

Trabajo final

Tráfico

Análisis de algoritmos de control de congestión TCP:
BBRv1, BBRv2, Reno y Cubic en Mininet

Bernardi, Martín
Remedi, Augusto
Rittano, Ignacio

Introducción

Introducción

- Herramientas
 - VM con BBRv2
 - Mininet
 - iperf
 - captcp y tcpdump
 - qlen_plot y script de pruebas
- FRRouting y Miniedit
 - Instalación
 - Uso
- Algoritmos de control de congestión
 - Reno
 - Cubic
 - BBRv1
 - BBRv2
- Resultados
 - Escenario 1
 - Escenario 2
 - Escenario 3
 - Escenario 4
 - Escenario 5

Herramientas

Herramientas

- VM con BBRv2
- Mininet
- iperf
- captcp y tcpdump
- qlen_plot y script de pruebas

Herramientas - VM con BBRv2

- Distribución de linux
- Cantidad de RAM y núcleos
- Compilar kernel con BBRv2

Pasos

1. Instalar dependencias: `sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils libssl-dev bc flex libelf-dev bison`
2. Clonar desde github: `git clone -o google-bbr -b v2alpha --depth 1 https://github.com/google/bbr.git`

Herramientas - VM con BBRv2

3. Copiar `/boot/config-5.0.0-23-generic` o similar a `bbr/.config`
4. Ejecutar el comando `make`
5. Responder a las preguntas:
`BBR2 TCP (TCP_CONG_BBR2) [N/m/y/?] (NEW) y`
6. Terminar de instalar con:
`sudo make modules_install`
`sudo make install`

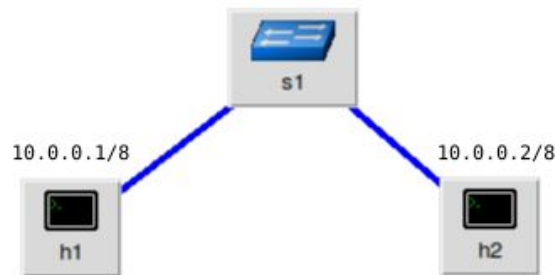


Herramientas

- VM con BBRv2
- **Mininet**
- iperf
- captcp y tcpdump
- qlen_plot y script de pruebas

Herramientas - Mininet

- Red virtual, similar a GNS3
- Abrir terminales en hosts
- Programar pruebas en Python 2
- **No** incluye interfaz gráfica (Miniedit)
- Procesos corren en anfitrión



```
git clone  
./mininet/util/install.sh
```

```
git://github.com/mininet/mininet
```

Herramientas - Mininet

`sudo mn`

- Se abre una consola con topología por defecto

`h1 ping 10.0.0.2`

- Están disponibles todos los programas de la PC anfitrión

```
mininet@traficoBBR:~$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> 
```

Herramientas - Mininet

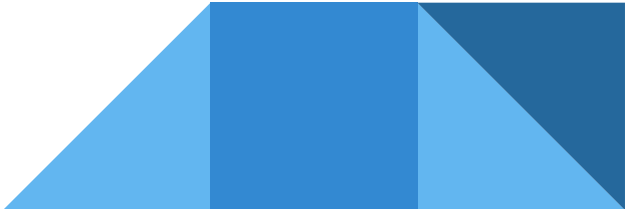
- Programar pruebas con python

```
net = Mininet(ipBase='10.0.0.0/8')

s1 = net.addSwitch('s1')
h1 = net.addHost('h1', ip='10.0.0.1', defaultRoute=None)
h2 = net.addHost('h2', ip='10.0.0.2', defaultRoute=None)

net.addLink(h1, s1)
net.addLink(h2, s1)
net.build()

h1.cmd('ping 10.0.0.2')
```

A decorative graphic in the bottom right corner consisting of several overlapping blue triangles and rectangles in different shades of blue.

Herramientas

- VM con BBRv2
- Mininet
- **iperf**
- **captcp** y **tcpdump**
- **qlen_plot** y **script de pruebas**

Herramientas - **iperf**

- Pruebas a grandes velocidades
- Cliente y servidor

```
sudo apt install iperf3
```

```
iperf3 -s
```

```
iperf3 -c "ip" -p "port"
```

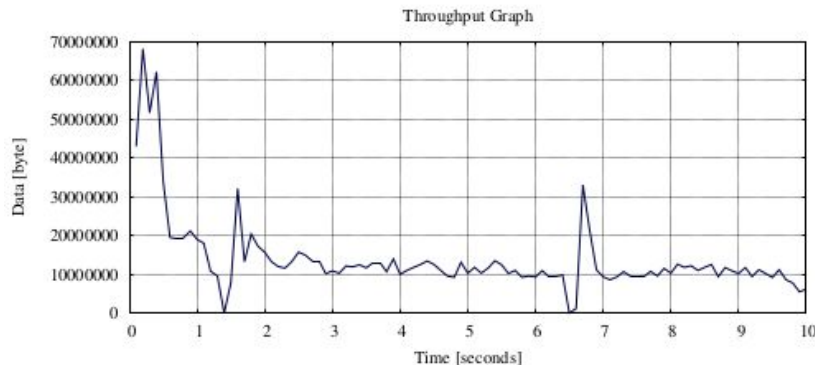


Herramientas

- VM con BBRv2
- Mininet
- iperf
- **captcp y tcpdump**
- qlen_plot y script de pruebas

Herramientas - **captcp** y **tcpdump**

- Capturar tráfico en la red
- Realizar gráficas
 - Throughput
 - RTT
 - Ventana de congestión
 - Inflight



Problemas:

- A veces no es posible ejecutar en el background
- Imprime demasiados mensajes

Herramientas - **captcp** y **tcpdump**

- Agregar en función `process_final()`

```
import signal  
signal.signal(signal.SIGINT, signal.default_int_handler)
```

- Instalación:

```
sudo apt install install python-dpkt python-numpy make gnuplot \  
    texlive-latex-extra texlive-font-utils mupdf  
git clone https://github.com/martinber/bbr2-mininet.git  
cd bbr2-mininet/captcp-mininet  
su -c 'make install'
```



Herramientas

- VM con BBRv2
- Mininet
- iperf
- captcp y tcpdump
- qlen_plot y script de pruebas

Herramientas - **qlen_plot**

- Medición de cantidad de paquetes en la cola TBF
- Modificación de código proveniente de Mininet
- Polling hacia `tc -s qdisc show dev`

```
sudo install -m 755 qlen_plot/qlen_plot.py /usr/local/bin/
```



Herramientas - **Script de pruebas**

- Basado en Mininet
- Escenarios con distintos parámetros de red
- Simular y graficar automáticamente

```
sudo python2 ./bbr2-mininet/tcp-mininet/tcp_mininet.py 2
```

(Mostrar en VM)



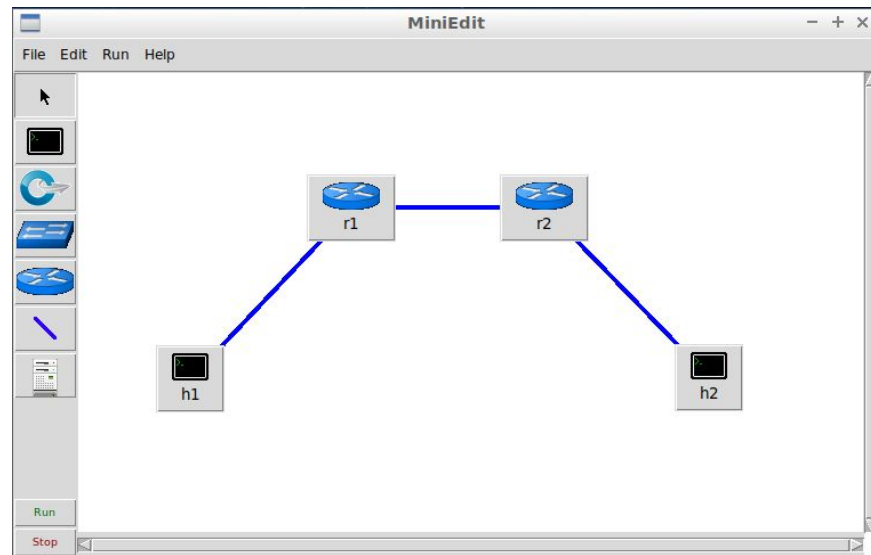
FRRouting y Miniedit

FRRouting y Miniedit

- Instalación
- Uso

FRRouting y Miniedit - Instalación

- Miniedit es un script con GUI
`/mininet/examples/miniedit.py`
- Cada host corre procesos en la PC anfitrión



FRRouting y Miniedit - **Instalación**

- Mininet+Miniedit no incluye routers, son hosts con ip_forwarding
- FRRouting no es compatible directamente, conflictos al crear múltiples instancias
- FRRouting usa Mininet para testing:
<https://github.com/FRRouting/frr/tree/master/tests/topotests/lib>
- Se modificó el código de Miniedit agregando código usado por el proyecto FRRouting



FRRouting y Miniedit - **Instalación**

1. Descargar código de Miniedit: `/mininet/examples/miniedit.py`
2. Descargar carpeta

<https://github.com/FRRouting/frr/tree/master/tests/topotests/lib>

3. Modificar `lib/topotest.py:775`

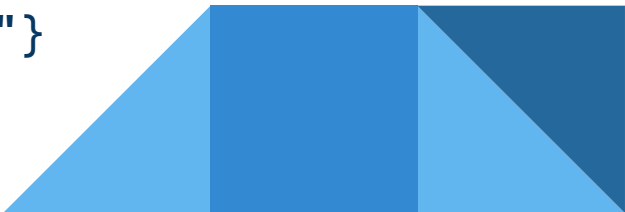
```
cur_test = os.environ["PYTEST_CURRENT_TEST"]  
cur_test = "miniedit_topology"
```



FRRouting y Miniedit - **Instalación**

4. Activar servicios en lib/topotest.py:790

```
self.daemons = {  
    "zebra": 1, "ripd": 0, "ripngd": 0, "ospfd": 0,  
    "ospf6d": 0, "isisd": 0, "bgpd": 1, "pimd": 0,  
    "ldpd": 0, "eigrpd": 0, "nhrrpd": 0, "staticd": 1,  
    "bfdd": 0, "sharpd": 0,  
}  
self.daemons_options =  
    {"zebra": "", "bgpd": "", "staticd": ""}
```

A decorative graphic in the bottom right corner consisting of several overlapping triangles and rectangles in various shades of blue, creating a modern, abstract design.

FRRouting y Miniedit - **Instalación**

5. Editar miniedit.py

- a. Borrar clase LegacyRouter
- b. `from lib.topotest import Router as LegacyRouter`

6. Editar miniedit.py:2772

```
elif 'LegacyRouter' in tags:  
newSwitch = net.addHost(name , cls=LegacyRouter)  
newSwitch = net.addHost(name , cls=LegacyRouter,  
    privateDirs=["/etc/frr", "/var/run/frr", "/var/log"])
```

FRRouting y Miniedit - **Instalación**

7. Agregar en `miniedit.py:3033`

```
if 'LegacyRouter' in tags:  
    self.net.get(name).startRouter( [] )  
    info(name + ' ' )
```



FRRouting y Miniedit

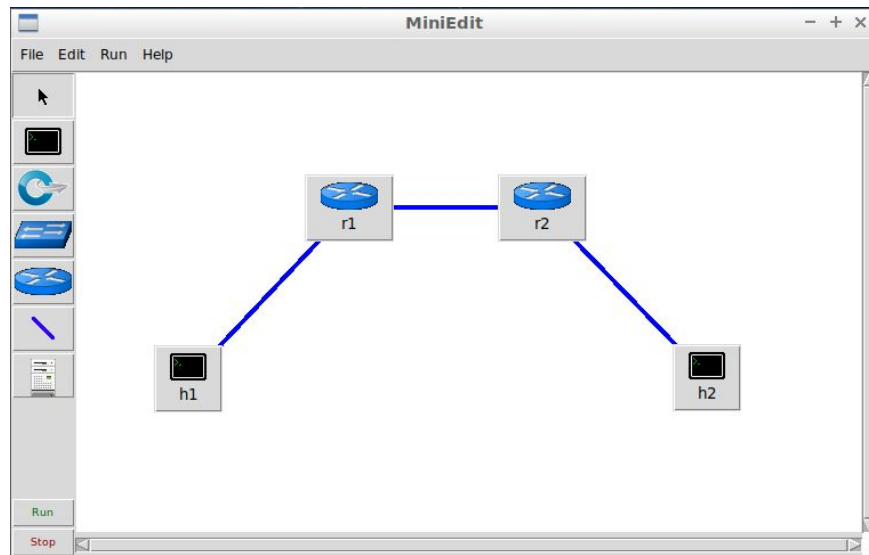
- Instalación
- Uso

FRRouting y Miniedit - **Uso**

```
sudo ./miniedit.py
```

- Agregar dispositivos y poner “Run”
- Borrar IPs configuradas automáticamente
- Abrir terminal en cada router:

```
echo 1 > /proc/sys/net/ipv4/ip_forward  
vtysh
```



FRRouting y Miniedit - Uso

```
configure terminal
interface r1-eth1
ip address 10.0.0.1/16
exit
```

```
router bgp 100
network 10.0.0.0/16
neighbor 10.9.0.2 remote-as 200
```

...

```
"Host: r1"
root@traficoBBR:/Desktop/bbr2-mininet# vtysh
Hello, this is FRRouting (version 7.3).
Copyright 1996-2005 Kunihiko Ishiguro, et al.

traficoBBR# show interface brief
Interface      Status VRF      Addresses
-----
lo             up      default  127.0.0.1
r1-eth0        up      default  10.9.0.1/16
r1-eth1        up      default  10.0.0.1/16

traficoBBR# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - FRR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C* 10.0.0.0/16 is directly connected, r1-eth1, 00:19:26
B* 10.1.0.0/16 [20/0] via 10.9.0.2, r1-eth0, 00:12:41
C* 10.9.0.0/16 is directly connected, r1-eth0, 00:19:57
traficoBBR#

"Host: r2"
root@traficoBBR:/Desktop/bbr2-mininet# vtysh
Hello, this is FRRouting (version 7.3).
Copyright 1996-2005 Kunihiko Ishiguro, et al.

traficoBBR# show interface brief
Interface      Status VRF      Addresses
-----
lo             up      default  127.0.0.1
r2-eth0        up      default  10.1.0.1/16
r2-eth1        up      default  10.9.0.2/16

traficoBBR# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - FRR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

B* 10.0.0.0/16 [20/0] via 10.9.0.1, r2-eth1, 00:13:06
C* 10.1.0.0/16 is directly connected, r2-eth0, 00:19:43
C* 10.9.0.0/16 is directly connected, r2-eth1, 00:19:53
traficoBBR#

"Host: h1"
root@traficoBBR:/Desktop/bbr2-mininet# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: h1-eth0@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 06:1c:0c:0a:0f:4c:5 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.0.10/16 scope global h1-eth0
        valid_lft forever preferred_lft forever
root@traficoBBR:/Desktop/bbr2-mininet# ping 10.1.0.1 -c 3
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data:
64 bytes from 10.1.0.1: icmp_seq=1 ttl=62 time=0.071 ms
64 bytes from 10.1.0.1: icmp_seq=2 ttl=62 time=0.074 ms
64 bytes from 10.1.0.1: icmp_seq=3 ttl=62 time=0.503 ms

--- 10.1.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2065ms
rtt min/avg/max/ndev = 0.071/0.218/0.503/0.205 ms
root@traficoBBR:/Desktop/bbr2-mininet#

"Host: h2"
root@traficoBBR:/Desktop/bbr2-mininet# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: h2-eth0@if2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 56:1c:0c:0a:0f:3e:397 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.0.1/16 scope global h2-eth0
        valid_lft forever preferred_lft forever
root@traficoBBR:/Desktop/bbr2-mininet# ping 10.0.0.1 -c 3
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_seq=1 ttl=63 time=0.047 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=63 time=0.125 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=63 time=0.070 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2102ms
rtt min/avg/max/ndev = 0.047/0.080/0.125/0.034 ms
root@traficoBBR:/Desktop/bbr2-mininet#
```

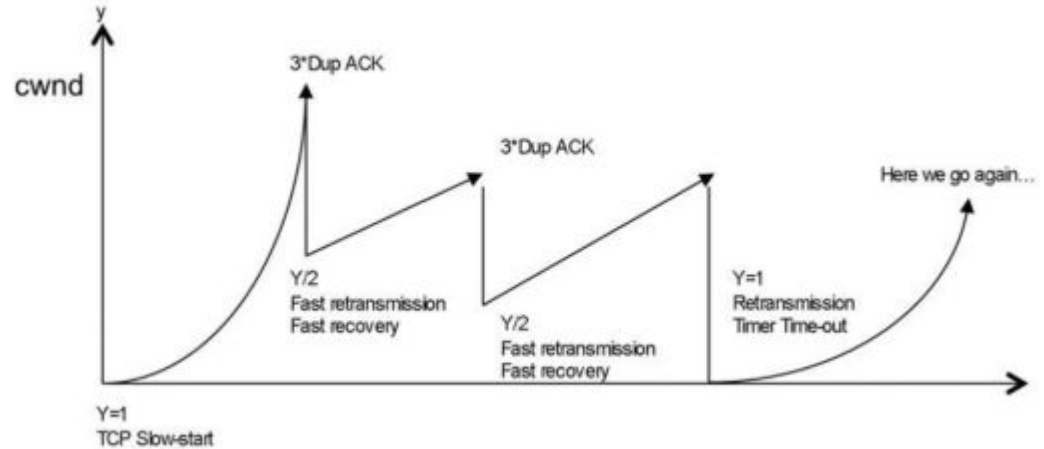
Algoritmos de control de congestión

Algoritmos de control de congestión

- Reno
- Cubic
- BBRv1
- BBRv2

Control de congestión - Reno

- Sensible a las pérdidas
- Slow start
- Congestion avoidance
- Fast recovery
- 3 ACK's duplicados
- Timeout

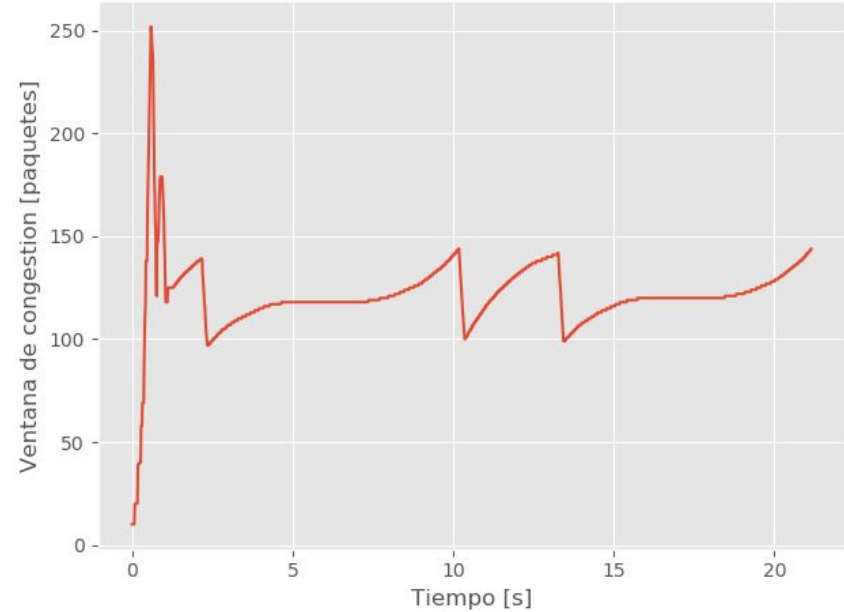


Algoritmos de control de congestión

- Reno
- Cubic
- BBRv1
- BBRv2

Control de congestión - Cubic

- Sensible a las pérdidas
- Mejor aprovechamiento del ancho de banda que Reno
- Establecimiento del umbral

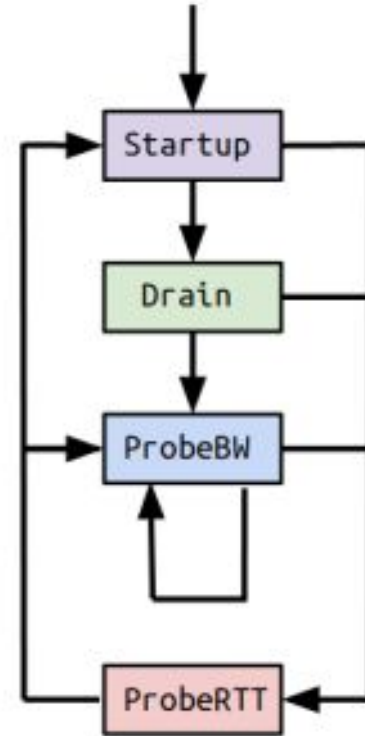


Algoritmos de control de congestión

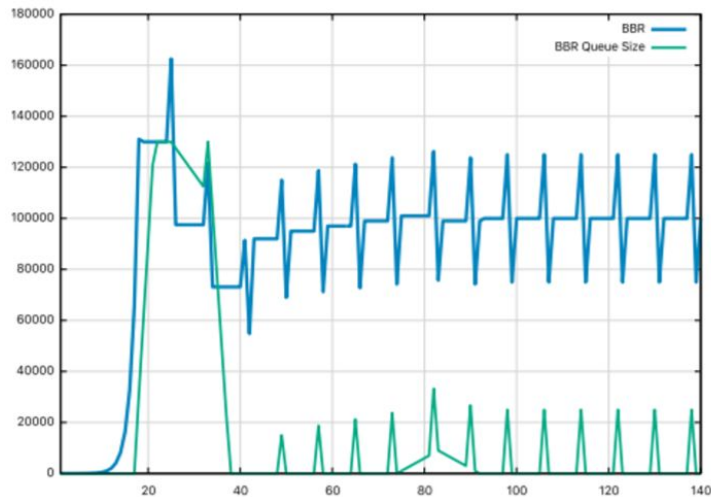
- Reno
- Cubic
- **BBRv1**
- BBRv2

Control de congestión - **BBRv1**

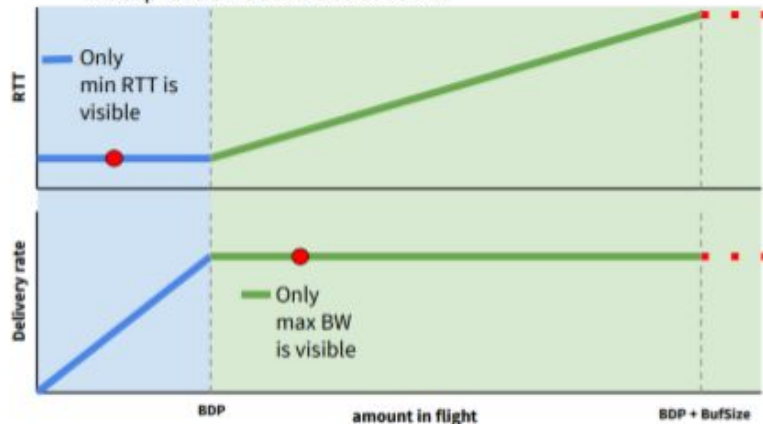
- Sensible al retardo de paquetes
- Drain
- Probe_RTT
- Probe_BW



BBRv1



But to see both max BW and min RTT,
must probe on both sides of BDP...

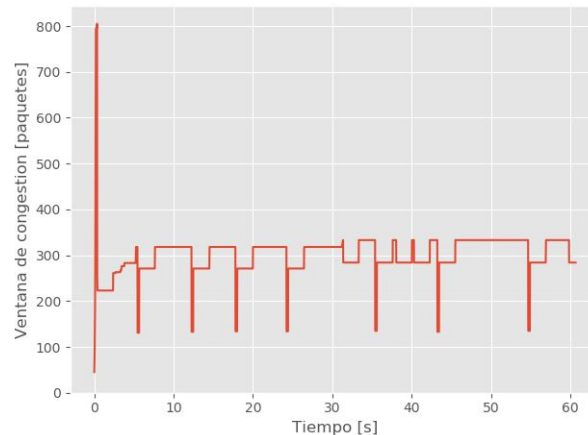


Algoritmos de control de congestión

- Reno
- Cubic
- BBRv1
- BBRv2

Control de congestión - BBRv2

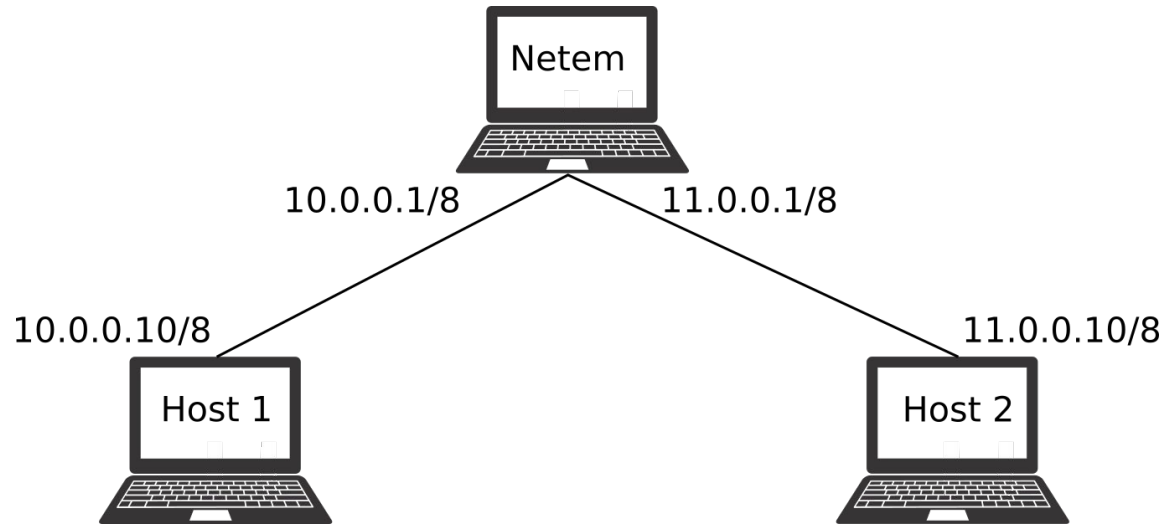
- Sensible al retardo de paquetes
- Tiene en cuenta pérdidas
- ECN
- Aggregation



Resultados

Escenarios

- Simulaciones automáticas realizadas con script de mininet



Resultados - **Escenario 1**

Observar comportamiento general de los algoritmos

- Ancho de banda: 10 Mbps
- Delay: 100 ms
- Pérdidas: 0%
- Latencia de buffer: 50 ms
- Duración de prueba: 20s



TCP Reno

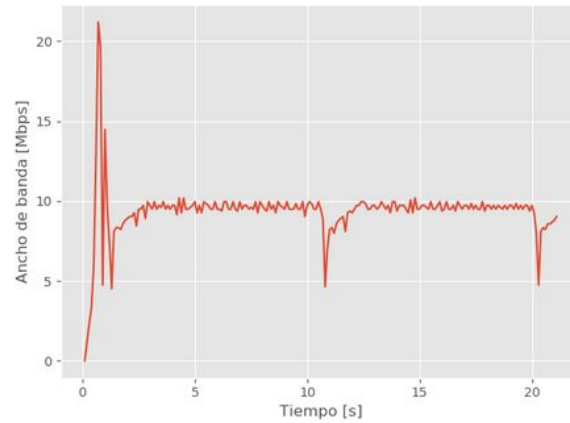


Figura 11.1: Throughput

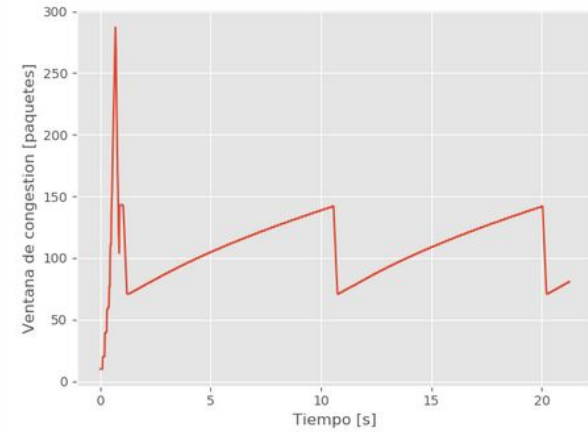


Figura 11.2: Tamaño de ventana de congestión

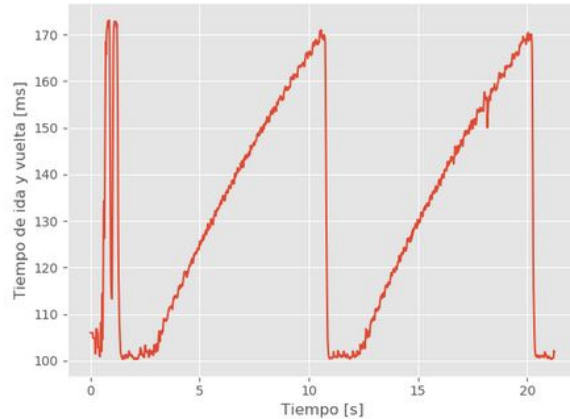


Figura 11.3: Round Trip Time

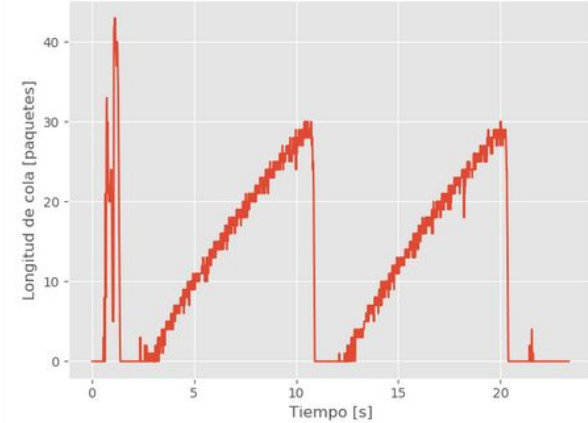


Figura 11.4: Longitud de cola

TCP Cubic

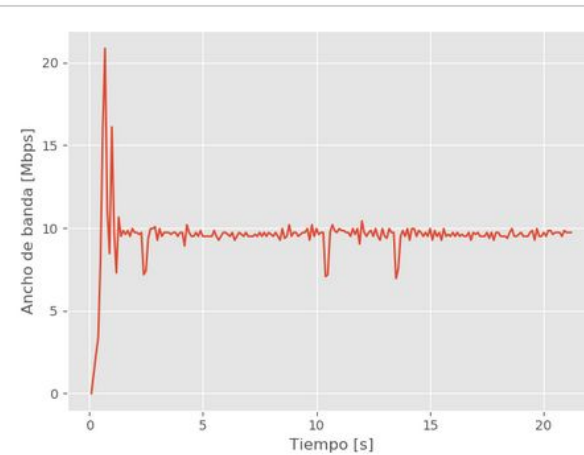


Figura 12.1: Throughput

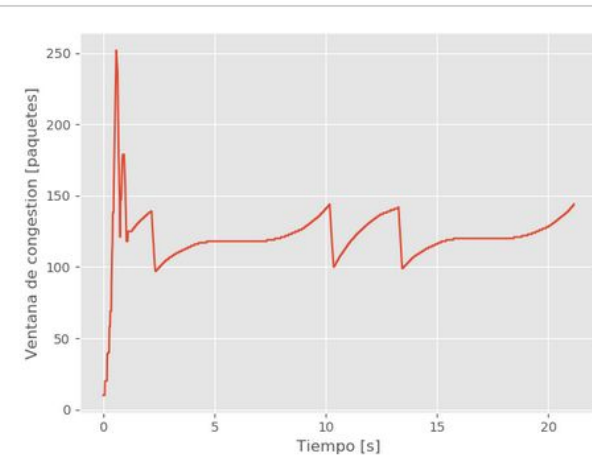


Figura 12.2: Tamaño de ventana de congestión

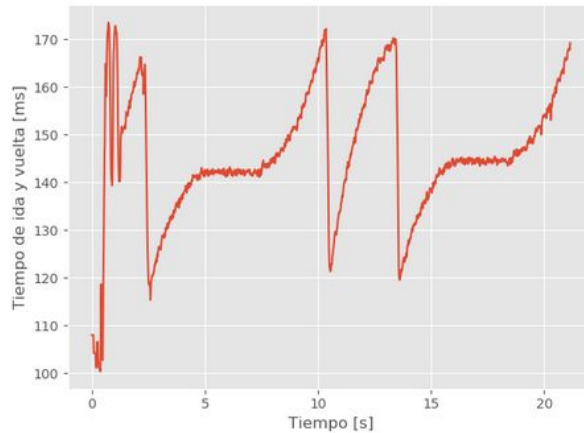


Figura 12.3: Round Trip Time

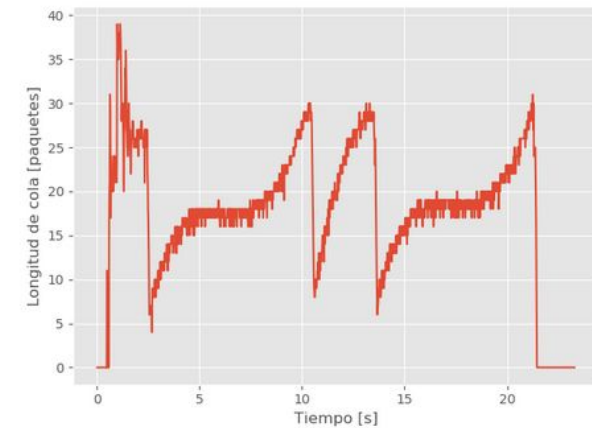


Figura 12.4: Longitud de cola

TCP BBRv1

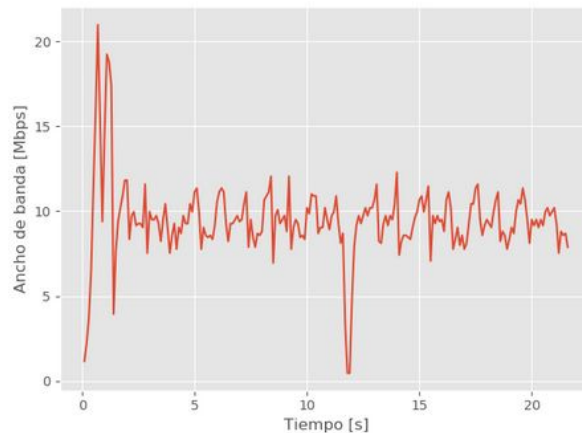


Figura 13.1: Throughput

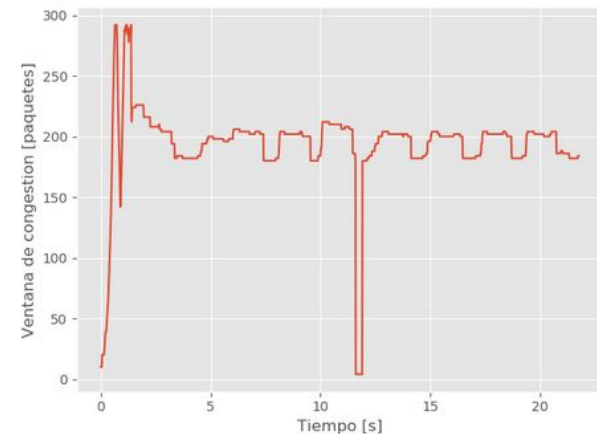


Figura 13.2: Tamaño de ventana de congestión

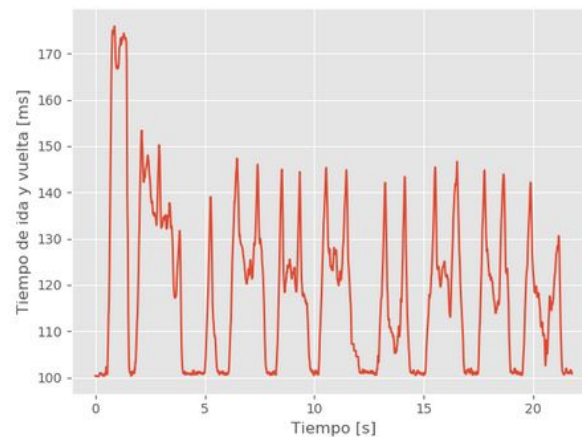


Figura 13.3: Round Trip Time

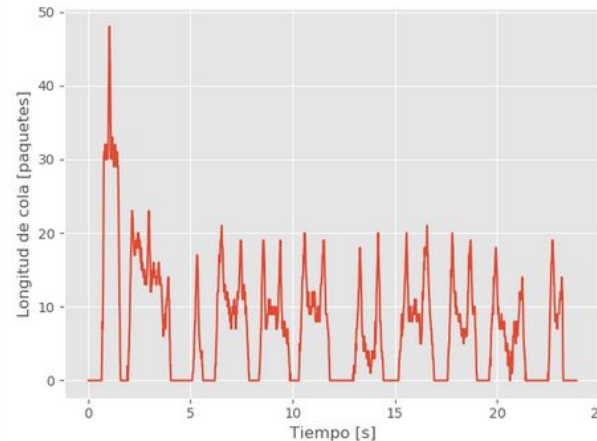


Figura 13.4: Longitud de cola

TCP BBRv2

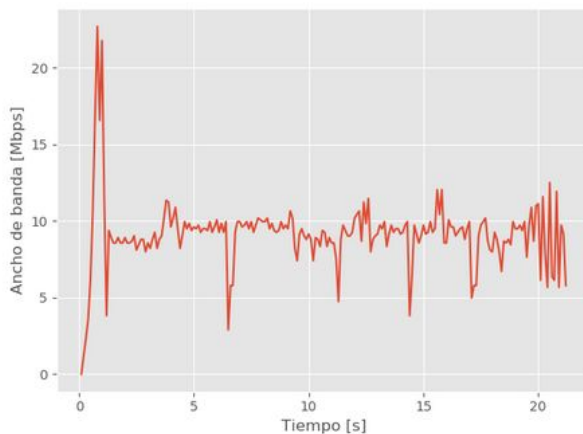


Figura 14.1: Throughput

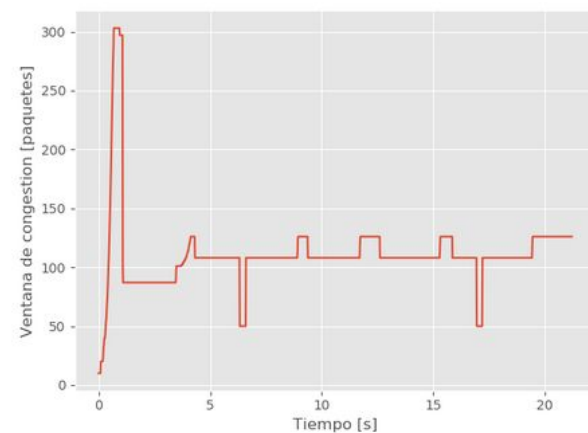


Figura 14.2: Tamaño de ventana de congestión

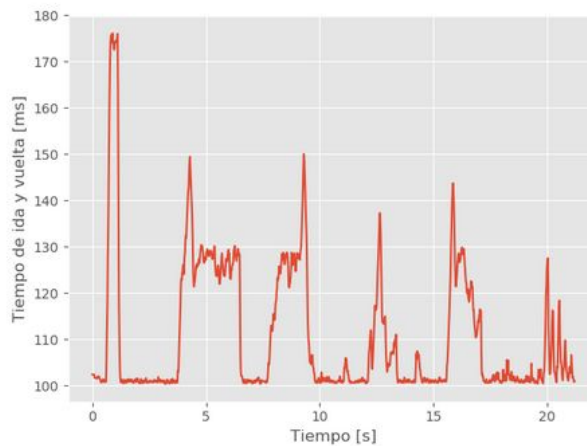


Figura 14.3: Round Trip Time

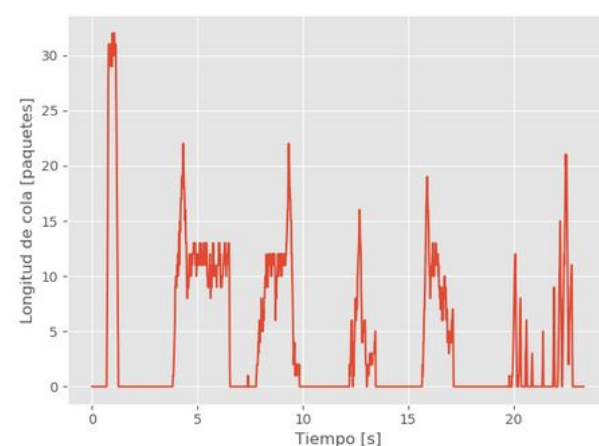


Figura 14.4: Longitud de cola

Resultados - **Escenario 2**

Analizar diferencias entre BBRv1 y BBRv2

- Ancho de banda: 100 Mbps
- Delay: 30 ms
- Pérdidas: 0%
- Latencia de buffer: 30 ms
- Duración de prueba: 60s



TCP
BBRv1

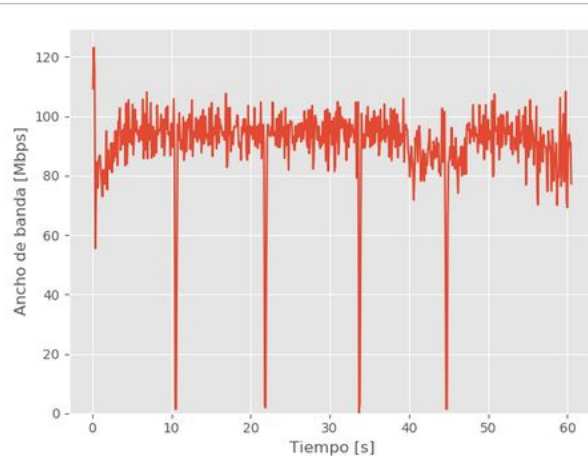


Figura 15.1: Throughput

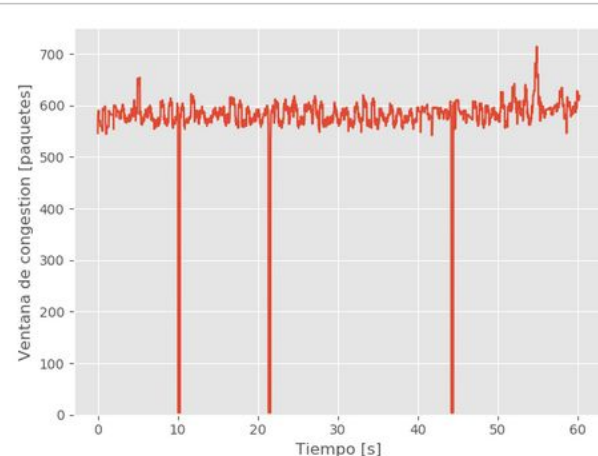


Figura 15.2: Tamaño de ventana de

TCP
BBRv2

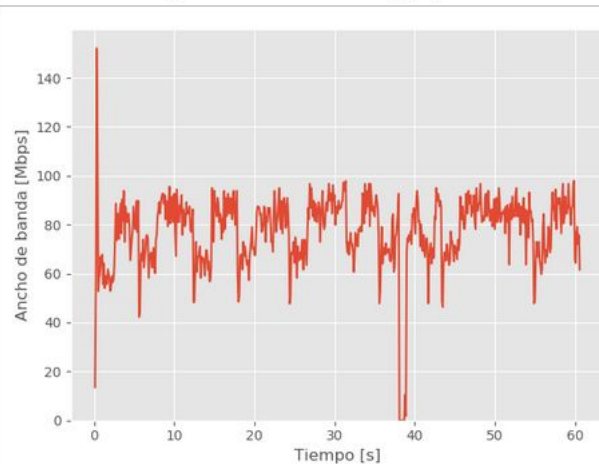


Figura 16.1: Throughput

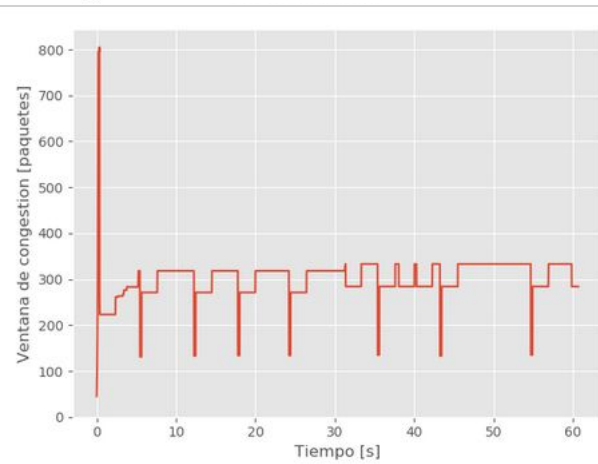


Figura 16.2: Tamaño de ventana de congestión

TCP
BBRv1

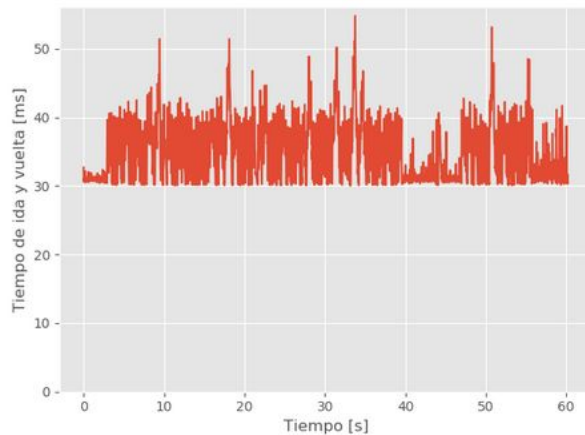


Figura 15.3: Round Trip Time

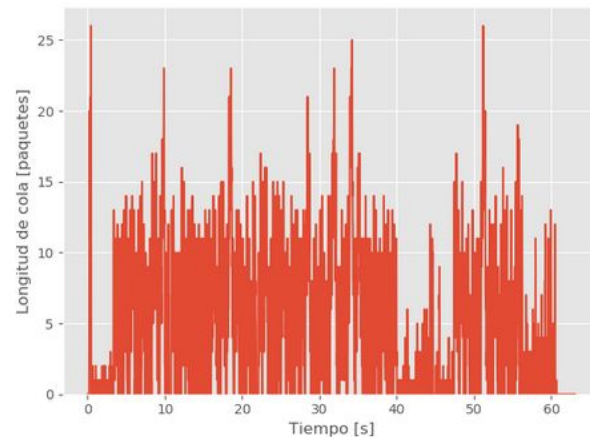


Figura 15.4: Longitud de cola

TCP
BBRv2

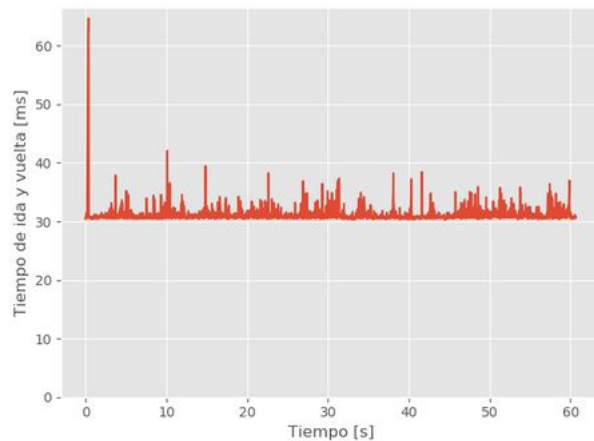


Figura 16.3: Round Trip Time

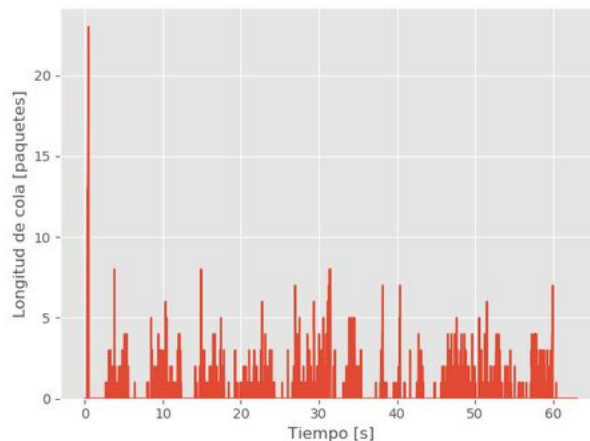


Figura 16.4: Longitud de cola

Resultados - **Escenario 3**

Observar comportamiento ante las pérdidas

- Ancho de banda: 100 Mbps
- Delay: 100 ms
- Pérdidas: 0.5%, correlación 25%
- Latencia de buffer: 50 ms
- Duración de prueba: 10s



TCP Reno

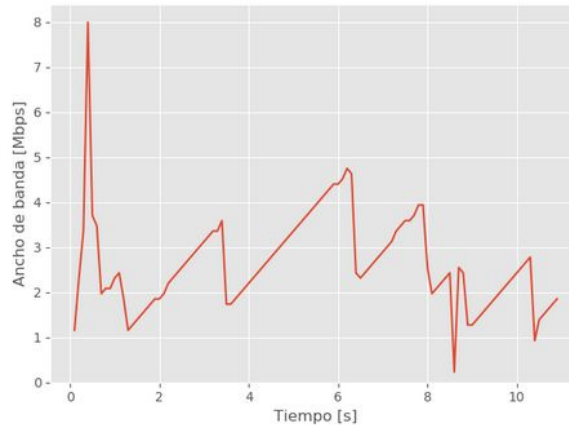


Figura 17.1: Throughput

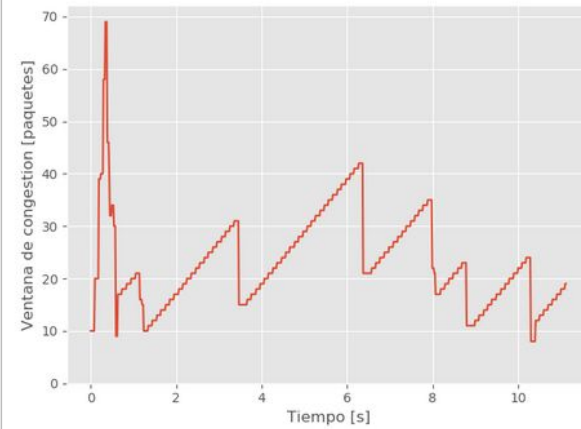


Figura 17.2: Tamaño de ventana de congestión

TCP Cubic

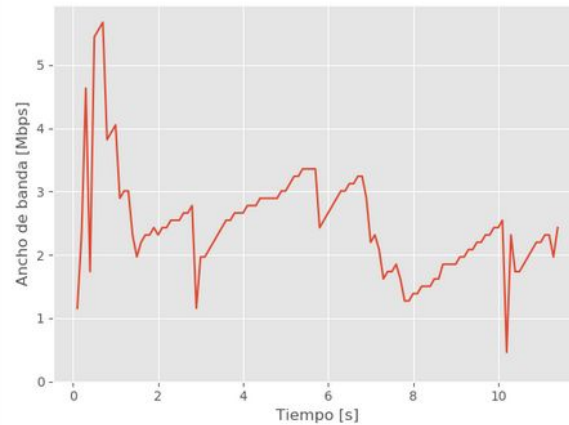


Figura 18.1: Throughput

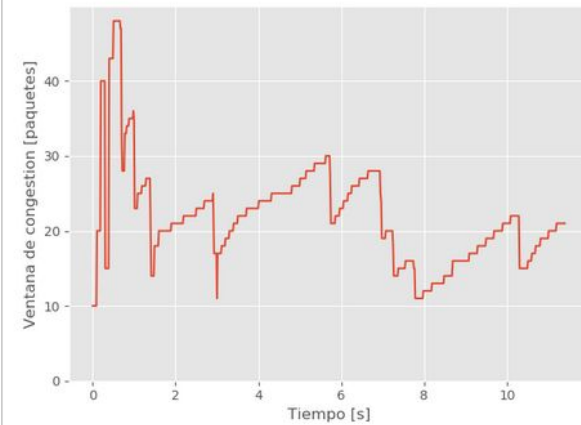


Figura 18.2: Tamaño de ventana de congestión

TCP BBRv1

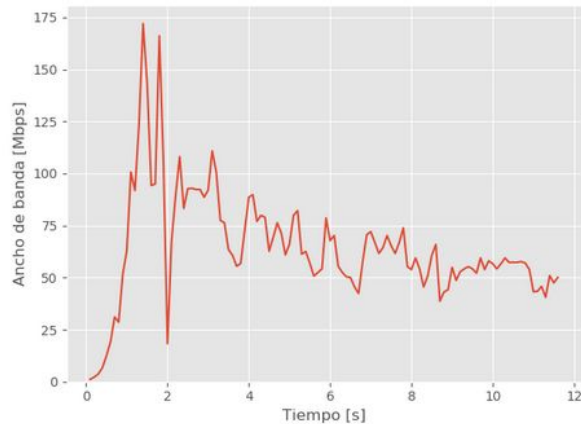


Figura 19.1: Throughput

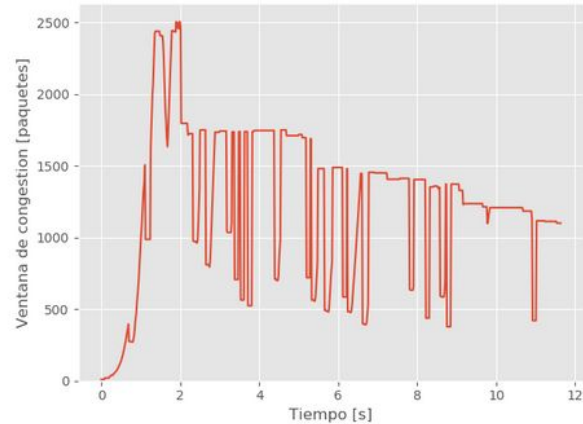


Figura 19.2: Tamaño de ventana de congestión

TCP BBRv2

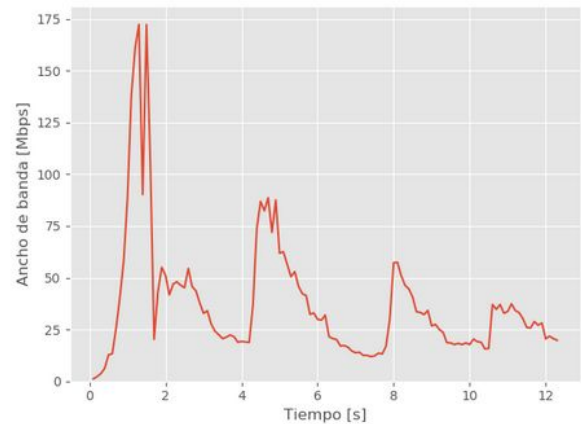


Figura 20.1: Throughput

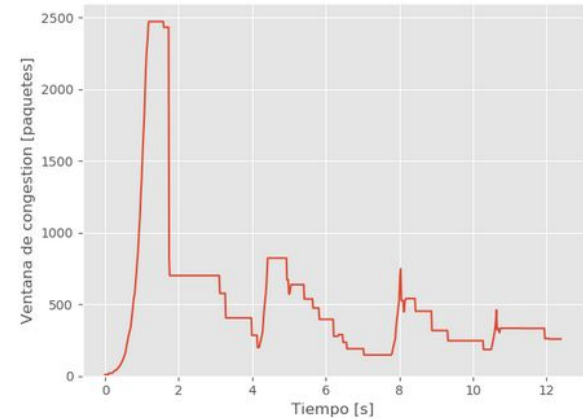


Figura 20.2: Tamaño de ventana de congestión

Resultados - **Escenario 4**

Problemas de rendimiento al trabajar con pocos recursos

- Ancho de banda: 100 Mbps
- Delay: 50 ms
- Pérdidas: 0%
- Latencia de buffer: 50 ms
- Duración de prueba: 60s



TCP
BBRv1

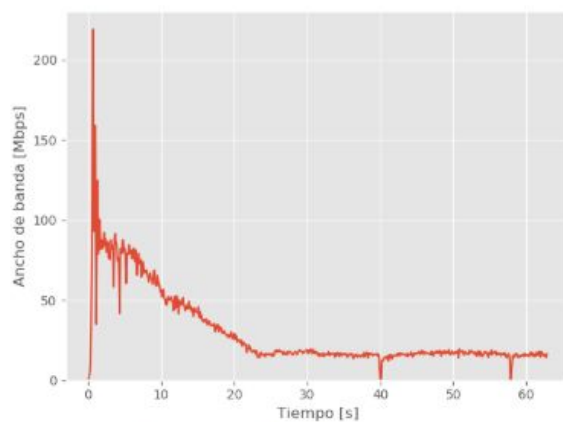


Figura 22.1: Throughput

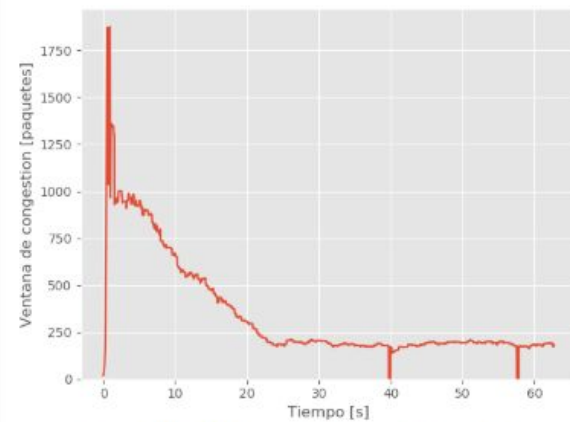


Figura 22.2: Tamaño de ventana de congestión

TCP
BBRv2

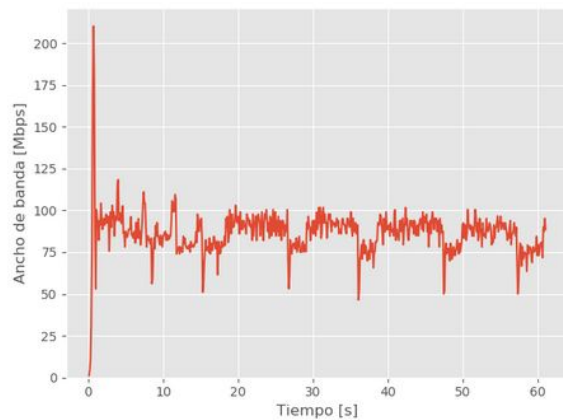


Figura 23.1: Throughput

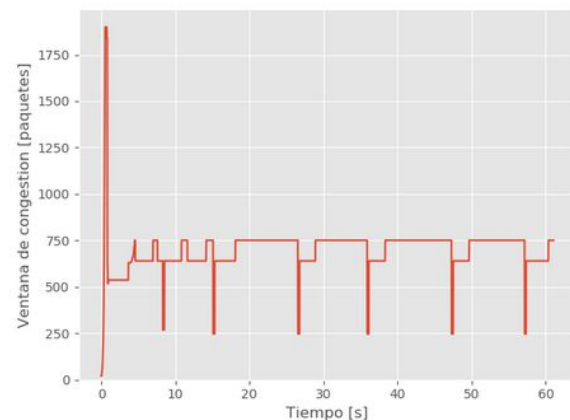


Figura 23.2: Tamaño de ventana de congestión

Resultados - **Escenario 5**

Redes con buffers pequeños

- Ancho de banda: 100 Mbps
- Delay: 200 ms
- Pérdidas: 0%
- Tamaño de buffer: 200 Kbytes
- Duración de prueba: 60s



TCP Reno

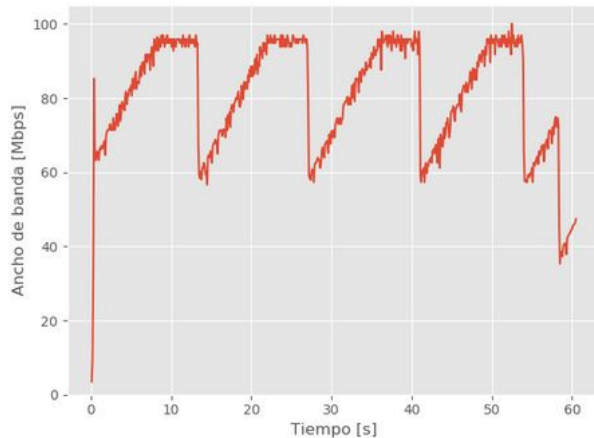


Figura 24.1: Throughput

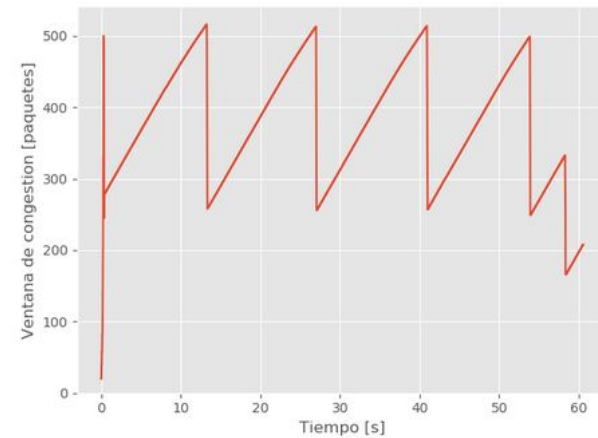


Figura 24.2: Tamaño de ventana de congestión

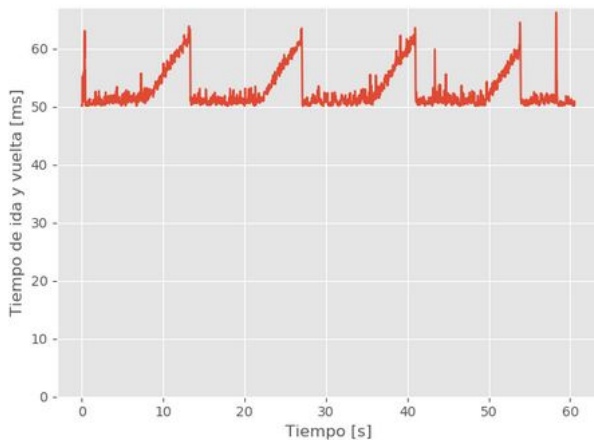


Figura 24.3: Round Trip Time

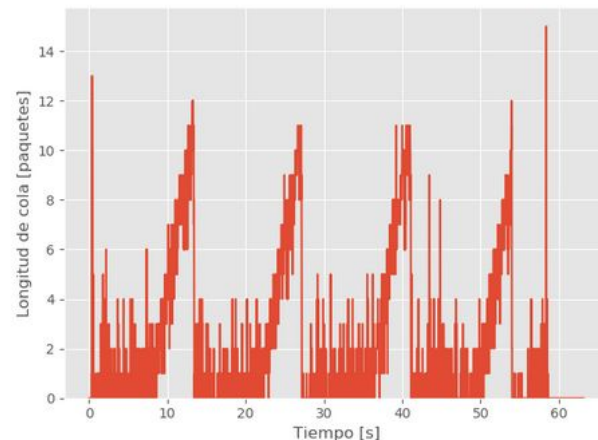


Figura 24.4: Longitud de cola

TCP BBRv1

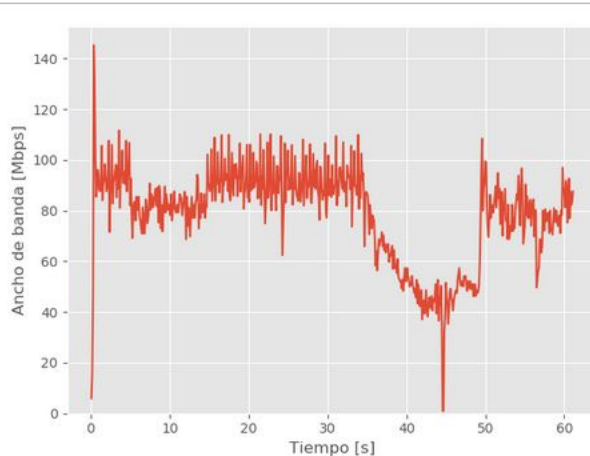


Figura 25.1: Throughput

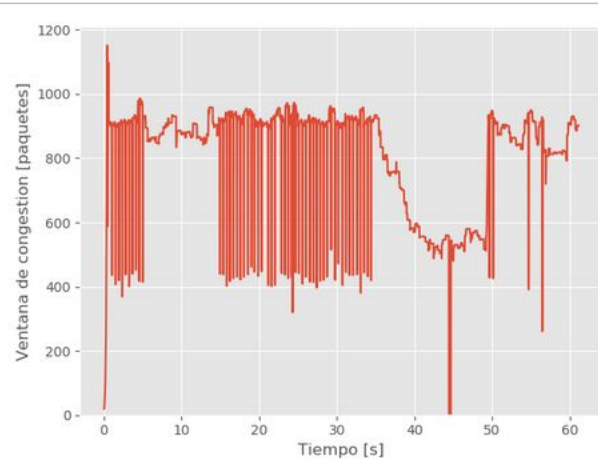


Figura 25.2: Tamaño de ventana de congestión

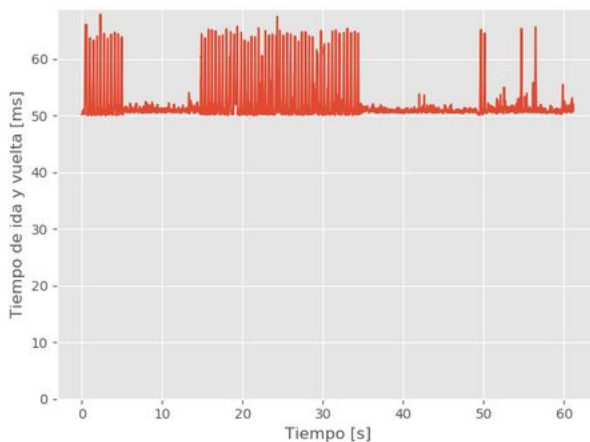


Figura 25.3: Round Trip Time

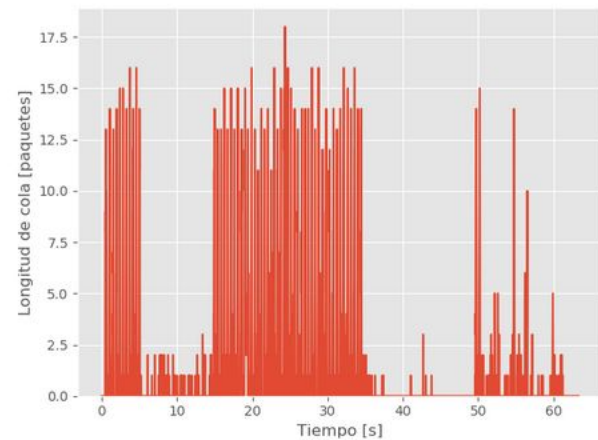


Figura 25.4: Longitud de cola

TCP BBRv2

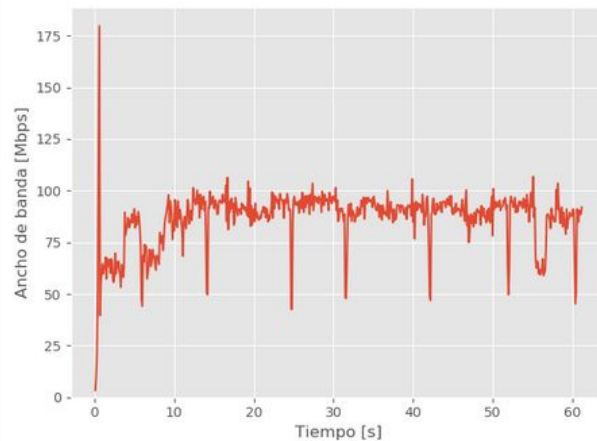


Figura 26.1: Throughput

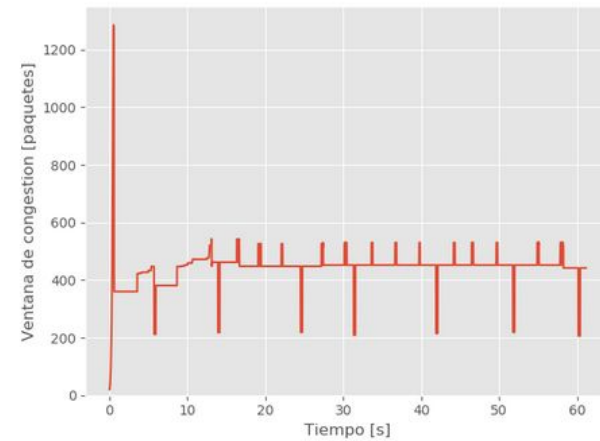


Figura 26.2: Tamaño de ventana de congestión

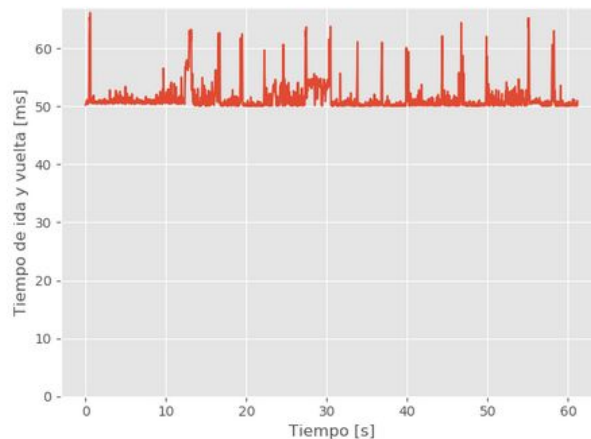


Figura 26.3: Round Trip Time

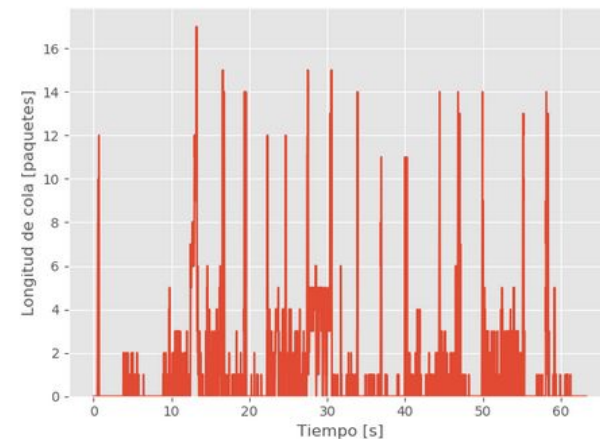


Figura 26.4: Longitud de cola

Conclusión

Conclusión

- Entorno de trabajo con Mininet, BBRv2, FRRouting, captcp, Miniedit, etc.
- Máquina virtual y código fuente para trabajos posteriores
- Análisis de diferentes algoritmos de control de congestión y simulación de su comportamiento

Trabajo futuro:

- Análisis de múltiples flujos TCP y observación de ECN





Muchas gracias!