

Презентация к лабораторной работе 2

Исследование протокола TCP и алгоритма управления очередью RED

Сидорова Н.А.

16 февраля 2025

Российский университет дружбы народов, Москва, Россия

Объединённый институт ядерных исследований, Дубна, Россия

Пример с дисциплиной RED

```

# создание объекта Simulator
set ns [new Simulator]
# открытие на запись файла out.nam для визуализатора nam
set nf [open out.nam w]
# все результаты моделирования будут записаны в переменную nf
$ns namtrace-all $nf
# открытие на запись файла трассировки out.tr
# для регистрации всех событий
set f [open out.tr w]
# все регистрируемые события будут записаны в переменную f
$ns trace-all $f
# процедура finish закрывает файлы трассировки
# и запускает визуализатор nam
# Процедура finish:
proc finish {} {
    global tchan
    # подключение кода AWK:
    set awkCode {
        {
            if ($1 == "Q" && NF>2) {
                print $2, $3 >> "temp.q";
            }
            else if ($1 == "a" && NF>2) {
                print $2, $3 >> "temp.a";
            }
        }
    }
    set f [open temp.queue w]
    puts $f "TitleText: red"
    puts $f "Device: Postscript"
    if { [info exists tchan_] } {
        close $tchan_
    }
    exec rm -f temp.q temp.a
    exec touch temp.a temp.q
    exec awk $awkCode all.q # выполнение кода AWK
    puts $f "\"queue
    exec cat temp.q >> $f
    puts $f "\\n\"ave_queue
    exec cat temp.a >> $f
    close $f
    # Запуск xgraph с графиком окна TCP и очереди:
    exec xgraph -bb -tk -x time -t "TCP Reno Cwnd" WindowVsTimeReno &
    exec xgraph -bb -tk -x time -y queue temp.queue &
    exit 0
}

# Формирование файла с данными о размере окна TCP:
proc plotWindow {tcpSource file} {
    global ns
    set time 0.01
    # ...
}

```

Рис. 1: 1 часть кода

Продолжение кода примера

```
* /home/openmodelica/mip/lab-ns/lab2_0.tcl - Mousepad
Файл  Правка  Поиск  Вид  Документ  Справка

set now [$ns now]
set cwnd [$tcpSource set cwnd]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file"
}

# Узлы сети:
set N 5
for {set i 1} {$i < $N} {incr i} {
  set node_($i) [$ns node]
}
set node_(r1) [$ns node]
set node_(r2) [$ns node]

# Соединения:
$ns duplex-link $node_(s1) $node_(r1) 10Mb 2ms DropTail
$ns duplex-link $node_(s2) $node_(r1) 10Mb 3ms DropTail
$ns duplex-link $node_(r1) $node_(r2) 1.5Mb 20ms RED
$ns queue-limit $node_(r1) $node_(r2) 25
$ns queue-limit $node_(r2) $node_(r1) 25
$ns duplex-link $node_(s3) $node_(r2) 10Mb 4ms DropTail
$ns duplex-link $node_(s4) $node_(r2) 10Mb 5ms DropTail
# Агенты и приложения:
set tcp1 [$ns create-connection TCP/Reno
$node_(s1) TCPSink $node_(s3) 0]
$tcp1 set window_ 15
set tcp2 [$ns create-connection TCP/Reno
$node_(s2) TCPSink $node_(s3) 1]
$tcp2 set window_ 15
set ftp1 [$tcp1 attach-source FTP]
set ftp2 [$tcp2 attach-source FTP]

# Мониторинг размера окна TCP:
set windowVsTime [open WindowVsTimeReno w]
set qmon [$ns monitor-queue $node_(r1) $node_(r2)
[open qmon.out w] 0.1];
[$ns link $node_(r1) $node_(r2)] queue-sample-timeout;
# Мониторинг очереди:
set redq [[$ns link $node_(r1) $node_(r2)] queue]
set tchan [open all.q w]
$redq trace curq_
$redq trace ave_
$redq attach $tchan_

# Добавление ат-событий:
$ns at 0.0 "$ftp1 start"
$ns at 1.1 "plotWindow $tcp1 $windowVsTime"
$ns at 3.0 "$ftp2 start"
$ns at 10 "finish"
# Запуск модели
$ns run
```

Рис. 2: 2 часть кода

Запустила Xgraph

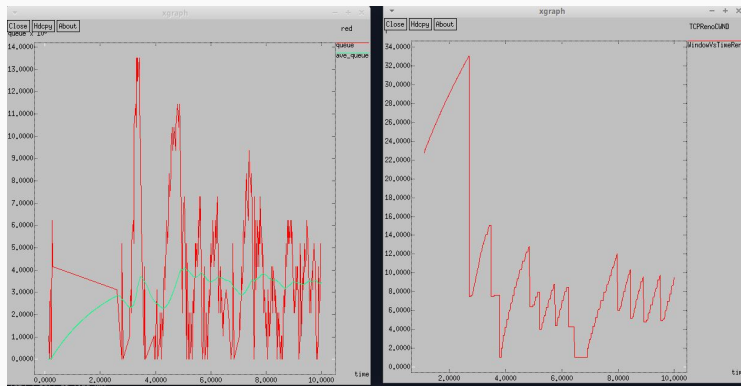


Рис. 3: Графики

```
# Агенты и приложения:  
set tcp1 [$ns create-connection TCP/Newreno $node_(s1) TCPSink $node_(s3) 0]  
$tcp1 set window_ 15  
set tcp2 [$ns create-connection TCP/Reno $node_(s2) TCPSink $node_(s3) 1]  
$tcp2 set window_ 15  
set ftp1 [$tcp1 attach-source FTP]  
set ftp2 [$tcp2 attach-source FTP]
```

Рис. 4: NewReno

NewReno график

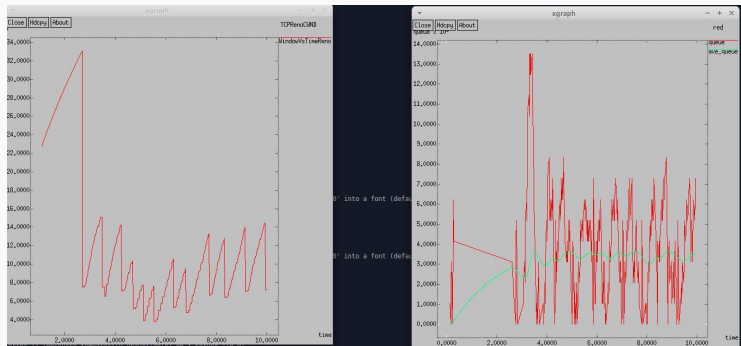


Рис. 5: NewReno график

```
|  
# Агенты и приложения:  
set tcp1 [$ns create-connection TCP/Vegas $node_(s1) TCPSink $node_(s3) 0]  
$tcp1 set window_ 15  
set tcp2 [$ns create-connection TCP/Reno $node_(s2) TCPSink $node_(s3) 1]  
$tcp2 set window_ 15  
set ftp1 [$tcp1 attach-source FTP]  
set ftp2 [$tcp2 attach-source FTP]
```

Рис. 6: Vegas

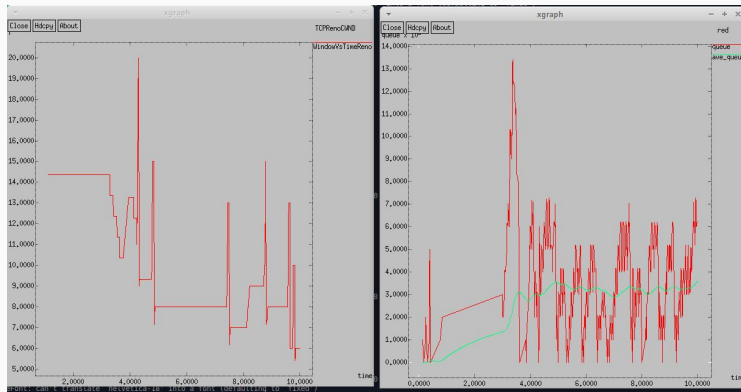


Рис. 7: Vegas график

Изменение оформления графика

```
# Процедура finish:
proc finish {} {
    global tchan_
    # подключение кода AWK:
    set awkCode {
        {
            if ($1 == "Q" && NF>2) {
                print $2, $3 >> "temp.q";
                set end $2
            }
            else if ($1 == "a" && NF>2)
                print $2, $3 >> "temp.a";
        }
    }
    set f [open temp.queue w]

    puts $f "TitleText: red"
    puts $f "Device: Postscript"
    puts $f "0.Color: White"
    puts $f "1.Color: Pink"

    if { [info exists tchan_] } {
        close $tchan_
    }
    exec rm -f temp.q temp.a
    exec touch temp.a temp.q
    exec awk $awkCode all.q
    puts $f "Line"
    exec cat temp.q >@ $f|
    puts $f \n\"M_Line
    exec cat temp.a >@ $f
    close $f
    # Запуск xgraph с графиками окна TCP и очереди:
    exec xgraph -fg pink -bg purple -bb -tk -x time -t "TCPRenoCWND" WindowVsTimeReno &
    exec xgraph -fg green -bg purple -bb -tk -x time -y line temp.queue &
    exit 0
}
```

```
set ttp2 [$tcp2 attach-source FTP]

# Мониторинг размера окна TCP:
set windowVsTime [open WindowVsTimeReno w]
puts $windowVsTime "0.Color: White"
puts $windowVsTime "\"Razmer_Okna"

set amon [$ns monitor queue $node ($s1) $node ($
```

Рис. 9: 2 часть

Новый график

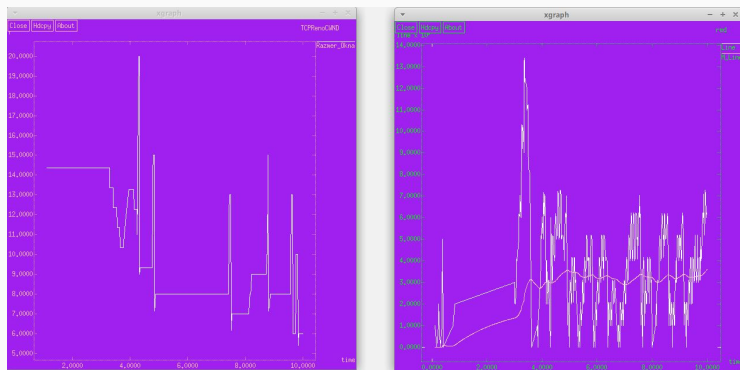


Рис. 10: Новый график