فصل پنجم

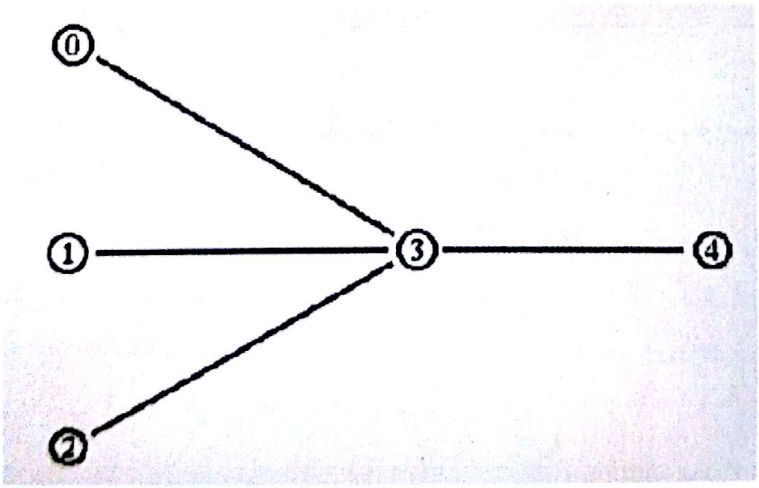
ایجاد فایل‌های خروجی برای Xgraph

۵-۱ مقدمه

یکی از قسمت‌های ns-allinone پکیج Xgraph است که یک برنامه ترسیم‌کننده برای ایجاد گراف‌های حاصل از شبیه‌سازی استفاده می‌شود. در این بخش ما نشان خواهیم داد که چگونه می‌توانیم فایل‌های output برای اسکریپت‌های tcl ایجاد کنیم و سپس مجموعه داده‌های آن‌ها را برای Xgraph استفاده کنیم. به عنوان مثال می‌خواهیم نحوه استفاده از ایجادکننده‌های ترافیک را نیز نشان دهیم. نکته‌ای که باید توجه کرد این است که روش‌های مختلفی برای ایجاد فایل‌های خروجی مناسب برای Xgraph وجود دارد.

۵-۲ توپولوژی و منبع‌های ترافیک

قبل از همه ما توپولوژی زیر را باید ایجاد کنیم.



کدهای زیر شبیه به همان کدهایی هستند که در قسمت‌های قبل آن‌ها را بررسی کرده‌ایم.

```
set n0 [$ns node]
```

```
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
$ns duplex-link $n0 $n3 1Mb 100ms DropTail
$ns duplex-link $n1 $n3 1Mb 100ms DropTail
$ns duplex-link $n2 $n3 1Mb 100ms DropTail
$ns duplex-link $n3 $n4 1Mb 100ms DropTail
```

ما قصد داریم که ترافیک‌های بین منبع‌های n0، n1 و n2 را ضبط کنیم اما ابتدا باید یک روال بنویسیم تا اضافه کردن ترافیک ایجادکننده‌ها و منبع‌ها را ساده‌تر کند.

```
  proc attach-expoo-traffic { node sink size burst idle rate } {
#Get an instance of the simulator
set ns [Simulator instance]
#Create a UDP agent and attach it to the node
set source [new Agent/UDP]
$ns attach-agent $node $source
#Create an Expoo traffic agent and set its configuration parameters
set traffic [new Application/Traffic/Exponential]
$traffic set packetSize_ $size
$traffic set burst_time_ $burst
$traffic set idle_time_ $idle
$traffic set rate_ $rate
# Attach traffic source to the traffic generator
$traffic attach-agent $source
#Connect the source and the sink
$ns connect $source $sink
return $traffic
}
```

ظاهر این روال پیچیده‌تر از باطن آن است. این روال ۶ پارامتر می‌گیرد: یک گره – یک sink ترافیک (گیرنده ترافیک یا فرو برنده ترافیک) که قبلاً ایجاد شده- اندازه پاکت برای منبع ترافیک- زمان‌های brust و idle (برای توزیع نمای) و نرخ اوج (peak rate). در روال بالا ابتدا یک منبع ترافیک را ایجاد می‌کنیم و بعد آن را به یک گره ضمیمه و پیوست می‌کنیم. سپس یک شئ Traffic/Expoo ایجاد می‌کنیم و پارامترهای پیکره بنده را تنظیم کرده و پیش از اینکه واقعاً به هم متصل شوند به منبع ترافیک ضمیمه می‌کنیم.

در آخر روال یک دستگیر یا handler (یا اداره‌کننده) به منبع ترافیک برخواهد گرداند (فرستنده ترافیک).

این روال یک مثال خوب از نحوه انجام دوباره وظیفه‌ها است (مثل ضمیمه کردن ترافیک به یک منبع ترافیک) که چندین گره می‌توانند آن را دستگیر کنند. حال ما از روال برای تخصیص منبع‌های ایجادکننده ترافیک با نرخ peakهای مجزا به گره‌های n0 و n1 و n3 استفاده می‌کنیم و سپس ترافیک ایجاد شده توسط آن‌ها را به گره n4 ارسال می‌کنیم (sink می‌کنیم که در ابتدا ایجاد کرده‌ایم).

```
set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
set sink2 [new Agent/LossMonitor]
$ns attach-agent $n4 $sink0
$ns attach-agent $n4 $sink1
$ns attach-agent $n4 $sink2
```

```
set source0 [attach-expoo-traffic $n0 $sink0 200 2s 1s 100k]
set source1 [attach-expoo-traffic $n1 $sink1 200 2s 1s 200k]
set source2 [attach-expoo-traffic $n2 $sink2 200 2s 1s 300k]
```

در این مثال ما از اشیاء Agent/ossMonitor به عنوان sinkهای ترافیک استفاده کرده‌ایم. از این‌رو آن‌ها مقداری از بایت‌های رسیده را ذخیره می‌کنند که می‌توانند برای محاسبه پهنای باند استفاده شوند.

۵-۳ ضبط داده‌ها در فایل‌های خروجی

هم اکنون ما سه فایل خروجی را باز می‌کنیم. خطوط زیر را در اسکریپت‌های قبلی به‌کار برده‌ایم.

```
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]
```

این فایل‌ها در یک نقطه بسته می‌شوند. ما در روال "finish" شبیه‌سازی این کار را انجام داده‌ایم.

```
proc finish {} {
global f0 f1 f2
#Close the output files
close $f0
close $f1
close $f2
#Call xgraph to display the results
exec xgraph out0.tr out1.tr out2.tr -geometry 800x400 &
exit 0
}
```

در روال "finish" فقط فایل‌ها را نبسته‌ایم، بلکه Xgraph را نیز برای نمایش نتایج فراخوانی کرده ایم. ممکن است که شما قصد داشته باشید که اندازه پنجره Xgraph به اندازه ۸۰۰×۴۰۰ باشد.

حال می‌توانیم روالی را بنویسیم که واقعاً داده‌ها را در فایل‌های خروجی می‌نویسد.

```
proc record {} {
global sink0 sink1 sink2 f0 f1 f2
#Get an instance of the simulator
set ns [Simulator instance]
#Set the time after which the procedure should be called again
set time 0.5
#How many bytes have been received by the traffic sinks?
set bw0 [$sink0 set bytes_]
set bw1 [$sink1 set bytes_]
set bw2 [$sink2 set bytes_]
#Get the current time
set now [$ns now]
#Calculate the bandwidth (in MBit/s) and write it to the files
puts $f0 "$now [expr $bw0/$time*8/1000000]"
puts $f1 "$now [expr $bw1/$time*8/1000000]"
puts $f2 "$now [expr $bw2/$time*8/1000000]"
#Reset the bytes_ values on the traffic sinks
$sink0 set bytes_ 0
$sink1 set bytes_ 0
$sink2 set bytes_ 0
#Re-schedule the procedure
$ns at [expr $now+$time] "record"
}
```

این روال تعداد بایت‌های دریافت شده از سه sink ترافیک را می‌خواند. سپس پهنای باند را محاسبه خواهد کرد (در MBit/s) و آن را به سه فایل خروجی پیش از اینکه مقدارهای bytes درsinkهای ترافیک مجدداً تغییر کند با زمان‌های کنونی آن‌ها خواهد نوشت. سپس آن‌ها را دوباره زمان‌بندی می‌کند.

اجرای شبیه‌سازی:

حال ما می‌توانیم رخدادهای زیر را زمان‌بندی کنیم:

```
$ns at 0.0 "record"
$ns at 10.0 "$source0 start"
$ns at 10.0 "$source1 start"
$ns at 10.0 "$source2 start"
$ns at 50.0 "$source0 stop"
$ns at 50.0 "$source1 stop"
$ns at 50.0 "$source2 stop"
$ns at 60.0 "finish"
$ns run
```

در ابتدای روال "record " فراخوانی شده و پس از آن به صورت دوره‌ای هر ۰/۵ ثانیه خود را دوباره زمان‌بندی خواهد کرد. سپس منبع‌های ترافیک در هر ۱۰ ثانیه شروع کرده و در ۵۰ ثانیه متوقف می‌شوند. در ۶۰ ثانیه روال "finish" فراخوانی می‌شود.

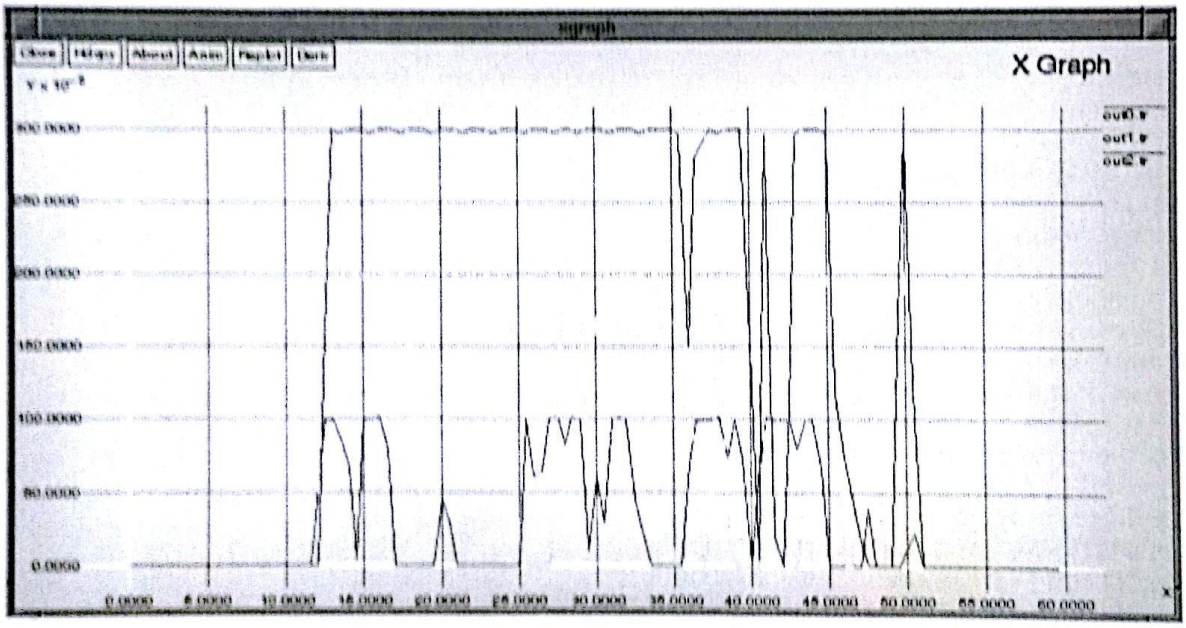
در ادامه کامل شده مثال را با نام example4.tcl آورده شده است.

```
#Create a simulator object
set ns [new Simulator]
#Open the output files
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]
#Create 5 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
#Connect the nodes
$ns duplex-link $n0 $n3 1Mb 100ms DropTail
$ns duplex-link $n1 $n3 1Mb 100ms DropTail
$ns duplex-link $n2 $n3 1Mb 100ms DropTail
$ns duplex-link $n3 $n4 1Mb 100ms DropTail
#Define a 'finish' procedure
proc finish {} {
global f0 f1 f2
#Close the output files
close $f0
close $f1
close $f2
#Call xgraph to display the results
exec xgraph out0.tr out1.tr out2.tr -geometry 800x400 &
exit 0
}
#Define a procedure that attaches a UDP agent to a previously created node
#'node' and attaches an Expoo traffic generator to the agent with the
```

```
#characteristic values 'size' for packet size 'burst' for burst time,
#'idle' for idle time and 'rate' for burst peak rate. The procedure connects
#the source with the previously defined traffic sink 'sink' and returns the
#source object.
proc attach-expoo-traffic { node sink size burst idle rate } {
#Get an instance of the simulator
set ns [Simulator instance]
#Create a UDP agent and attach it to the node
set source [new Agent/UDP]
$ns attach-agent $node $source
#Create an Expoo traffic agent and set its configuration parameters
set traffic [new Application/Traffic/Exponential]
$traffic set packetSize_ $size
$traffic set burst_time_ $burst
$traffic set idle_time_ $idle
$traffic set rate_ $rate
# Attach traffic source to the traffic generator
$traffic attach-agent $source
#Connect the source and the sink
$ns connect $source $sink
return $traffic
}
#Define a procedure which periodically records the bandwidth received by the
#three traffic sinks sink0/1/2 and writes it to the three files f0/1/2.
proc record {} {
global sink0 sink1 sink2 f0 f1 f2
#Get an instance of the simulator
set ns [Simulator instance]
#Set the time after which the procedure should be called again
set time 0.5
#How many bytes have been received by the traffic sinks?
set bw0 [$sink0 set bytes_]
set bw1 [$sink1 set bytes_]
set bw2 [$sink2 set bytes_]
#Get the current time
set now [$ns now]
#Calculate the bandwidth (in MBit/s) and write it to the files
puts $f0 "$now [expr $bw0/$time*8/1000000]"
puts $f1 "$now [expr $bw1/$time*8/1000000]"
puts $f2 "$now [expr $bw2/$time*8/1000000]"
#Reset the bytes_ values on the traffic sinks
$sink0 set bytes_ 0
$sink1 set bytes_ 0
$sink2 set bytes_ 0
#Re-schedule the procedure
$ns at [expr $now+$time] "record"
}
#Create three traffic sinks and attach them to the node n4
set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
set sink2 [new Agent/LossMonitor]
$ns attach-agent $n4 $sink0
$ns attach-agent $n4 $sink1
$ns attach-agent $n4 $sink2
#Create three traffic sources
set source0 [attach-expoo-traffic $n0 $sink0 200 2s 1s 100k]
```

```
set source1 [attach-expoo-traffic $n1 $sink1 200 2s 1s 200k]
set source2 [attach-expoo-traffic $n2 $sink2 200 2s 1s 300k]
#Start logging the received bandwidth
$ns at 0.0 "record"
#Start the traffic sources
$ns at 10.0 "$source0 start"
$ns at 10.0 "$source1 start"
$ns at 10.0 "$source2 start"
#Stop the traffic sources
$ns at 50.0 "$source0 stop"
$ns at 50.0 "$source1 stop"
$ns at 50.0 "$source2 stop"
#Call the finish procedure after 60 seconds simulation time
$ns at 60.0 "finish"
#Run the simulation
$ns run
```

وقتی که شما شبیه‌سازی را اجرا می‌کنید بعد از چند لحظه Xgraph ظاهر خواهد شد و نمودار آن شبیه به شکل زیر خواهد بود.



همان‌طوری که شما ملاحظه می‌فرمائید burstهای جریان اول و در peak حدود 0.1 Mbit/s قرار دارد و در جریان دوم 0.2 Mbit/s و در سومی 0.3 Mbit/s می‌باشد. حال اگر شما مقدار "time" در روال "record" را تغییر دهید خواهید دید که تغییراتی اتفاق خواهد افتاد. مثلاً آن را به ۰/۱ تغییر دهید و سپس به ۱/۰ تغییر دهید.

توجه کنید که در سناریوهای شبیه‌سازی پیدا کردن مقدار مناسب برای "time" خیلی مهم است. نکته دیگر اینکه فایل‌های خروجی که با روال "record" ایجاد می‌شوند همچنین می‌توانند با gnuplot (یک ابزار برای رسم نمودار) استفاده شوند.