

# Лабораторная работа №2

Исследование протокола TCP и алгоритма управления очередью RED

---

Кадров Виктор Максимович

16 февраля 2025

Российский университет дружбы народов имени Патриса Лумумбы, Москва, Россия

Исследовать протокол TCP и алгоритм управления очередью RED.

1. Рассмотреть пример с дисциплиной RED.
2. Изменить в модели на узле s1 тип протокола TCP с Reno на NewReno, затем на Vegas.  
Сравнить и пояснить результаты.
3. Внести изменения при отображении окон с графиками (изменить цвет фона, цвет траекторий, подписи к осям, подпись траектории в легенде).

Постановка задачи. Описание моделируемой сети: – сеть состоит из 6 узлов; – между всеми узлами установлено дуплексное соединение с различными пропускной способностью и задержкой 10 мс; – узел r1 использует очередь с дисциплиной RED для накопления пакетов, максимальный размер которой составляет 25; – TCP-источники на узлах s1 и s2 подключаются к TCP-приёмнику на узле s3; – генераторы трафика FTP прикреплены к TCP-агентам.

## Скрипт модели с дисциплиной RED

```

Terminale - qtconsole/qtconsole-VirtualBox-16bit2
File Edit Shell Terminale View Window Help
@QtConsole 2.9.3 lab2.tc QtConsole

ns ns [new simulator]
ns sf [open out.com w]
ns nantrace-all $f
ns f [open out.tr.w]
ns trace-all $f

new finish () {
    global tchan;
    # message q chan $f
    set subCode 1
    if ($1 == "q" && NF>2) {
        print $2, $3 == "temp.q";
        set end $2
    }
    else if ($1 == "w" && NF>2) {
        print $2, $3 == "temp.w";
    }
}
set f [open temp.queue w]
puts $f "titleText: ran"
puts $f "Device: Postscript"
if ([info exists tchan.] ) {
    close $tchan;
}

exec rm -f temp.q temp.a
exec touch temp.a temp.q
exec mv $subCode all.q @ autoNewName.sgg $sk
puts $f "queue"
exec cat temp.q @ $f
puts $f "n: new queue"
exec cat temp.a @ $f
close $f

# message q chan $f
ns graph -db -tk -x time -t "TCPBaseCWD" Window/Viniferno &
exec graph -db -tk -x time -y queue Temp.queue &
exit 0

# message q chan $f
new plotWindow (tcpSource file) {
    global ns
    set time 0.01
    set now [now round]
    set cwnd [tcpSource cwnd]
    puts $file "[now round]"
    exec x [now roundtime] "plotWindow tcpSource $file"
}

```

Рис. 1: Скрипт модели с дисциплиной RED

# Скрипт модели с дисциплиной RED

```
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}

$ns duplex-link $node_($i) $node_r1 10Mb 2ms DropTail
$ns duplex-link $node_($i) $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 $node_($i) $node_r2 10Mb 4ms DropTail
$ns duplex-link $node_($i) $node_r2 10Mb 5ms DropTail

set tcp1 [$ns create-connection TCP/Reno $node_($i) TCPSink $node_($i) 0]
$tcp1 set window 15
set tcp2 [$ns create-connection TCP/Reno $node_($i) TCPSink $node_($i) 1]
$tcp2 set window 15
set ftp1 [$tcp1 attach-source FTP]
set ftp2 [$tcp2 attach-source FTP]

set windowVsTime [open WindowVsTimeReno w]
set qmon [$ns link $node_r1 $node_r2] [open qm.out w] 0.1;
$ns link $node_r1 $node_r2 queue-sample-timeout;
# Monitor queue discipline
set redq [$ns link $node_r1 $node_r2] queue]
set tchan [open all.q w]
$redq trace curq
$redq trace ave
$redq attach $tchan

$ns at 0.0 "ftpl start"
$ns at 1.1 "plotWindow $tcp1 $windowVsTime"
$ns at 3.0 "ftfp2 start"
$ns at 10 "finish"
```

Unbound key: M-4

Помощь	Записать	Поиск	Вырезать	Выровнять	ТонПозиц	Отмена	Отметить
Выход	ЧитФайл	Замена	Отменить	Словари	К строке	Повтор	Копировать

Рис. 2: Скрипт модели с дисциплиной RED

График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP Reno на узле s1.

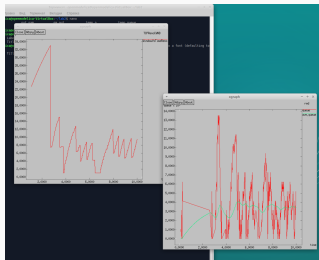


Рис. 3: График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP Reno на узле s1

## Изменения в модели на узле s1 типа протокола TCP с Reno на NewReno, затем на Vegas.

Скрипт изменений на узле s1 типа протокола TCP с Reno на Newreno.

```
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: Скрипт изменений на узле s1 типа протокола TCP с Reno на Newreno



График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP NewReno на узле s1.

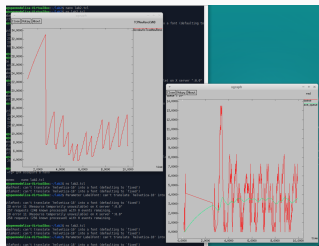


Рис. 5: График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP NewReno на узле s1

## Скрипт изменений на узле s1 типа протокола TCP с Reno на Vegas.

```
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]

set windowVsTime [open WindowVsTimeVegas w]
set proc [$ns monitor-queue $node_(s1) $node_(s2) [open proc.txt w] 0 1]
```

Рис. 6: Скрипт изменений на узле s1 типа протокола TCP с Reno на Vegas

График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP Vegas на узле s1.

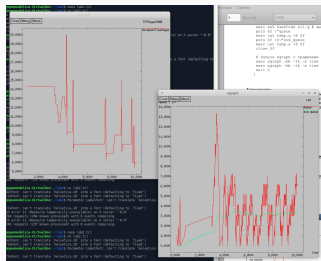


Рис. 7: График динамики размера окна TCP(сверху) и график динамики длины очереди и средней длины очереди(снизу) при типе протокола TCP Vegas на узле s1

Изменения при отображении окон с графиками (изменить цвет фона, цвет траекторий, подписи к осям, подпись траектории в легенде).

Изменение процедуры finish.

```
proc finish () {  
    global tchan  
    # finish procedure  
    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 Algorithm"  
    puts $f "Device: Postscript"  
    puts $f "Q.Color: Blue"  
    puts $f "I.Color: Orange"  
    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 "\nqueue"  
    exec cat temp.q > $f  
    puts $f "\nave queue"  
    exec cat temp.a >> $f  
    close $f  
    # cleanup  
    exec xgraph -bg white -bb -tk -x time -y "Window Size" -t "TCP Vegas CMD" WindowVsTimeVegas &  
    exec xgraph -bg white -bb -tk -x time -y "Queue" temp.queue &  
    exit 0  
}
```

Рис. 8: Изменение процедуры finish

## Изменение мониторинга размера окна TCP.

```
set tcp2 [tcp2 attach source r1]  
  
set windowVsTime [open WindowVsTimeVegas w]  
puts $windowVsTime "0.Color: Blue"  
set qmon [$ns monitor-queue $node_(r1) $node_(r2) [open qm.out w] 0.1];  
[$ns link $node_(r1) $node_(r2)] queue-sample-timeout;  
# Мониторинг очереди:  
set rade [$ns link $node_(r1) $node_(r2)] queue
```

Рис. 9: Изменение мониторинга размера окна TCP

# Результаты изменений отображения окон с графиками.



Рис. 10: Результаты изменений отображения окон с графиками

Мы исследовали протокол TCP и алгоритм управления очередью RED.