Wireless Networking and Systems

ns2 tutorial

ns2- Network Simulator

- One of the most popular simulator among networking researchers.
- Discrete event, Packet level simulator
  - Events like ‘received an *ack* packet’, ‘enqueued a data packet’
- Network protocol stack written in C++
- Tcl (Tool Command Language) used for specifying scenarios and events.
- Unix Based. Runs also in windows using *cygwin*
- Simulates both wired and wireless networks.

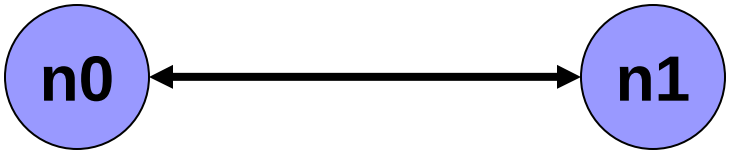
Goal of this tutorial

- Understand how to write Tcl scripts to simulate simple network topologies and traffic patterns.
- Analyze the trace files and understand the performance of the protocols.

Overview

- Wired
  - Creating a simple two node network topology
  - Adding traffic to the link
  - UDP traffic
  - TCP traffic
  - Fault injection
- Wireless
  - TCP performance on a linear chain of n nodes

Simple two node wired network



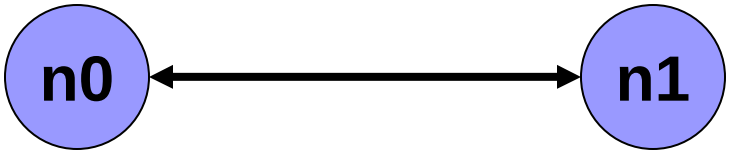
Step 1:

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

Step 2:

```
#Open trace files
set f [open out.tr w]
$ns trace-all $f
```

Simple two node wired network



Step 3:

```
#Create two nodes
set n0 [$ns node]
set n1 [$ns node]
```

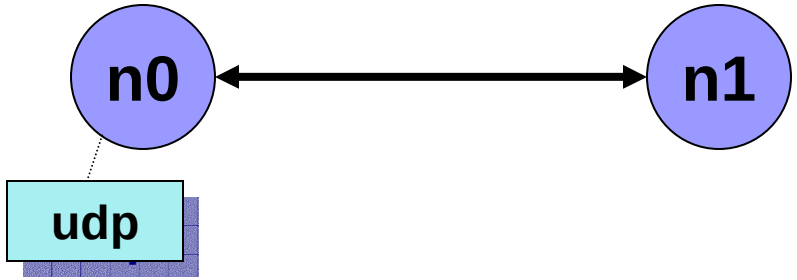
Step 4:

```
#Create a duplex link between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

Simple two node wired network

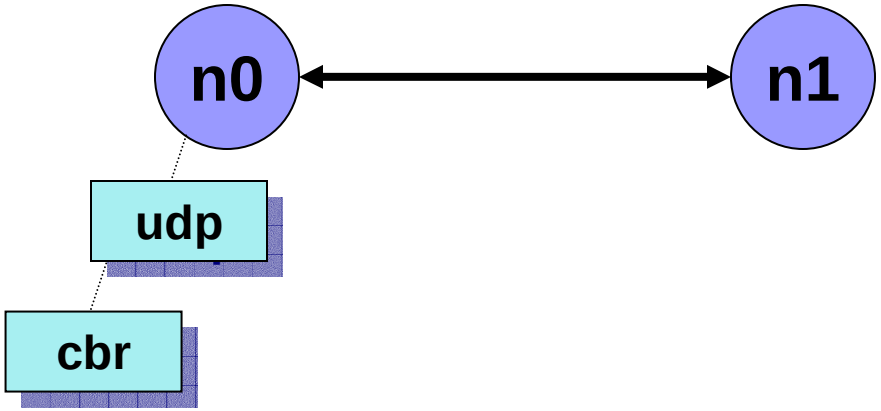
```
#Create a simulator object
set ns [new Simulator]
#Open trace files
set f [open out.tr w]
$ns trace-all $f
#Define a 'finish' procedure
proc finish {} {
        global ns
        $ns flush-trace
        exit 0
}
#Create two nodes
set n0 [$ns node]
set n1 [$ns node]
#Create a duplex link between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

Adding traffic to the link



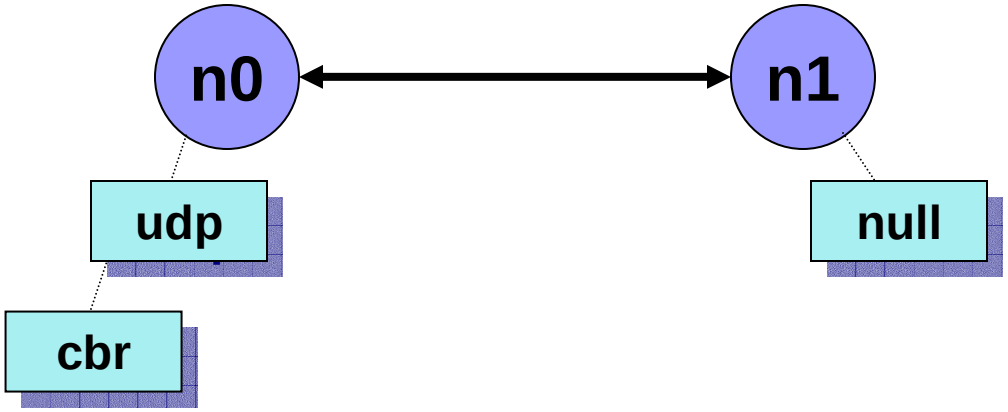

```
#Create a UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

Adding traffic to the link



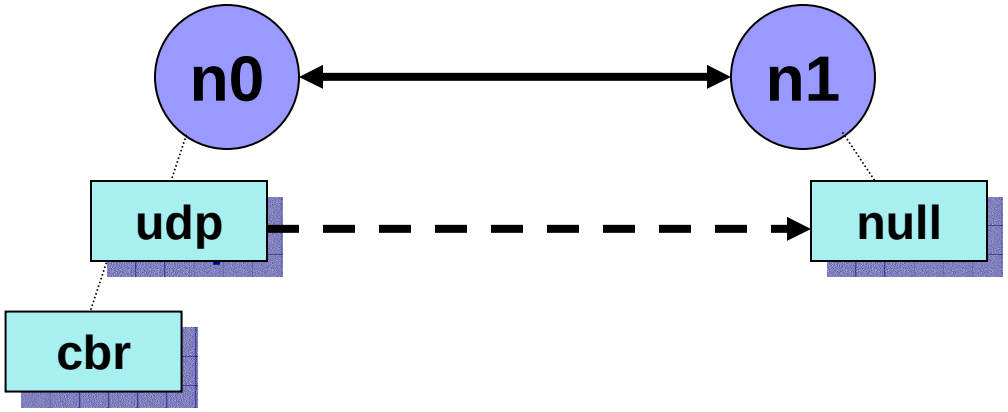

```
# Create a CBR traffic source and attach it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

Adding traffic to the link



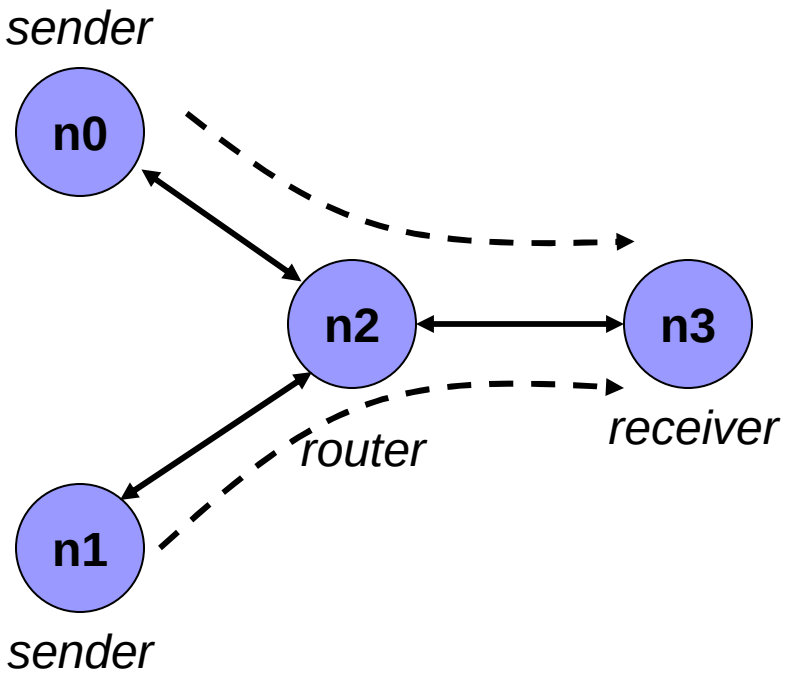

```
#Create a Null agent (a traffic sink) and
attach it to node n1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

Adding traffic to the link



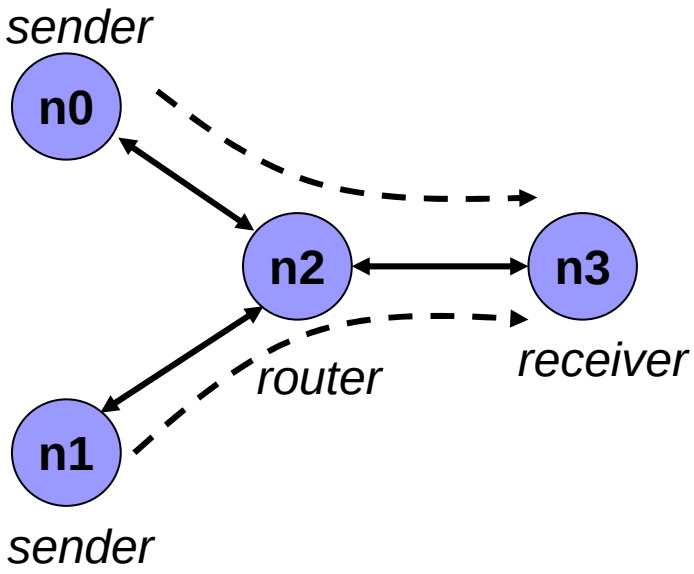

```
#Connect the traffic source with the traffic sink
$ns connect $udp0 $null0
#Schedule events for the CBR agent
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop“
```

Simulate a simple topology – UDP Traffic




```
#Create a simulator object
set ns [new Simulator]
#Open trace files
set f [open out.tr w]
$ns trace-all $f
#Define a 'finish' procedure
proc finish {} {
    global ns
    $ns flush-trace
    exit 0
}
#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

Simulate a simple topology – UDP Traffic



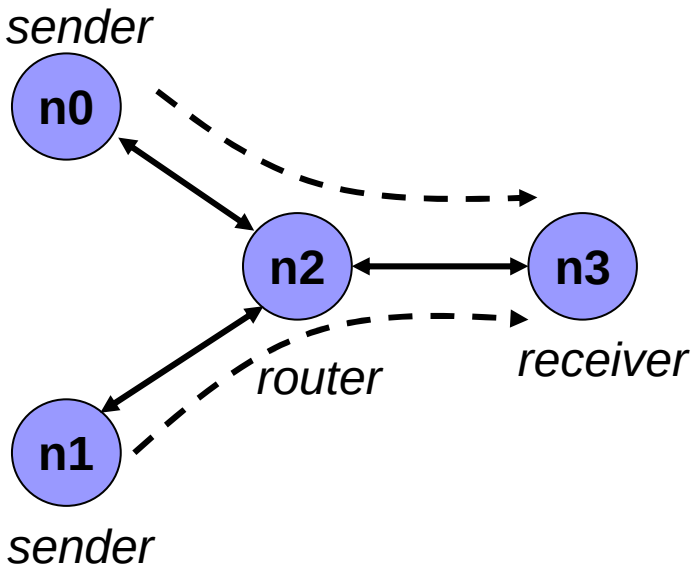
#Create links between the nodes

$ns duplex-link $n0 $n2 1Mb 10ms DropTail

$ns duplex-link $n1 $n2 1Mb 10ms DropTail

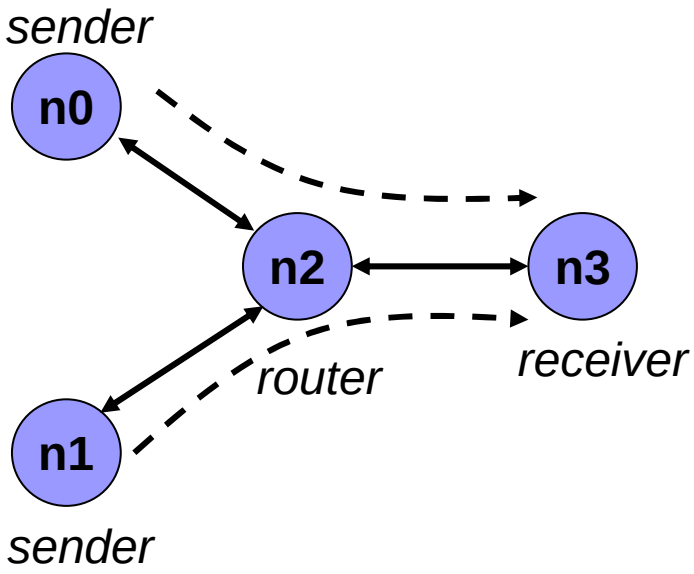
$ns duplex-link $n3 $n2 1Mb 10ms SFQ

Simulate a simple topology – UDP Traffic



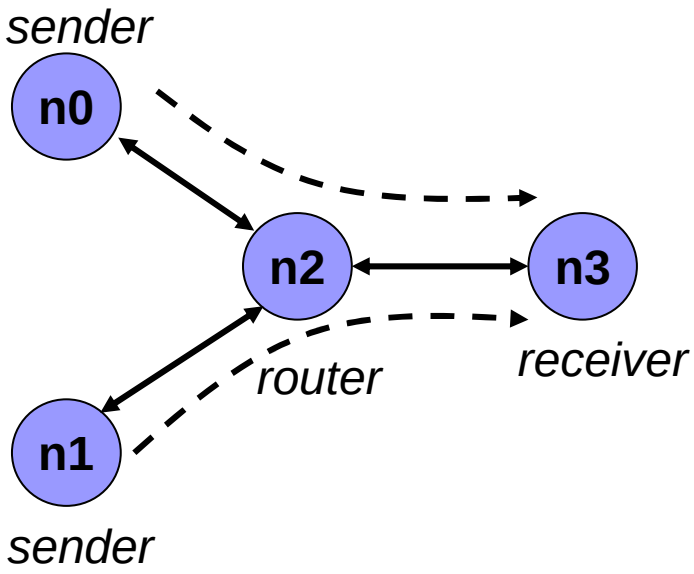

```
#Create a UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$udp0 set class_ 1
$ns attach-agent $n0 $udp0
```

Simulate a simple topology – UDP Traffic



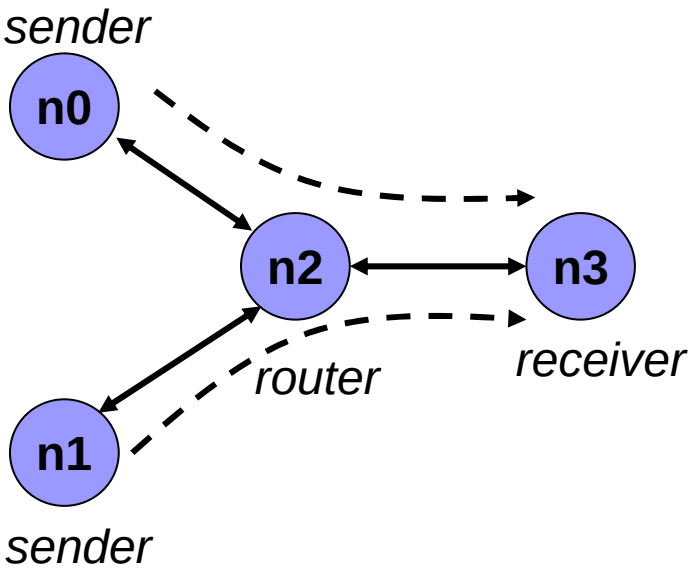

```
# Create a CBR traffic source and attach it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

Simulate a simple topology – UDP Traffic



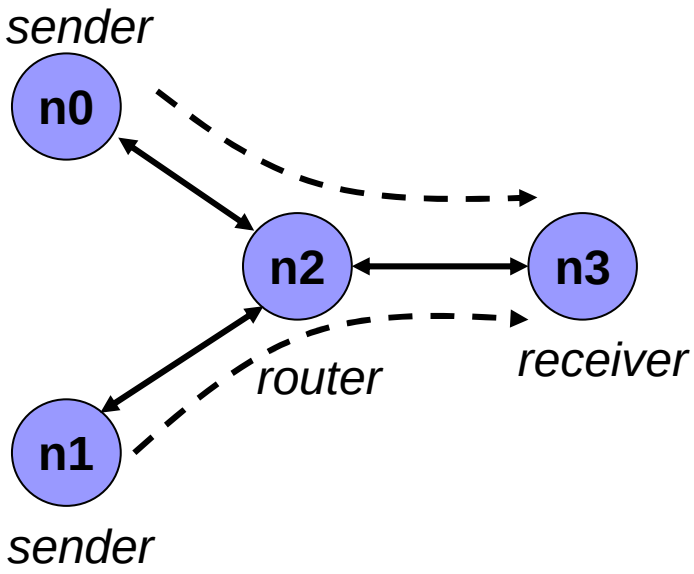

```
#Create a UDP agent and attach it to node n1
set udp1 [new Agent/UDP]
$udp1 set class_ 2
$ns attach-agent $n1 $udp1
```

Simulate a simple topology – UDP Traffic




```
# Create a CBR traffic source and attach it to udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 500
$cbr1 set interval_ 0.005
$cbr1 attach-agent $udp1
```

Simulate a simple topology – UDP Traffic

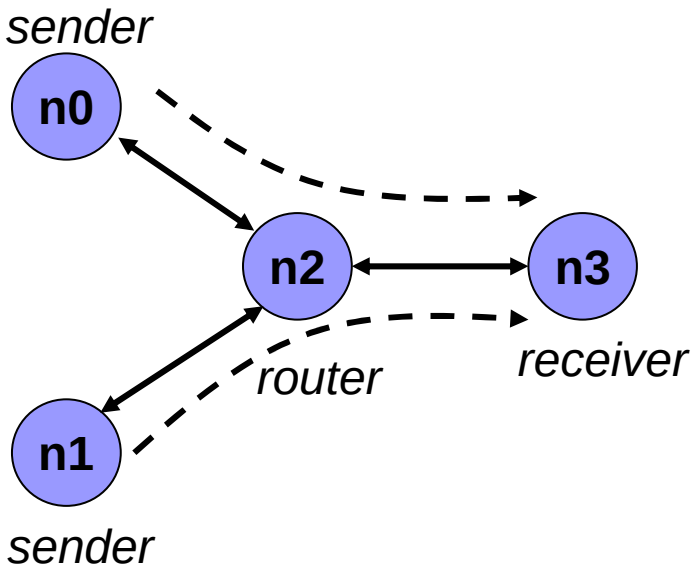


#Create a Null agent (a traffic sink) and attach it to node n3

set null0 [new Agent/Null]

$ns attach-agent $n3 $null0

Simulate a simple topology – UDP Traffic



#Connect the traffic sources with the traffic sink

$ns connect $udp0 $null0

$ns connect $udp1 $null0

Simulate a simple topology – UDP Traffic

```
#Schedule events for the CBR agents
$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$cbr1 start"
$ns at 4.0 "$cbr1 stop"
$ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 5 seconds of
   simulation time
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

Trace Analysis

http://nsnam.isi.edu/nsnam/index.php/NS-2_Trace_Formats

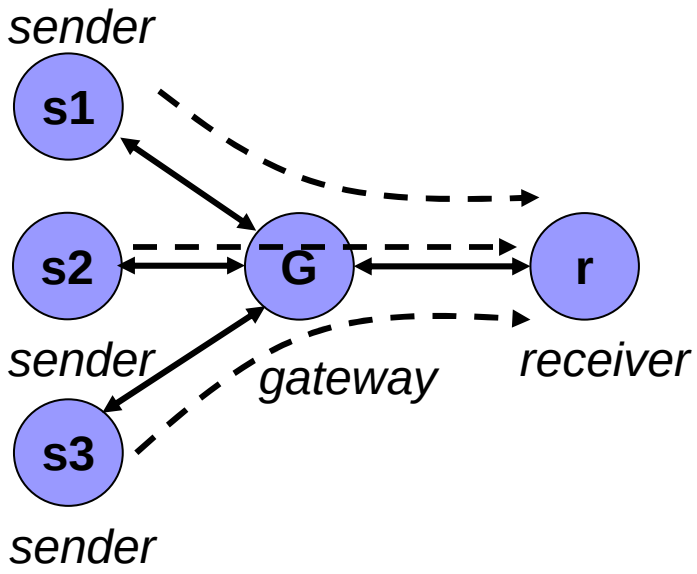
event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

```
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)
```

```
r 1.3556 3 2 ack 40 ------- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 ------- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 ------- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 ------- 2 1.0 3.1 157 207
```

TCP Traffic



- 0, 1, 2 are senders
- 3 is a Gateway
- 4 receiver

TCP Traffic

- #Create a TCP agent and attach it to node s1
- set tcp1 [new Agent/TCP/Reno]
- $ns attach-agent $s1 $tcp1
- $tcp1 set window_ 8
- $tcp1 set fid_ 1

TCP Traffic

- #Create a TCP agent and attach it to node s2
- set tcp2 [new Agent/TCP/Reno]
- $ns attach-agent $s2 $tcp2
- $tcp2 set window_ 8
- $tcp2 set fid_ 2

- #Create a TCP agent and attach it to node s3
- set tcp3 [new Agent/TCP/Reno]
- $ns attach-agent $s3 $tcp3
- $tcp3 set window_ 4
- $tcp3 set fid_ 3

TCP Traffic

- #Create TCP sink agents and attach them to node r
- set sink1 [new Agent/TCPSink]
- set sink2 [new Agent/TCPSink]
- set sink3 [new Agent/TCPSink]

- $ns attach-agent $r $sink1
- $ns attach-agent $r $sink2
- $ns attach-agent $r $sink3

TCP Traffic

- #Connect the traffic sources with the traffic sinks
- $ns connect $tcp1 $sink1
- $ns connect $tcp2 $sink2
- $ns connect $tcp3 $sink3

TCP Traffic

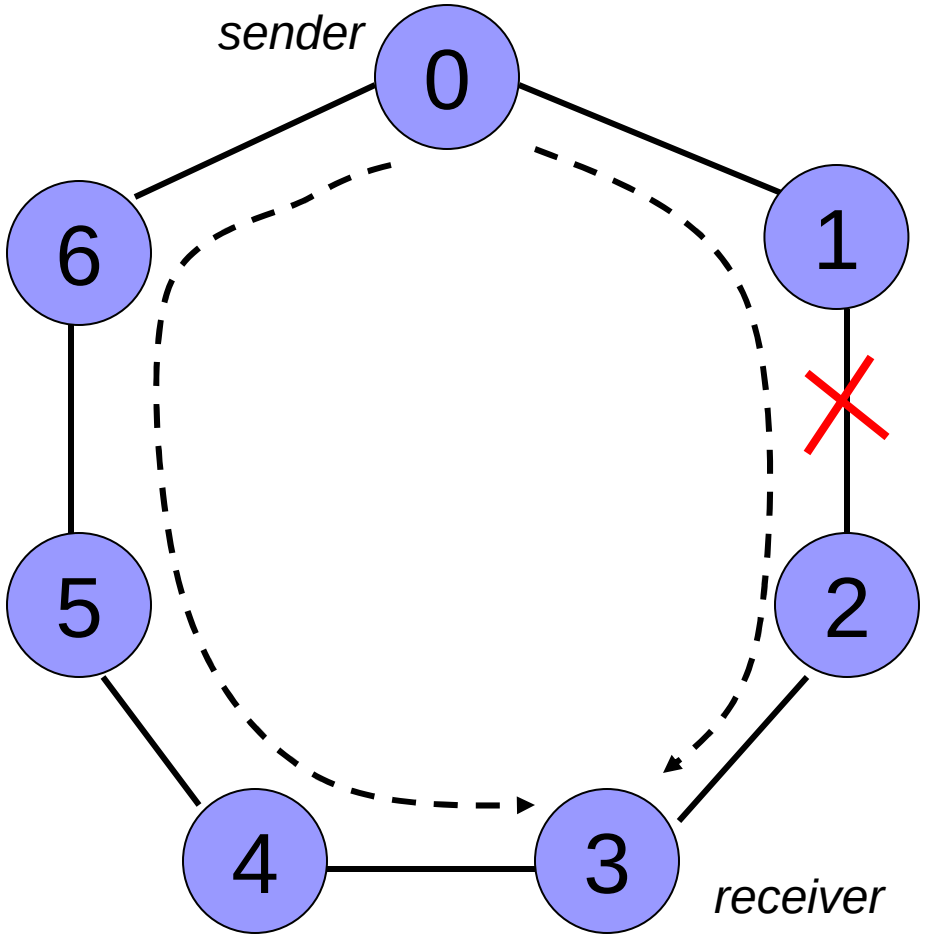
- #Create FTP applications and attach them to agents
- set ftp1 [new Application/FTP]
- $ftp1 attach-agent $tcp1
- set ftp2 [new Application/FTP]
- $ftp2 attach-agent $tcp2
- set ftp3 [new Application/FTP]
- $ftp3 attach-agent $tcp3

TCP Traffic

```
#Define a 'finish' procedure
proc finish {} {
        global ns
        $ns flush-trace
        exit 0
}

$ns at 0.1 "$ftp1 start"
$ns at 0.1 "$ftp2 start"
$ns at 0.1 "$ftp3 start"
$ns at 5.0 "$ftp1 stop"
$ns at 5.0 "$ftp2 stop"
$ns at 5.0 "$ftp3 stop"
$ns at 5.25 "finish"
$ns run
```

Complex topology and link failure



Complex topology and link failure

```
#Create a simulator object
set ns [new Simulator]
#Tell the simulator to use dynamic routing
$ns rtproto DV
#Define a 'finish' procedure
proc finish {} {
        global ns
        $ns flush-trace
        exit 0
}
```

Complex topology and link failure

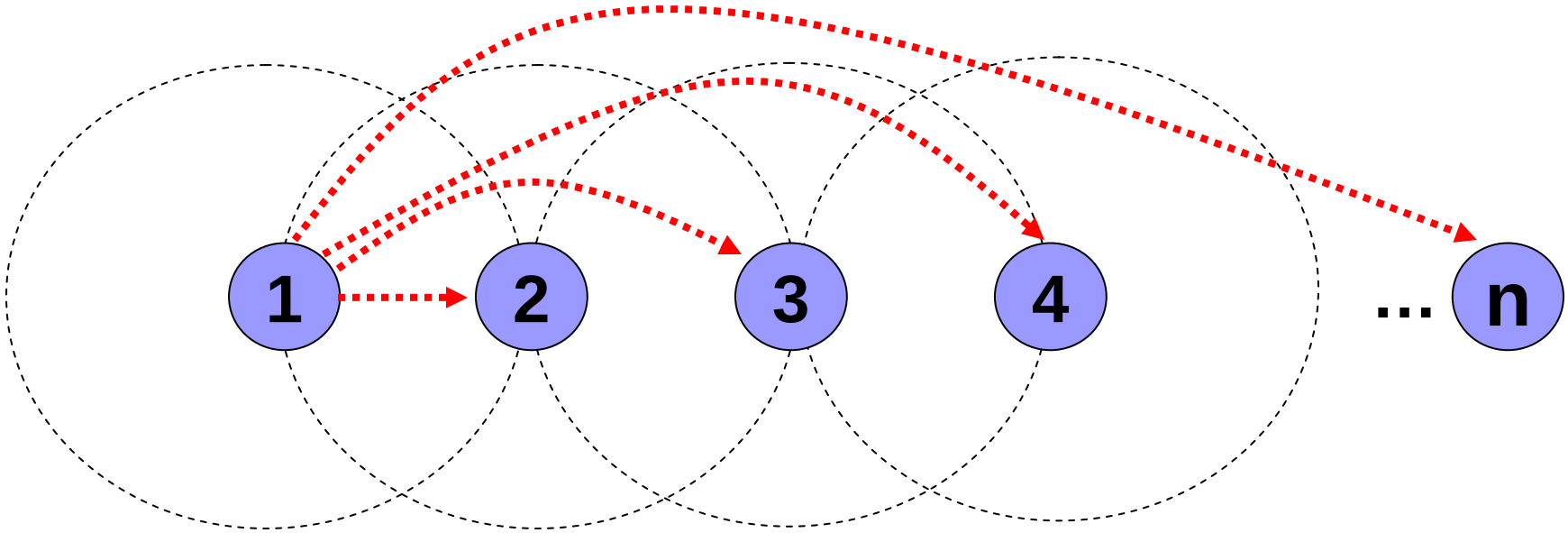
```
#Create seven nodes
for {set i 0} {$i < 7} {incr i} {
set n($i) [$ns node]
}
#Create links between the nodes
for {set i 0} {$i < 7} {incr i} {
$ns duplex-link $n($i) $n([expr ($i+1)%7]) 1Mb
    10ms DropTail
}
```

Complex topology and link failure

- #Create a UDP agent and attach it to node n(0)
- # Create a CBR traffic source and attach it to udp0
- #Create a Null agent (a traffic sink) and attach it to node n(3)
- #Connect the traffic source with the traffic sink

```
#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
```

Wireless Linear Topology



Run Wireless TCP for each connection separately and look how the TCP congestion window changes.

- Refer:

  http://www.cs.sunysb.edu/~samir/cse590/ns-simulator.htm