


Understanding Buffer Size Requirements in a Router

Thanks to
Nick McKeown and John Lockwood for numerous slides



Buffer Requirements in a Router

Buffer size matters:


- Small queues reduce delay
- Large buffers are expensive

Theoretical tools predict requirements

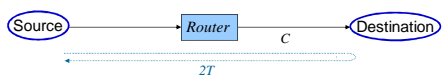
- Queuing theory
- Large deviation theory
- Mean field theory

Yet, there is no direct answer


- Flows have a closed-loop nature
- Question arises on whether focus should be on equilibrium state or transient state



Rule-of-thumb




- **Universally applied rule-of-thumb:**
 - A router needs a buffer size: $B = 2T \times C$
 - $2T$ is the two-way propagation delay (or just 250ms)
 - C is capacity of bottleneck link
- **Context**
 - Mandated in backbone and edge routers
 - Appears in RFPs and IETF architectural guidelines
 - Already known by inventors of TCP
 - [Van Jacobson, 1988]
 - Has major consequences for router design



The Story So Far

# packets at 10Gb/s	1,000,000	10,000	20
	$2T \times C \xrightarrow{(1)} \frac{2T \times C}{\sqrt{n}} \xrightarrow{(2)} O(\log W)$		

(1) Assume: Large number of desynchronized flows; 100% utilization
 (2) Assume: Large number of desynchronized flows; <100% utilization




Using NetFPGA to explore buffer size

- **Need to reduce buffer size and measure occupancy**
- **Alas, not possible in commercial routers**
- **So, we will use the NetFPGA instead**

Objective:

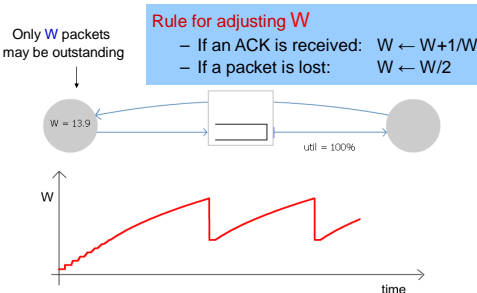

- Use the NetFPGA to understand how large a buffer we need for a **single** TCP flow.

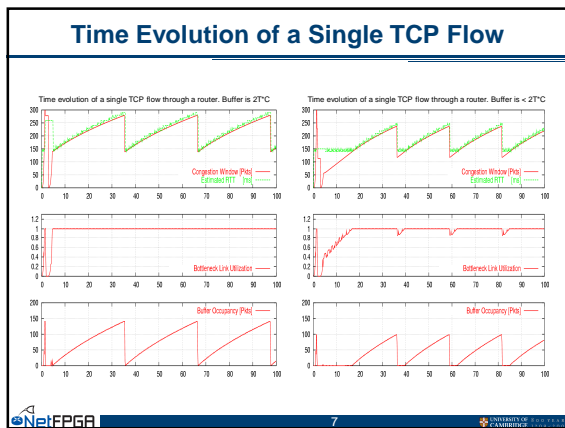


Why 2TxC for a single TCP Flow?

Rule for adjusting W

- If an ACK is received: $W \leftarrow W+1/W$
- If a packet is lost: $W \leftarrow W/2$

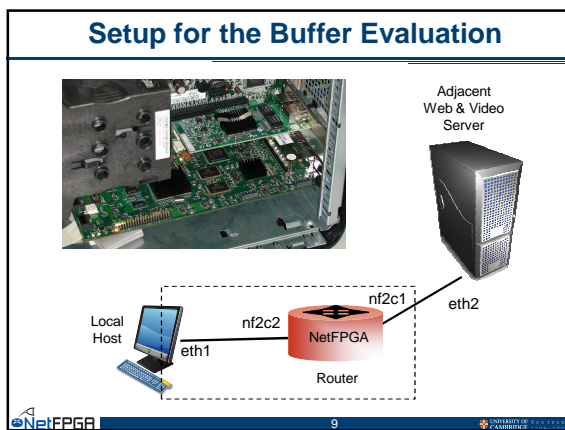





Observing and Controlling the Queue Size

Instructions for a real evaluation using NetFPGA

NetFPGA UNIVERSITY OF CAMBRIDGE



- ### Enhanced Router
- #### Objectives
- Observe router with new modules
 - New modules: rate limiting, event capture
- #### Execution
- Run event capture router
 - Look at routing tables
 - Explore details pane
 - Start tcp transfer, look at queue occupancy
 - Change rate, look at queue occupancy
- NetFPGA UNIVERSITY OF CAMBRIDGE

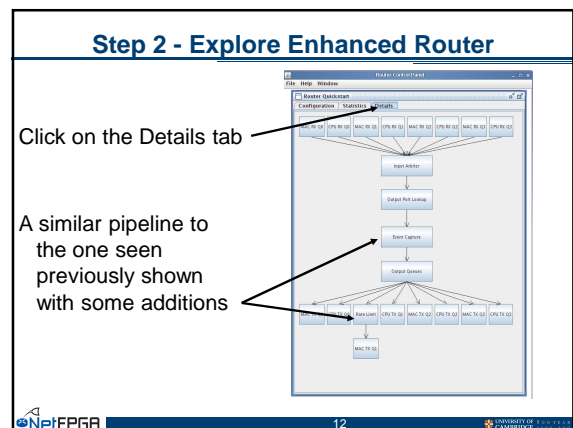
Step 1 - Run Pre-made Enhanced Router

Start terminal and cd to "NF2/projects/tutorial_router/sw"

Type ". / tut_adv_router_gui.pl"

A familiar GUI should start

NetFPGA UNIVERSITY OF CAMBRIDGE



Enhanced Router Pipeline

Two modules added

- 1. Event Capture** to capture output queue events (writes, reads, drops)
- 2. Rate Limiter** to create a bottleneck

The diagram illustrates the router pipeline. It starts with multiple MAC RxQ and CPU RxQ modules feeding into an Input Arbiter. The data then goes through an Output Port Lookup, followed by an Event Capture module. Finally, the data is sent to Output Queues, which are connected to MAC TxQ and CPU TxQ modules. A Rate Limiter module is shown being added to the output path between the Event Capture and the Output Queues.

Step 3 - Decrease the Link Rate

To create bottleneck and show the TCP "sawtooth," link-rate is decreased.

In the Details tab, click the "Rate Limit" module

Check Enabled

Set link rate to 1.953Mbps

The screenshot shows the configuration window for the Rate Limiter module. The 'Enabled' checkbox is checked. The link rate is set to 1.953Mbps. The current link rate is also shown as 1.953Mbps.

Step 4 - Decrease Queue Size

Go back to the Details panel and click on "Output Queues"

Select the "Output Queue 2" tab

Change the output queue size in packets slider to 16

The screenshot shows the configuration panel for Output Queues. The 'Output Queue 2' tab is selected. The queue size in packets is set to 16. The panel also shows various statistics and graphs for the queue.

Step 5 - Start Event Capture

Click on the Event Capture module under the Details tab

This should start the configuration page

The screenshot shows the configuration page for the Event Capture module. It includes sections for 'Event Capture' and 'Monitoring'. The 'Event Capture' section has several checkboxes for enabling capture and sending data to the local host. The 'Monitoring' section has checkboxes for monitoring specific queues.

Step 6 - Configure Event Capture

Check **Send to local host** to receive events on the local host

Check **Monitor Queue 2** to monitor output queue of MAC port1

Check **Enable Capture** to start event capture

The screenshot shows the configuration page for the Event Capture module. The 'Enable Capture' checkbox is checked. The 'Send to local host' checkbox is checked. The 'Monitor Queue 2' checkbox is checked.

Step 7 - Start TCP Transfer

We will use *iperf* to run a large TCP transfer and look at queue evolution

Start a terminal and cd "NF2/projects/tutorial_router/sw"

Type "./iperf.sh"

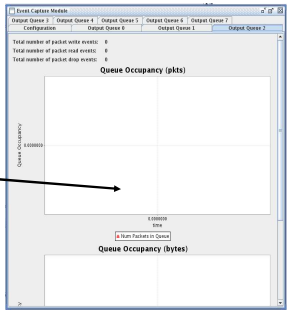
The screenshot shows a terminal window with the following commands and output:

```
root@NF2:~/projects/tutorial_router/sw# ./iperf.sh
iperf: error: can't read /usr/share/iperf/iperf.conf: No such file or directory
```


Step 8 - Look at Event Capture Results

Click on the Event Capture module under the Details tab.

The sawtooth pattern should now be visible.




The screenshot shows the 'Event Capture Module' window with several tabs for different queues. The 'Details' tab is active, showing statistics for 'Queue 0' through 'Queue 7'. Below the statistics are two graphs: 'Queue Occupancy (pkts)' and 'Queue Occupancy (bytes)'. An arrow points from the text 'The sawtooth pattern should now be visible.' to the 'Queue Occupancy (pkts)' graph.


19
UNIVERSITY OF CAMBRIDGE


Queue Occupancy Charts

Observe the TCP/IP sawtooth



The screenshot shows two graphs: 'Queue Occupancy (pkts)' and 'Queue Occupancy (bytes)'. Both graphs display a sawtooth pattern, indicating periodic queue clearing. A person is visible in the foreground, pointing at the graphs.

Leave the control windows open


20
UNIVERSITY OF CAMBRIDGE