Lab 8 TCP Error Control

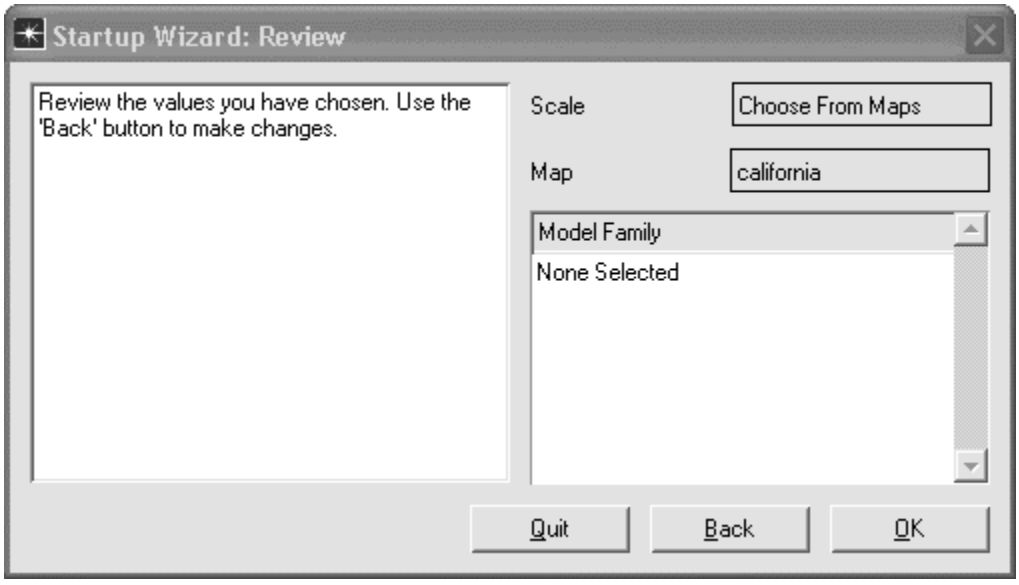
Overview
In order to correct for segments that have been lost or corrupted, TCP buffers data at the source and does retransmissions when necessary. Typically, losses are detected at the source via a timeout mechanism. If a segment is sent and an acknowledgement is not received before the timer expires, the segment is retransmitted. By default, TCP relies on cumulative acknowledgements. A cumulative acknowledgement for byte **x** acknowledges all bytes less than **x**. Since many segments are sent at once, it is often not clear to the TCP source which segments have been lost. As a result, the TCP source retransmits all unacknowledged segments when a loss is detected (a go-back-n strategy). This strategy may be inefficient if most of the segments actually did arrive at the destination.

An enhancement to TCP to deal with this inefficiency is the Selective Acknowledgment (SACK) option. If both the TCP source and destination agree to use this option, selective acknowledgements are used rather than cumulative. With this specific information regarding loss, the TCP source can do selective repeat retransmissions, i.e., retransmit only the lost segments, rather than all unacknowledged segments.

Objective
To examine the throughput of a TCP connection using the Selective Acknowledgment option when run over a lossy connection.

Build the Simulation Model

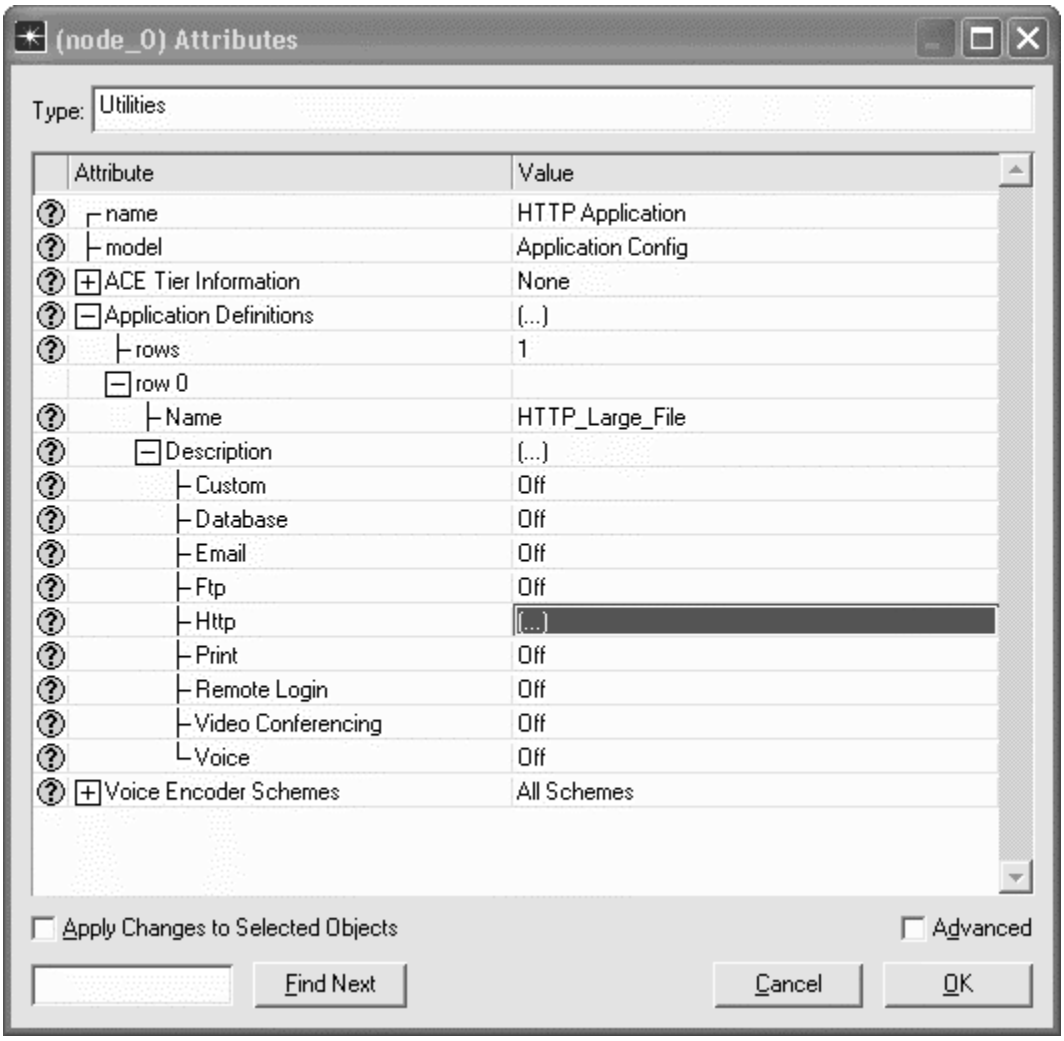
Start up OPNET IT Guru Academic Edition.
Select the **File** tab => **New...**
Choose **Project** and click on **OK**.
Change the **Project Name** to **xx_TCP_SACK** (where **xx** are your initials) and click on **OK**.
In the **Initial Topology** window, select **Create Empty Scenario** and click on **Next**.
In the **Choose Network Scale** window, select **Choose from Maps** and click on **Next**.
In the **Choose Map** window, choose **california** and click on **Next**.
In the **Select Technologies** window, click on **Next**.
In the **Review** window, click on **OK**.

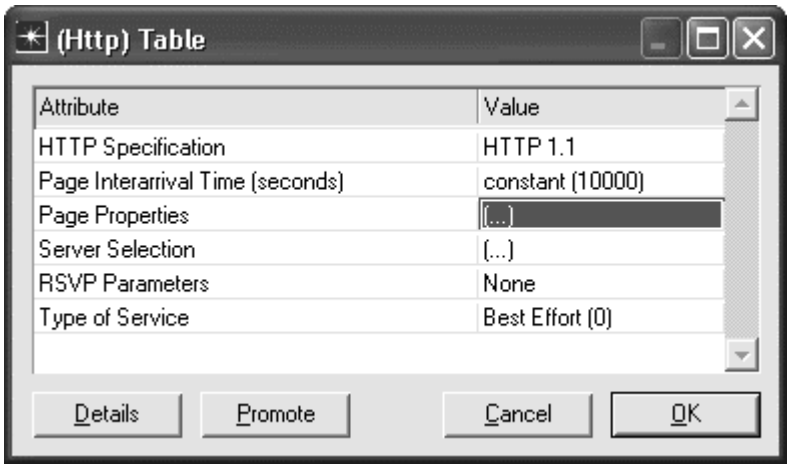


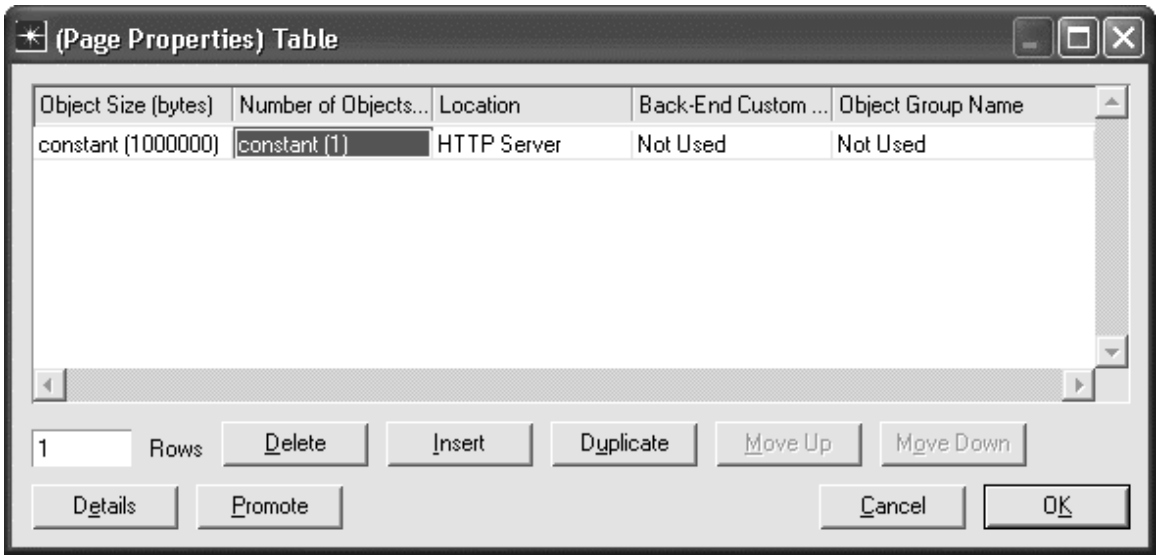
First, it would be helpful to familiarize yourself with the TCP functionality of OPNET.
Select the **Protocols** tab => **TCP** => **Model Usage Guide**. Read through the guide and click on the close window icon to close the PDF viewer when you are done.

Next, we need to configure an application which uses TCP, and associate the application with a profile that will drive our network devices. The HyperText Transfer Protocol (HTTP) is built on top of TCP and will serve our purposes.

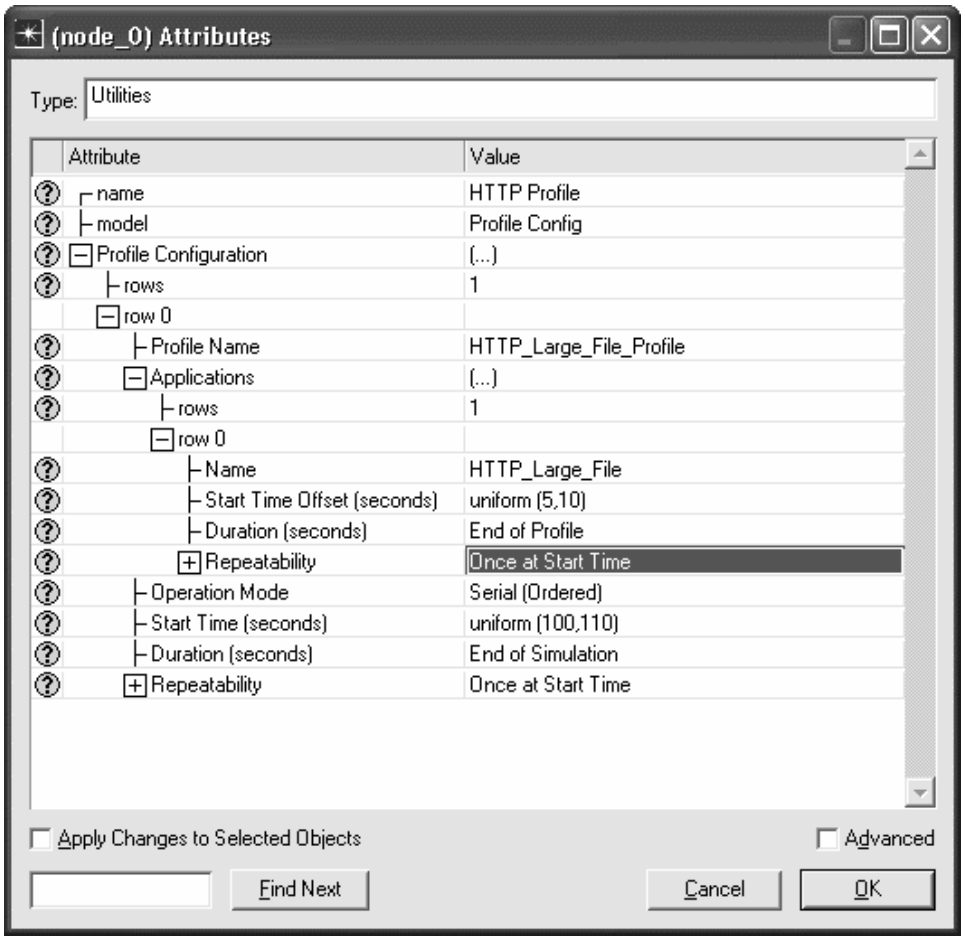
Select an **Application Config** object from the Object Palette and place it in the project workspace. Right click on the object and choose **Edit Attributes**. Set the **name** attribute to **HTTP Application**.
Expand the **Application Definitions** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** attribute to **HTTP_Large_File**. Expand the **Description** attribute and edit the value of the **Http** attribute. Set the **Page Interarrival Time (secs)** attribute to **constant(10000)**. Click on the **Page Properties** value field to bring up the **(Page Properties) Table** window. Set the **Object Size (bytes)** attribute to **constant(1000000)** and the **Number of Objects…** to **constant(1)**. You will need to set the **Special Value** field to **Not Used** in order to modify these values. The application you have now defined will transfer one 1-MB file at a time so that we can easily analyze the TCP behavior.
Click on **OK** three times to close the three windows.







Select a **Profile Config** object from the Object Palette and place it in the project workspace. Right click on the object and choose **Edit Attributes**. Set the **name** attribute to **HTTP Profile**.
Expand the **Profile Configuration** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Profile Name** to **HTTP_Large_File_Profile**. Expand the **Applications** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** to **HTTP_Large_File**. Set the **Repeatability** attribute to **Once at Start Time**.
Click on **OK** to close the window.



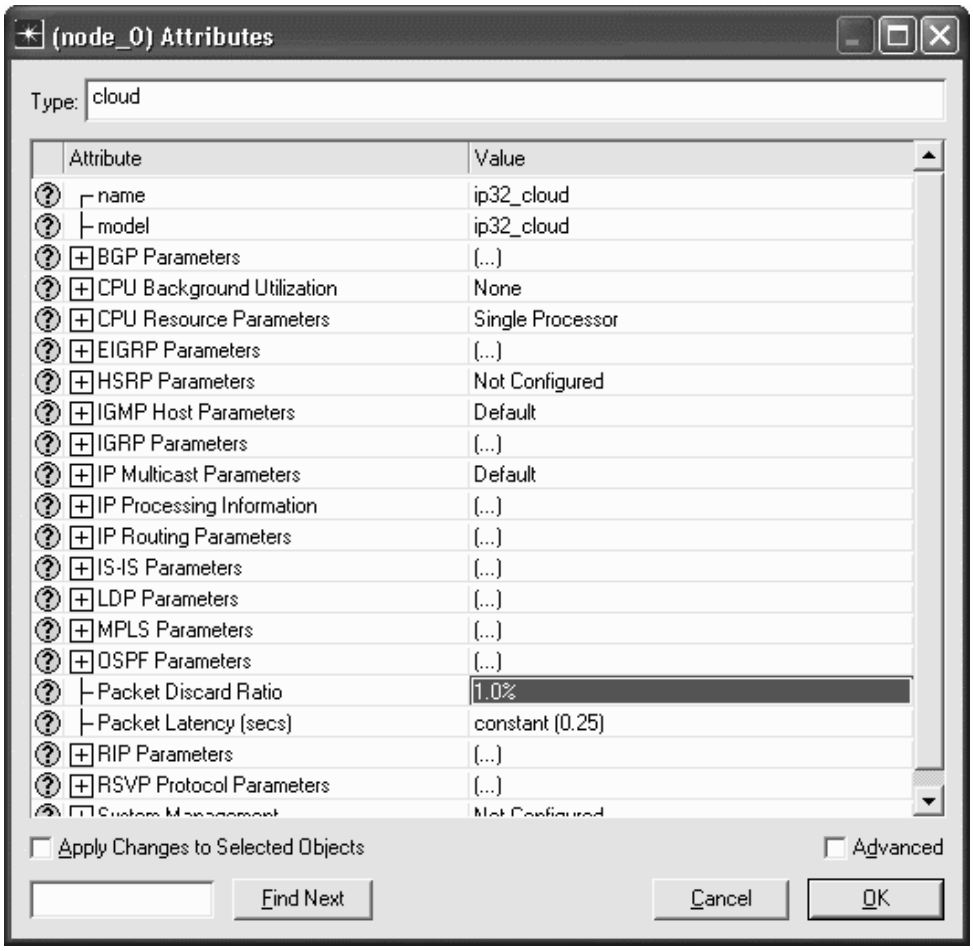
Select an **ip32_cloud** object from the Object Palette and place it in the project workspace. Right click on the cloud and choose **View Node Description**. The cloud represents a WAN consisting of IP-capable that supports up to 32 serial links

Right click on the cloud and select **Set Name**. Set the name to **ip32_cloud**. Click on OK to close the window.

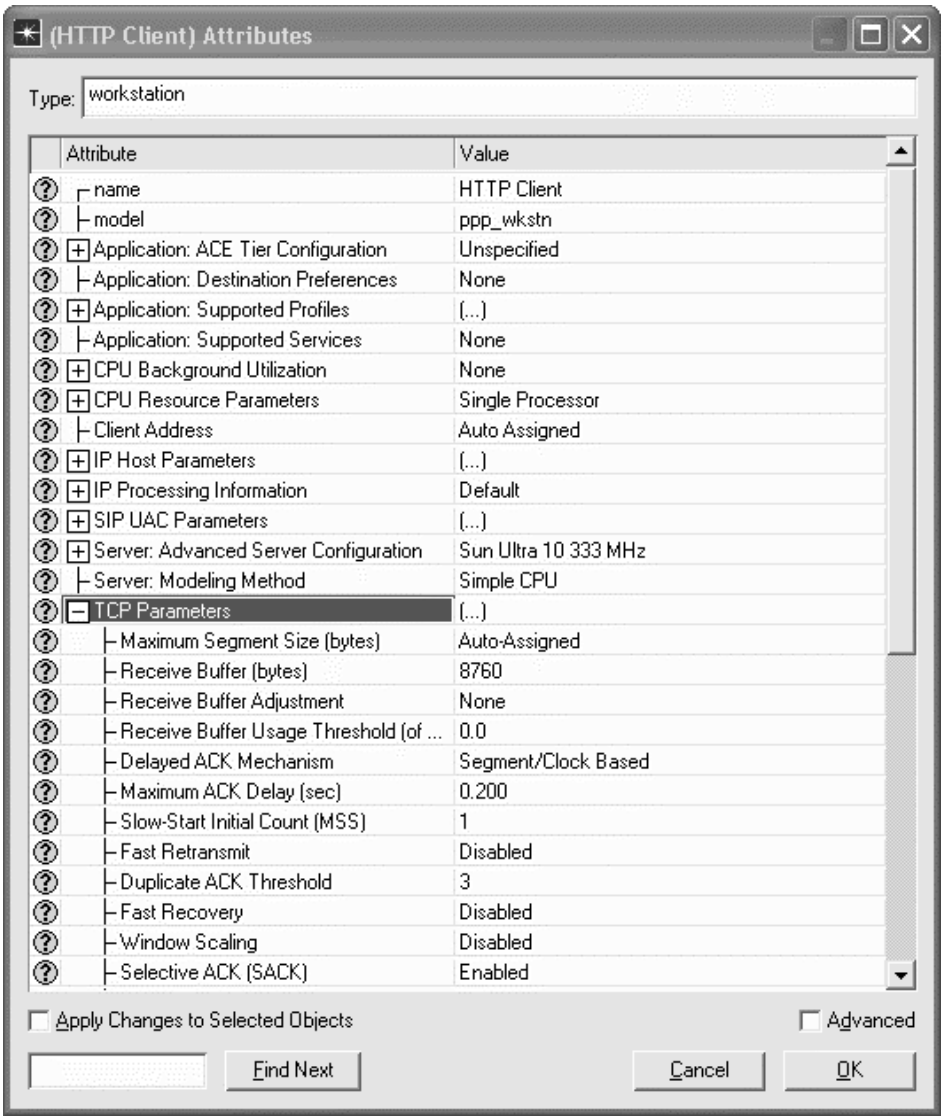
Right click on the cloud and select **Edit Attributes**. Set the **Packet Discard Ratio** to **1%**. This means that 1 in 100 packets will be discarded as they pass through the WAN and retransmissions will be required.

Set the **Packet Latency (secs)** to **constant(0.25)**. This means that the round-trip delay will be at least 0.5 second. You will need to set the **Special Value** field to **Not Used** in order to modify the latency.

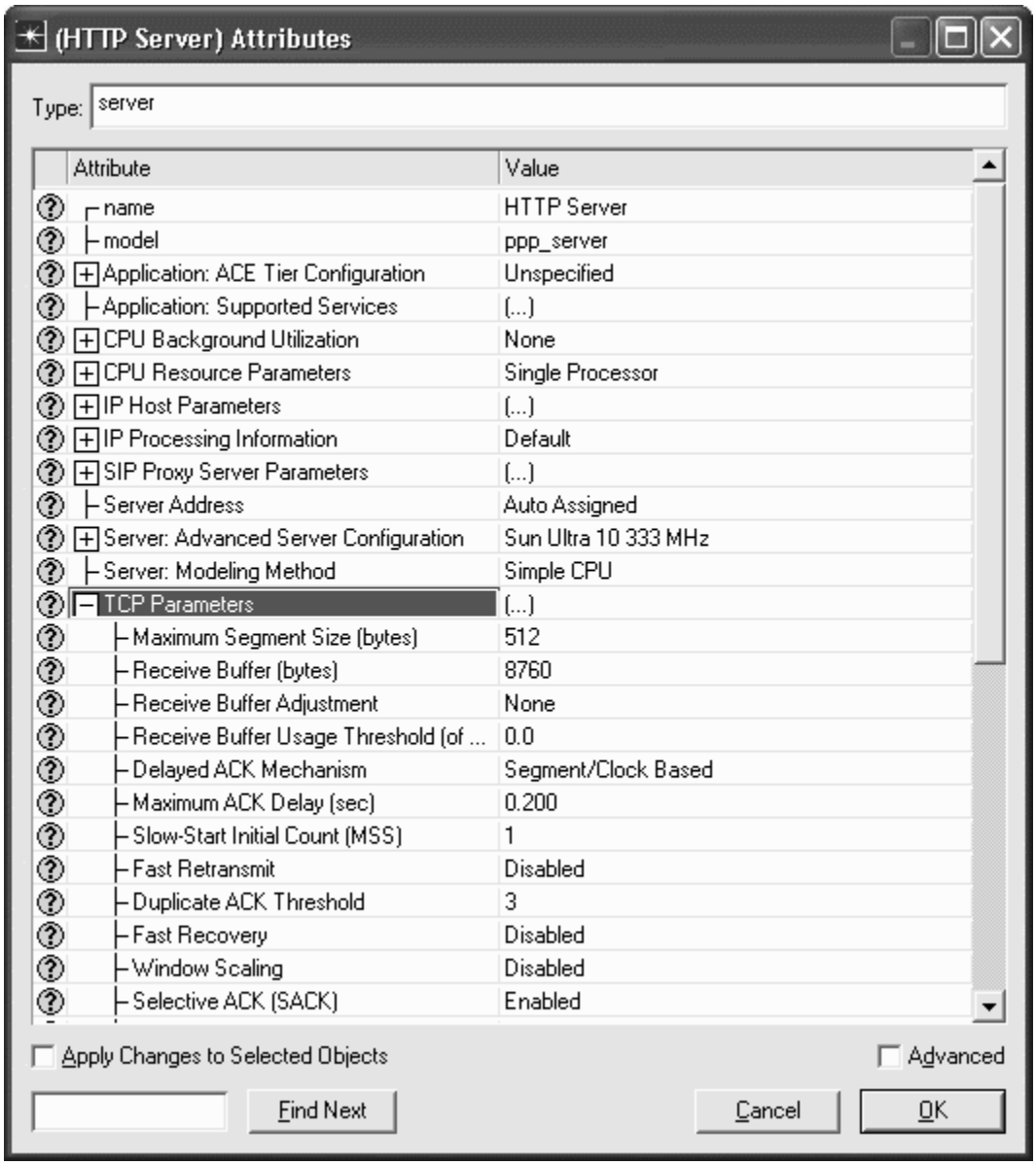
Click on **OK** to close the window.



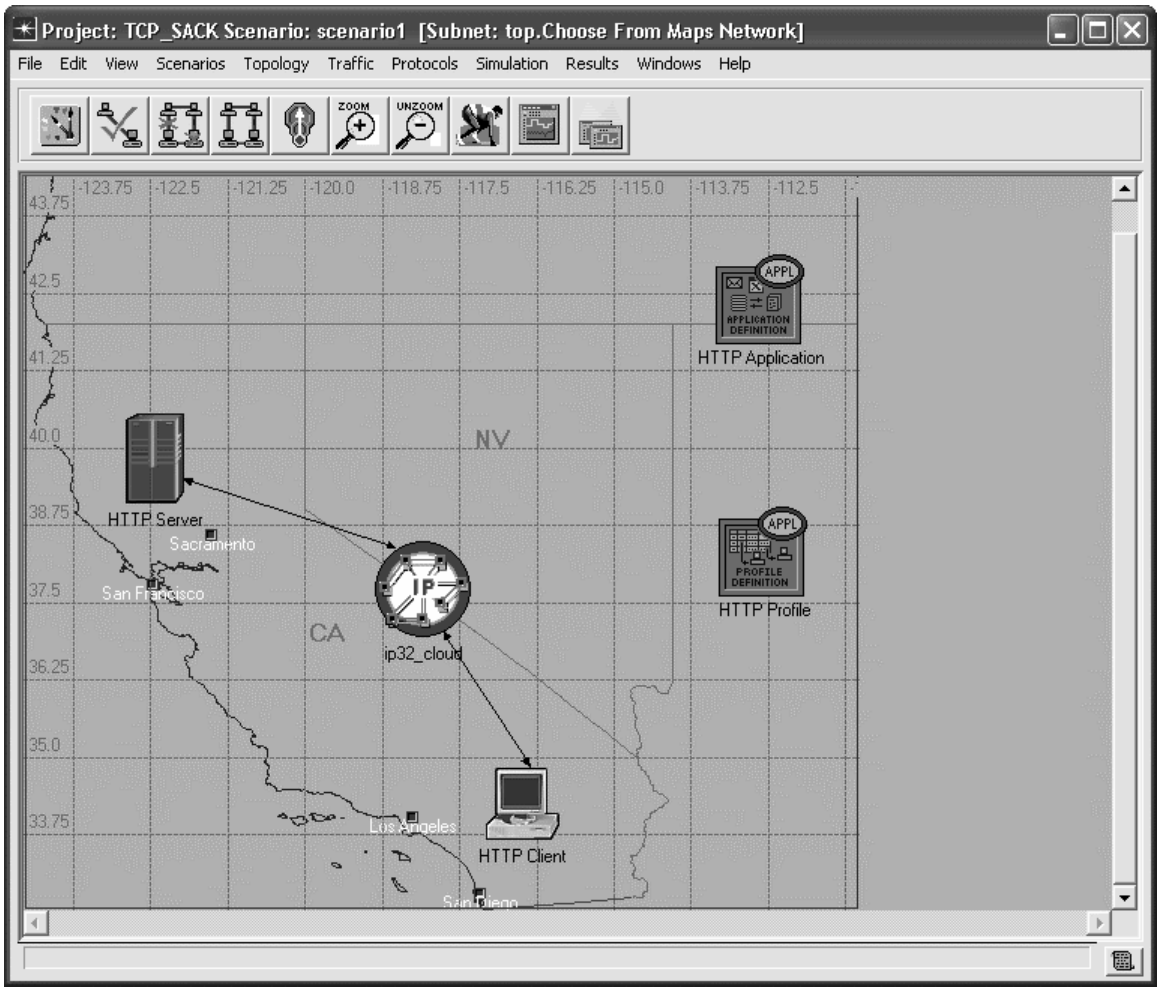
Select a **ppp_wkstn** device from the Object Palette and place it in the project workspace.
Right click on the station and choose **View Node Description**. Note that the station supports the TCP protocol. We will now set up the client to use the HTTP application that we defined.
Right click on the station and choose **Edit Attributes**. Modify the **name** attribute of the device to **HTTP Client**.
Edit the **Application: Supported Profiles** attribute. Insert a row and set the **Profile Name** to **HTTP_Large_File_Profile**.
Expand the **TCP Parameters** attribute and set the **Selective ACK (SACK)** attribute to **enabled**. Set the **Fast Retransmit** and **Fast Recovery** attributes to **disabled**. Fast Retransmit and Fast Recovery are alternate TCP error control mechanisms. By disabling them, we can concentrate on the effects of the SACK mechanism.
Click on **OK** to close the window.



Select a **ppp_server** device from the Object Palette and place it in the project workspace.
Right click on the server and choose **View Node Description**. Note that the server supports the TCP protocol. We will now set up the server to support the HTTP application that we defined.
Right click on the device and choose **Edit Attributes**. Modify the **name** attribute of the server to **HTTP Server**.
Edit the **Application: Supported Services** attribute. Set the **rows** field to **1**. In the first row, set the **Name** to **HTTP_Large_File**. Click on **OK** to close the table window.
Expand the **TCP Parameters** attribute and set the **Maximum Segment Size (bytes)** attribute to **512**. This will ensure that each TCP packet sent is 512 bytes long.
Set the **Selective ACK (SACK)** attribute to **enabled**. Set the **Fast Retransmit** and **Fast Recovery** attributes to **disabled**.
Click on **OK** to close the window.

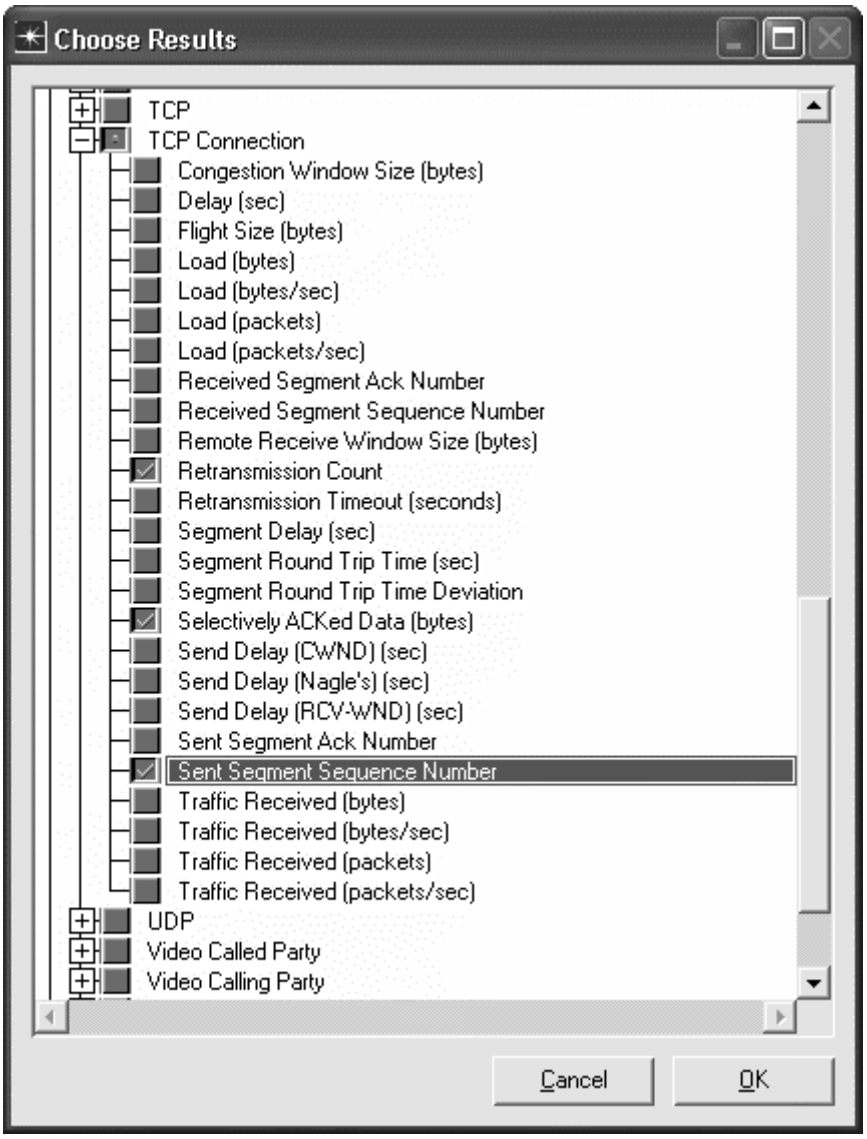


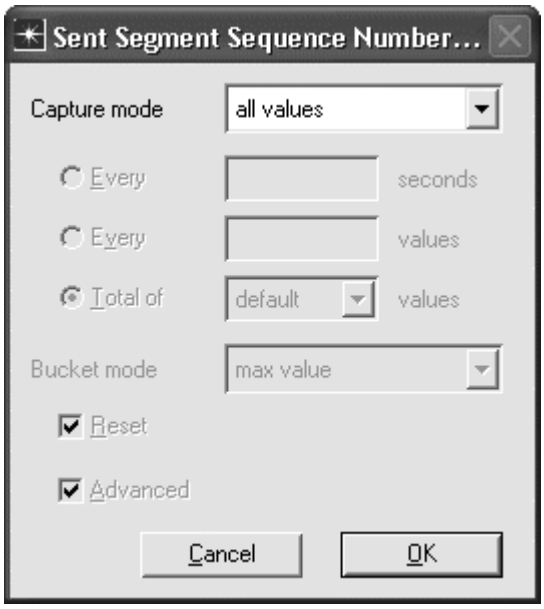
Select a **PPP_DS1** link from the Object Palette and use it to connect the HTTP client to the ip32_cloud. Use another to connect the HTTP server to the cloud. The PPP protocol is commonly used for long-distance links. Also, remember that DS1 speed is 1.5 Mbps.



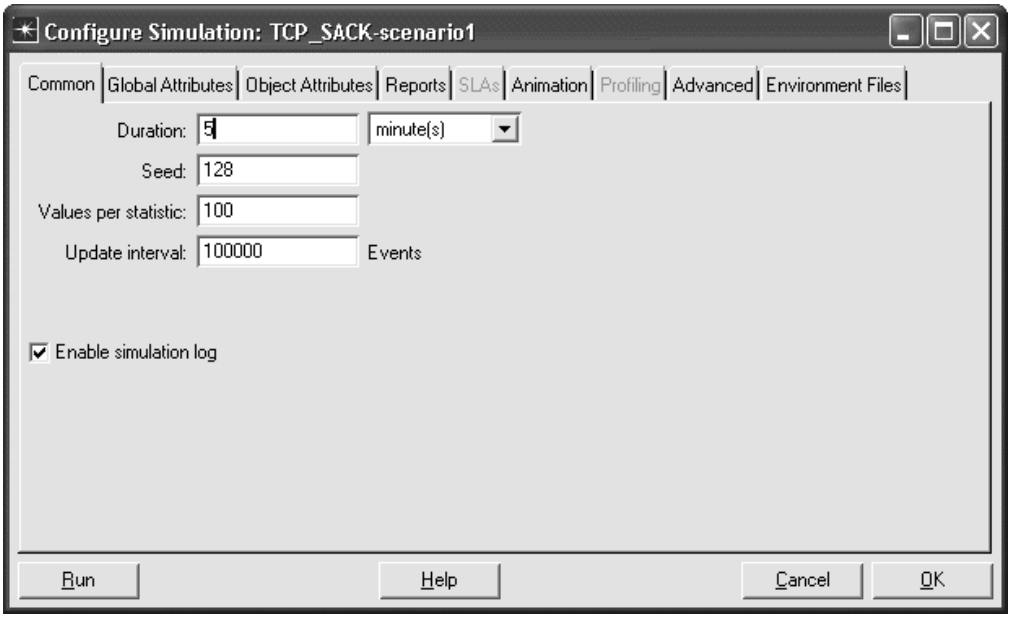
Configure and Run the Simulation

Select the **Simulation** tab => **Choose Individual Statistics…**
Expand the **Global Statistics** item and the **HTTP** item, and select the **Page Response Time (sec)** statistic.
Expand the **Node Statistics** item and the **TCP Connection** item, and select the **Retransmission Count**, **Selectively ACKed Data (bytes)**, and **Sent Segment Sequence Number** statistics. Right-click on the **Sent Segment Sequence Number** statistic, and choose **Change Collection Mode**. Click on **Advanced** and set the **Capture mode** to **all values**. This will ensure that IT Guru will store the sequence number of every packet sent to give us a more detailed graph.
Click on **OK** to close the window.


Choose Results
TCP
TCP Connection
Congestion Window Size (bytes)
Delay (sec)
Flight Size (bytes)
Load (bytes)
Load (bytes/sec)
Load (packets)
Load (packets/sec)
Received Segment Ack Number
Received Segment Sequence Number
Remote Receive Window Size (bytes)
Retransmission Count
Retransmission Timeout (seconds)
Segment Delay (sec)
Segment Round Trip Time (sec)
Segment Round Trip Time Deviation
Selectively ACKed Data (bytes)
Send Delay (CWND) (sec)
Send Delay (Nagle's) (sec)
Send Delay (RCV-WND) (sec)
Sent Segment Ack Number
Sent Segment Sequence Number
Traffic Received (bytes)
Traffic Received (bytes/sec)
Traffic Received (packets)
Traffic Received (packets/sec)
UDP
Video Called Party
Video Calling Party
Cancel
OK


Sent Segment Sequence Number...
Capture mode
all values
Every
seconds
Every
values
Total of
default
values
Bucket mode
max value
Reset
Advanced
Cancel
OK

Select **Simulation => Configure Discrete Event Simulation…**
Under the **Common** tab, set the **Duration** to **5**, and the unit to **minute(s)**.
Click on **Run** to run the simulation.
When the simulation has completed, click on **Close** to close the window.

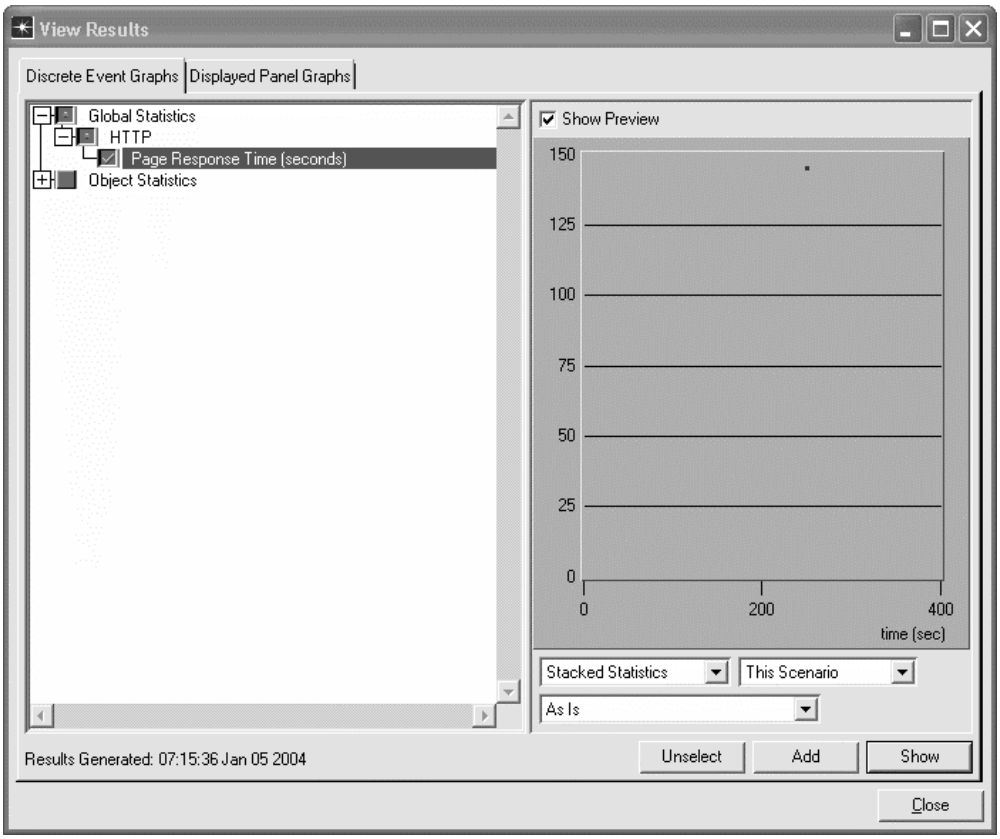


Inspect and Analyze Results

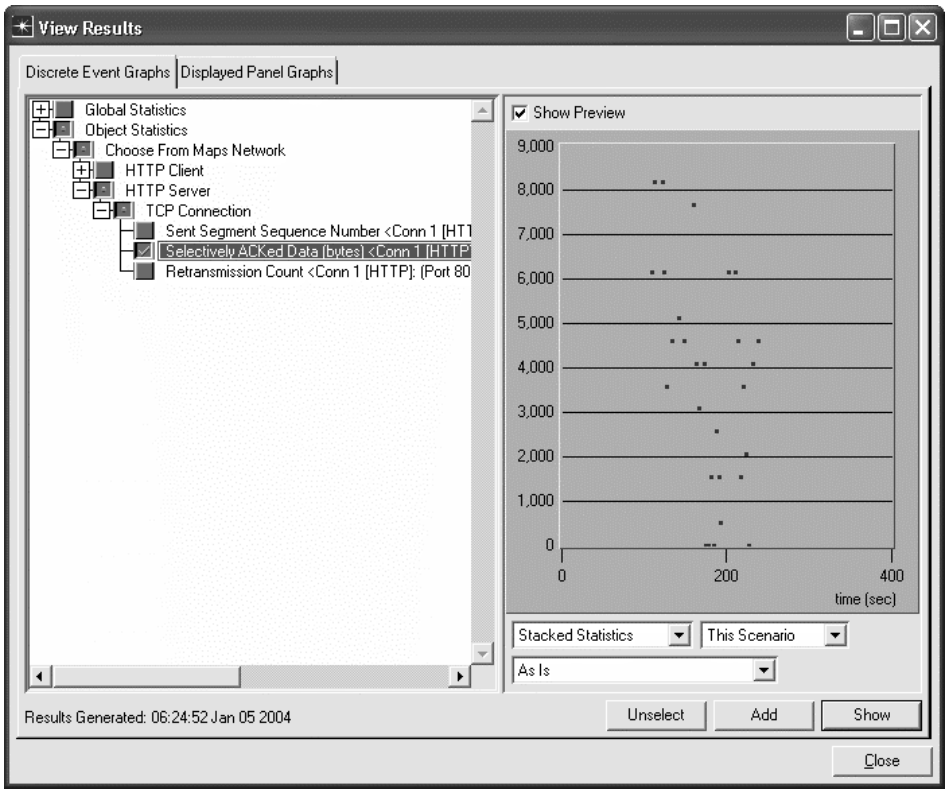
Select the **Results** tab => **View Results…**

Select and expand the **Global Statistics** item and the **Http** item, and select the **Page Response Time (sec)** statistic. This statistic shows how long the download took. There will be one point for each page downloaded. Click on the statistic again to disable the preview.

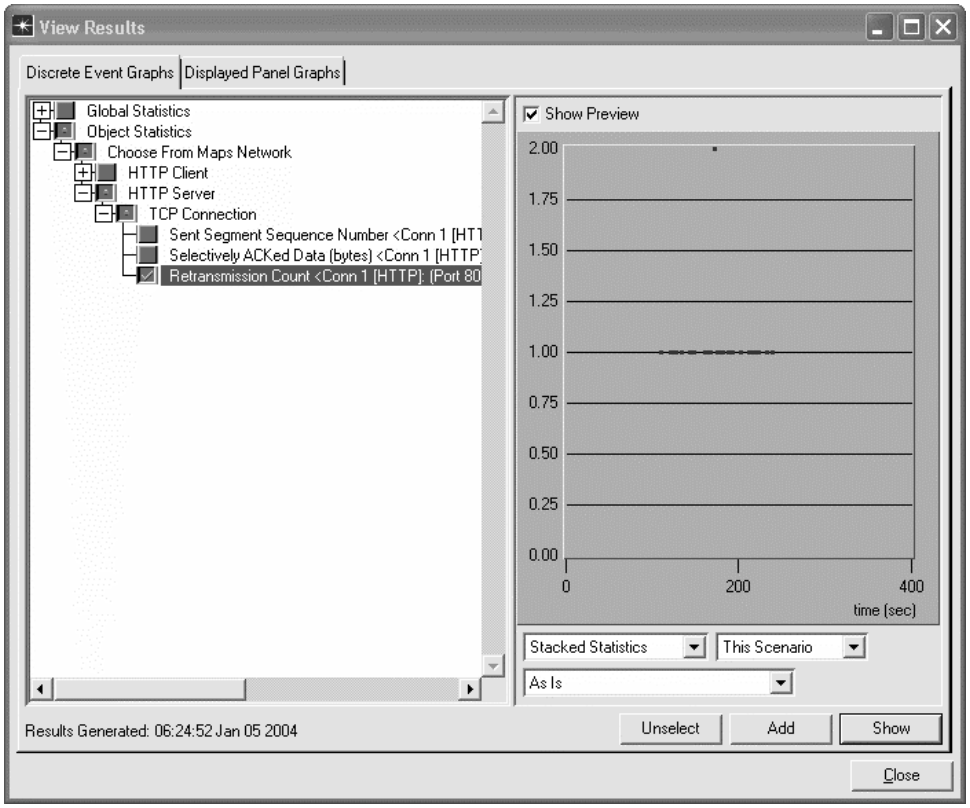
We can calculate the TCP throughput from the download response time. We set the file size to be 1 Mbyte when we configured the HTTP application [(1 Mbyte * 8 bits/byte) / 145 seconds = 55.2 Kbps].



Select and expand the **Object Statistics** item, the **Choose From Maps Network** item, the **HTTP Server** item, and the **TCP Connection** item. Select the **Selectively ACKed Data (bytes)** statistic. This graph shows how much data was selectively acknowledged as opposed to being acknowledged via the standard cumulative acknowledgment mechanism. You can see that a fair amount of data is selectively acknowledged at any time during the transfer indicating that the SACK mechanism is performing a useful service to the TCP source (the HTTP server in this case).
Click on the statistic again to disable the preview.



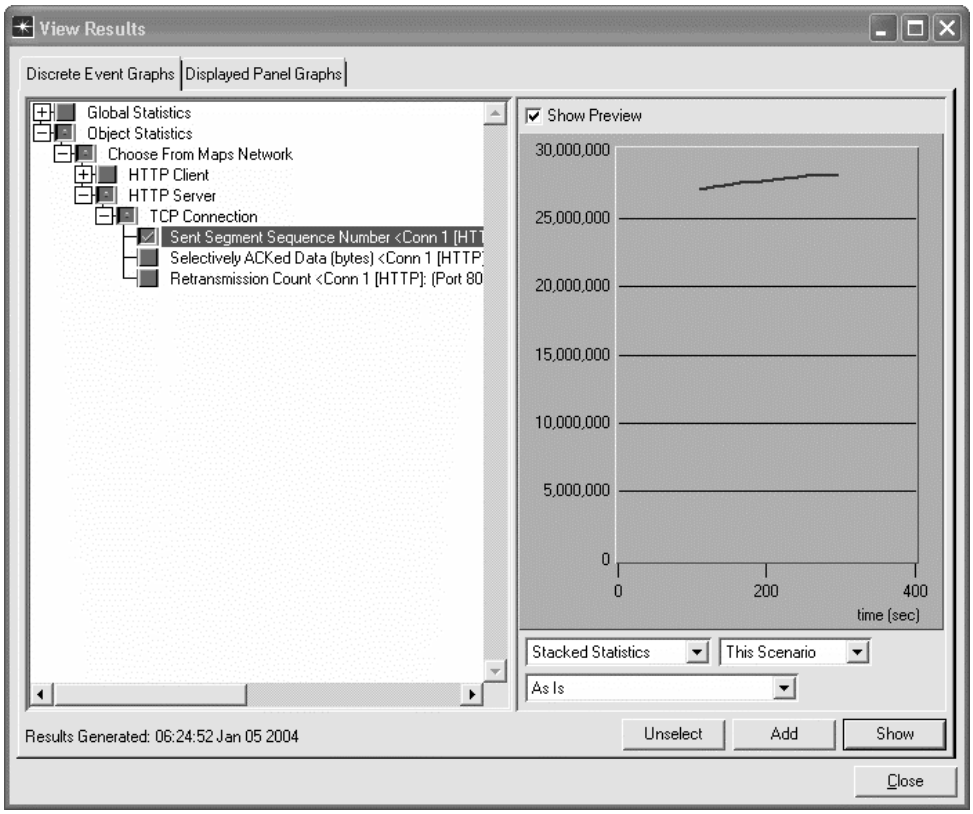
Now select the **Retransmission Count** statistic. From this graph, you can see that many retransmissions were required. When using the SACK option, these retransmissions may be done more efficiently.
Click on the statistic again to disable the preview.

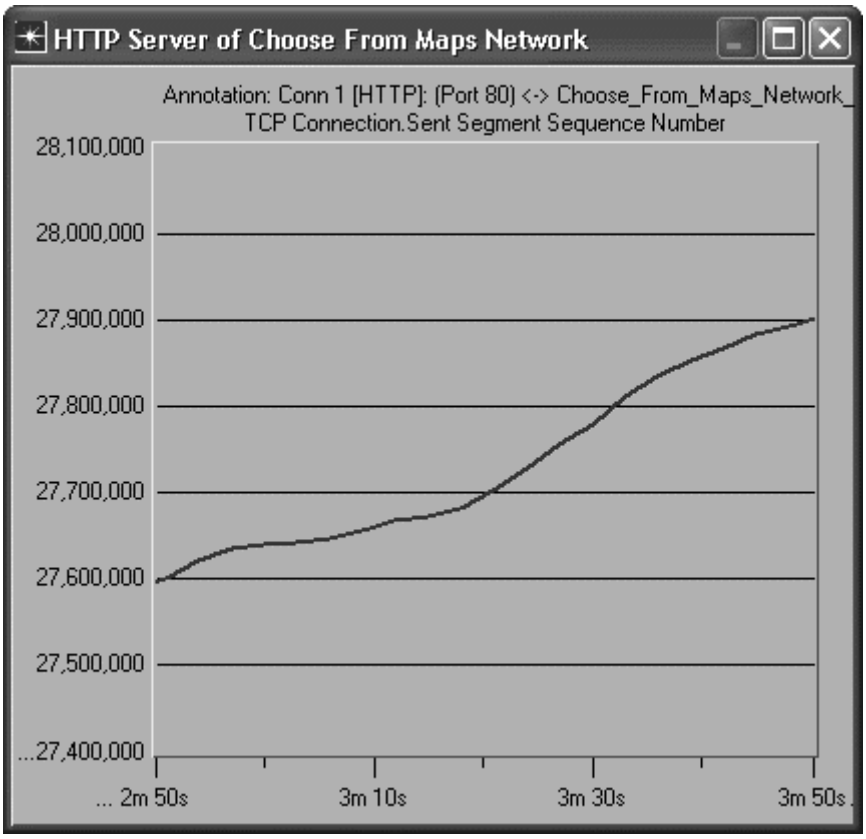


Finally, select the **Sent Segment Sequence Number** statistic. Click on the **Show** button and expand a small portion of the graph so that you can see more detail. From this graph, you can see that, while the throughput is not constant, the TCP source quickly recovers from errors. There are few or no spots in the graph where the sequence numbers do not increase.

Note that you may right click on the graph and select **Export Graph Data to Spreadsheet**. This will allow you to examine the actual time value/ sequence number pairs in more detail.

Click on the close window icon and choose to **Delete** the panel to close the statistics window. Click on **Close** to close the results window.





Save your model and close all windows.

Questions

1. Duplicate the scenario and modify the HTTP Client and HTTP Server TCP Connection attributes to disable Selective Acknowledgements. Rerun the simulation and record the HTTP Page Response Time. Calculate the associated TCP throughput. Does the change improve or degrade the response time and throughput?

2. Duplicate the scenario and repeat the simulation with Packet Discard Ratios of 0.0%, 0.5%, 5%, and 10%. Record the Page Response Time and, together with your results using a 1% discard ratio, plot the values using a spreadsheet. For what discard ratios is the SACK mechanism effective? Explain why.

3. Two additional TCP error control mechanisms are the Fast Retransmit and Fast Recovery schemes, which are usually implemented together. Duplicate the scenario and disable the SACK option on the HTTP server and HTTP Client. Set the discard rate to 1% in the ip32_cloud. Modify the values of the Fast Retransmit and Fast Recovery attributes under the TCP Parameters attribute on both the client and server. They should be set to enabled, and Reno respectively. Run the simulation and record the Page Response Time. Disable the Fast Retransmit and Fast Recovery attributes and rerun the simulation. Record the Page Response Time. How did the error control mechanisms affect the response time?

4. Compare the Page Response Time for three scenarios: a) Selective Acknowledgements only, b) Fast Retransmit and Fast Recovery only, c) Selective Acknowledgements combined with Fast Retransmit and Fast Recovery. Which scenario gives the best results? Why do you think this is so?