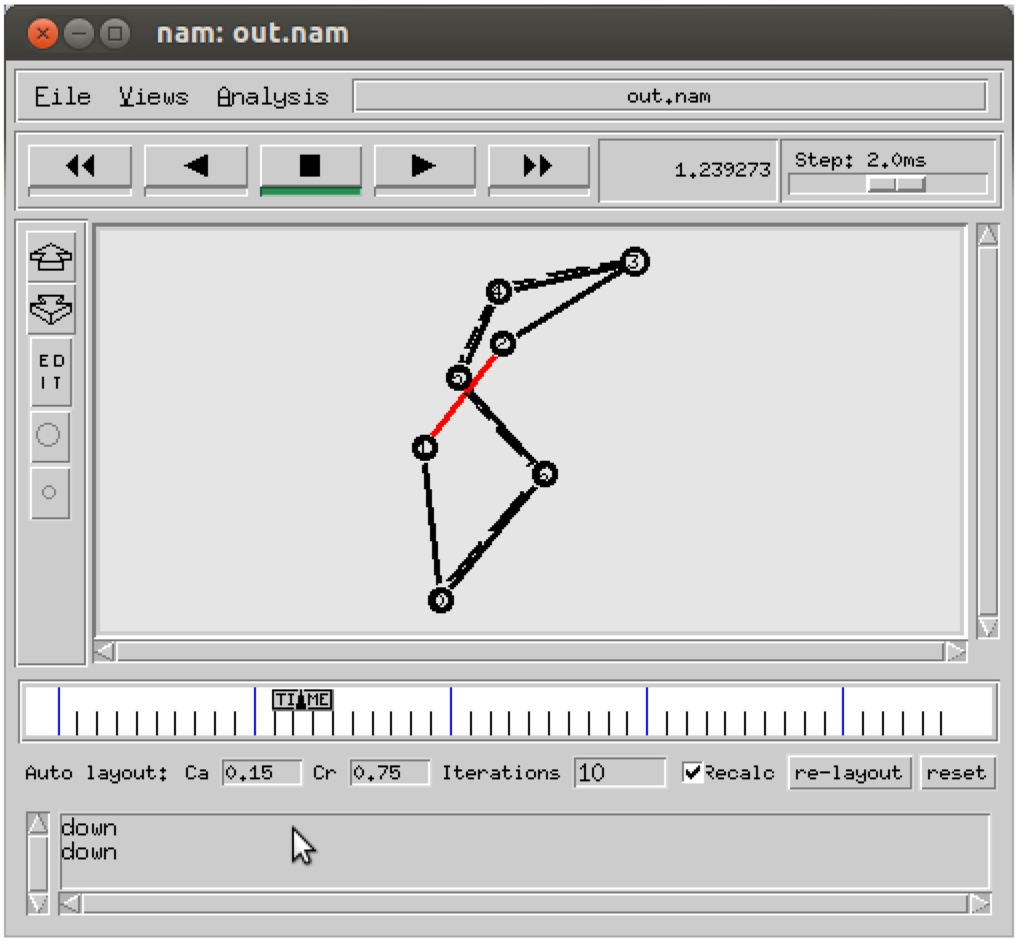
Source Code:- [A]

```
#Create a simulator object

set ns [new Simulator]

#Define different colours for data flows (for NAM)

$ns color 1 Blue

#Tell the simulator to use dynamic routing

$ns rtproto DV

#Open the nam trace file

set nf [open out.nam w]

$ns namtrace-all $nf

#Define a 'finish' procedure

proc finish {} {

global ns nf

$ns flush-trace

#Close the trace file

close $nf

#Execute nam on the trace file

exec nam out.nam &

exit 0

}

#Create seven nodes

for {set i 0} {$i < 7} {incr i} {

set n($i) [$ns node]

}

#Create links between the nodes

for {set i 0} {$i < 7} {incr i} {

$ns duplex-link $n($i) $n([expr ($i+1)%7]) 1Mb 10ms DropTail
}


#Create a UDP agent and attach it to node n(0)

set udp0 [new Agent/UDP]

$ns attach-agent $n(0) $udp0

# Create a CBR traffic source and attach it to udp0

set cbr0 [new Application/Traffic/CBR]

$cbr0 set packetSize_ 500

$cbr0 set interval_ 0.005

$cbr0 attach-agent $udp0

#Create a Null agent (a traffic sink) and attach it to node n(3)

set null0 [new Agent/Null]

$ns attach-agent $n(3) $null0

#Connect the traffic source with the traffic sink

$ns connect $udp0 $null0

#Schedule events for the CBR agent and the network dynamics

$ns at 0.5 "$cbr0 start"

$ns rtmodel-at 1.0 down $n(1) $n(2)

$ns rtmodel-at 2.0 up $n(1) $n(2)

$ns at 4.5 "$cbr0 stop"

#Call the finish procedure after 5 seconds of simulation time

$ns at 5.0 "finish"

#Run the simulation

$ns run
```



```
students@ubuntu:~$ ns exp7.tcl
```

NAM file:-


nam: out.nam
File Views Analysis
out.nam
1.239273
Step: 2.0ms
EDIT
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc re-layout reset

```
down
down
```

Source Code:- [B]

p2pnetwork2.tcl

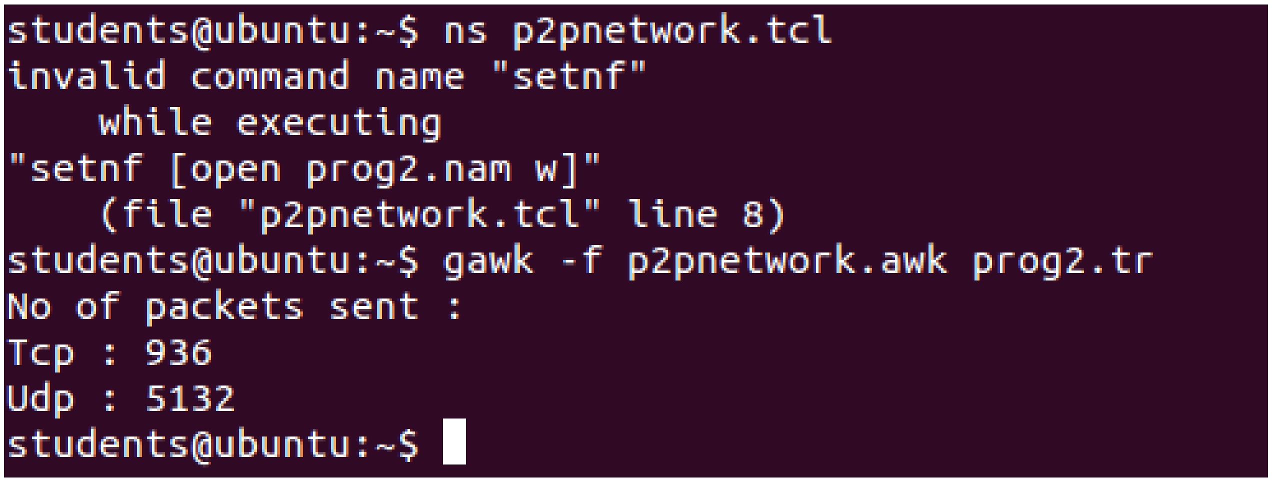
```
#Create simulator
set ns [new Simulator]

#Define different colors for data flows
$ns color 1 purple

#Open the nam trace file
setnf [open prog2.nam w]
$ns namtrace-all $nf
setnd [open prog2.tr w]
$ns trace-all $nd

#Define a 'finish' procedure
proc finish {} {
global ns nfnd
$ns flush-trace
close $nf
execnam prog2.nam &
exit 0
}

#create nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

$n0 shape circle
$n0 color red
$n1 shape circle
$n1 color red
$n2 shape square
$n2 color blue
$n3 shape circle
$n3 colordarkgreen

$n0 label TCP
$n1 label UDP
$n3 label NULL-TCPSINK

#create link
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 10ms DropTail

#setup tcp connection
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n0 $tcp0
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
$ns connect $tcp0 $sink0

set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0

set udp0 [new Agent/UDP]
$udp0 set class_ 1
$ns attach-agent $n1 $udp0
set null0 [new Agent/Null]
$ns attach-agent $n3 $null0
$ns connect $udp0 $null0

set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

```
#Schedule events for the CBR agents
$ns at 0.2 "$cbr0 start"
$ns at 0.1 "$ftp0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 4.4 "$ftp0 stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Run the simulation
$ns run
```

p2pnetwork2.awk

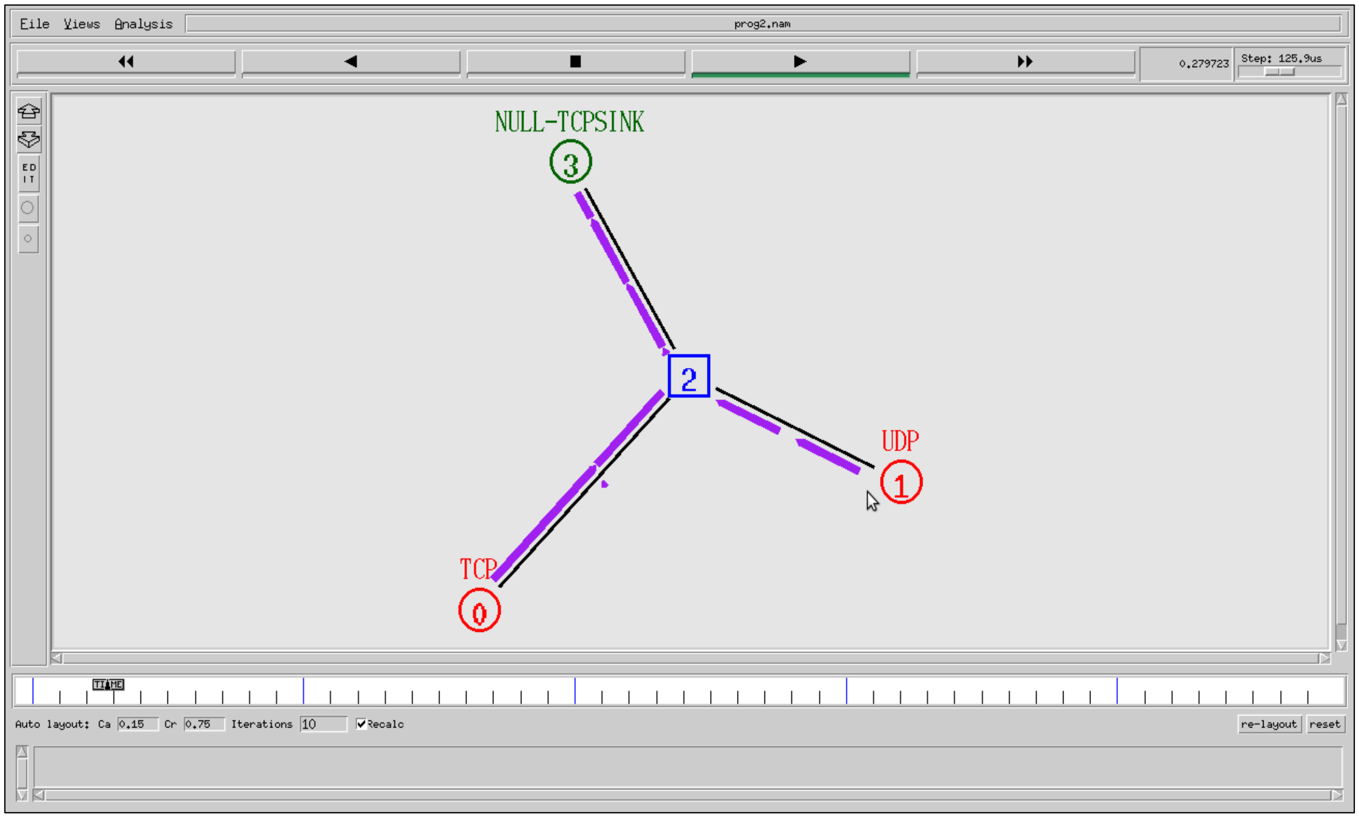
```
BEGIN {
ctcp=0;
cudp=0;
}
{
pkt=$5;
if(pkt=="cbr") { cudp++;}
if(pkt=="tcp") { ctcp++;}
}
END {
printf("No of packets sent : \nTcp : %d\nUdp : %d\n",ctcp,cudp);
}
```

Output -:



```
students@ubuntu:~$ ns p2pnetwork.tcl
invalid command name "setnf"
    while executing
"setnf [open prog2.nam w]"
    (file "p2pnetwork.tcl" line 8)
students@ubuntu:~$ gawk -f p2pnetwork.awk prog2.tr
No of packets sent :
Tcp : 936
Udp : 5132
students@ubuntu:~$ 
```

nam file –



```
prog2.nam
0.279723   Step: 125.9us
Auto layout: Ca 0.15  Cr 0.75  Iterations 10  ✓Recalc
```

13. Experiment/Assignment Evaluation

SR	Parameters	Weight	Excellent	Good	Average	Poor	Not as per requirement
		Scale Factor ->	5	4	3	2	0
1	Technical Understanding	25					
2	Performance / Execution	25					
3	Question Answers	20					
4	Punctuality	20					
5	Presentation	10					
	Total out of 100 --> #(to be converted as per term-work evaluation applicable to the subject)		**∑ (Weight * Scale Factor)/5 = ________**				

References:

1. https://www.isi.edu/nsnam/ns/tutorial/
2. Wireless Networks: by Nicopolitidia, M S Obaidat, GI Papadimitriou; Wiley India (student edition2010)
3. Wireless communications: by T L Singal; Tata McGraw Hill Education private Ltd.(edition 2011)

Viva Questions

1. Explain Structure of Trace files
2. How can we give positions to the nodes in NAM?
3. In tcl script how to schedule the events?
4. Explain Constant Bit Rate(CBR) and TCP.
5. Explain FTP over TCP.