Análisis del protocolo MPTCP en plataformas Linux

Sebastián Álvarez Méndez
Grado en Ingeniería Telemática
100275783@alumnos.uc3m.es

Índice

1. Introducción
2. Estado del arte
3. Mediciones y resultados
4. Conclusiones

1 Introducción

1. Introducción

1.1 Motivación

Estudio y análisis del protocolo MPTCP

- Protocolo novedoso
- Evolución del protocolo TCP
- Diseñado para optimizar recursos de red y aumentar fiabilidad
- Mejor adaptado a elementos de red modernos
- Implementación para sistemas Linux activa y estable

1. Introducción

1.2 Objetivos

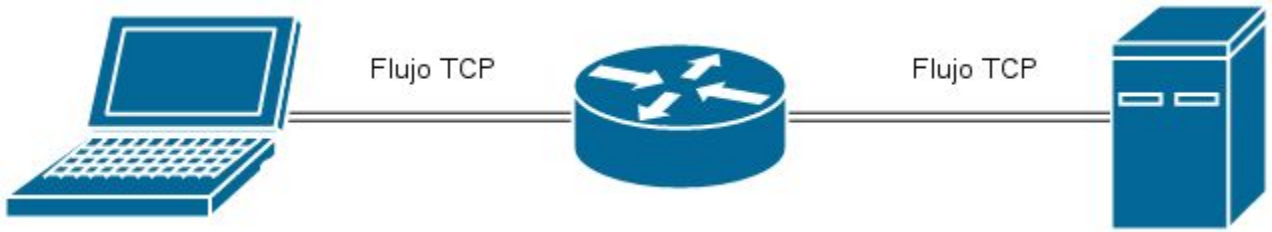
- Estudio de en un entorno de red real
- Comparar rendimiento de TCP y MPTCP
- Estudiar comportamiento ante pérdidas de conexión
- Validar la implementación de MPTCP para plataformas Linux

2 Estado del arte

2. Estado del arte

2.1 Protocolo TCP

- Protocolo de nivel de transporte utilizado para la transferencia de datos.
- Parte fundamental de Internet.
- Transmisión de flujo de datos.
- Orientado a conexión.
- Servicio fiable



2. Estado del arte

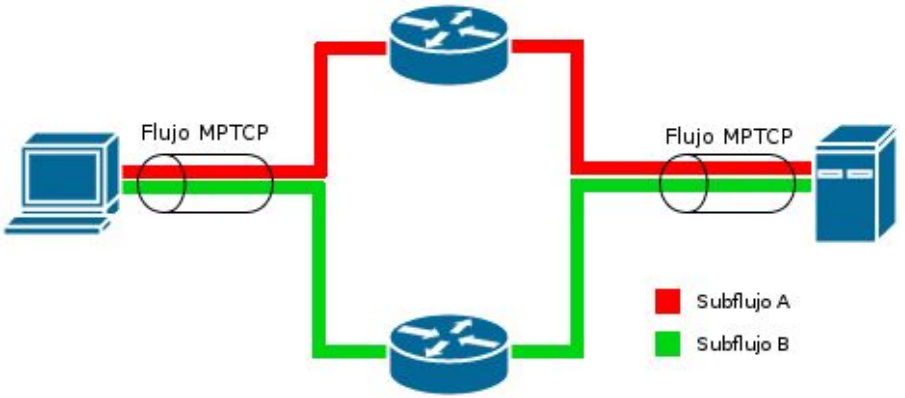
2.2 Protocolo MPTCP

- Extensión del protocolo TCP
- Permite utilizar múltiples interfaces y subflujos para la transmisión de datos
  - Incremento de eficiencia
  - Mayor robustez de la conexión
- Restricciones de diseño:
  - Debe funcionar a través de dispositivos intermedios (firewalls, NATs, proxys)
  - Debe ser transparente para las aplicaciones
  - Debe ser capaz de funcionar en modo TCP con hosts que no lo soporten

2. Estado del arte

2.2.1 Funcionamiento de MPTCP

- Un único flujo de datos distribuido en múltiples subflujos
- Una conexión TCP para cada subflujo
- Datos transmitidos independientemente en cada subflujo y unidos en destino



2.2.2 Ventajas y problemas

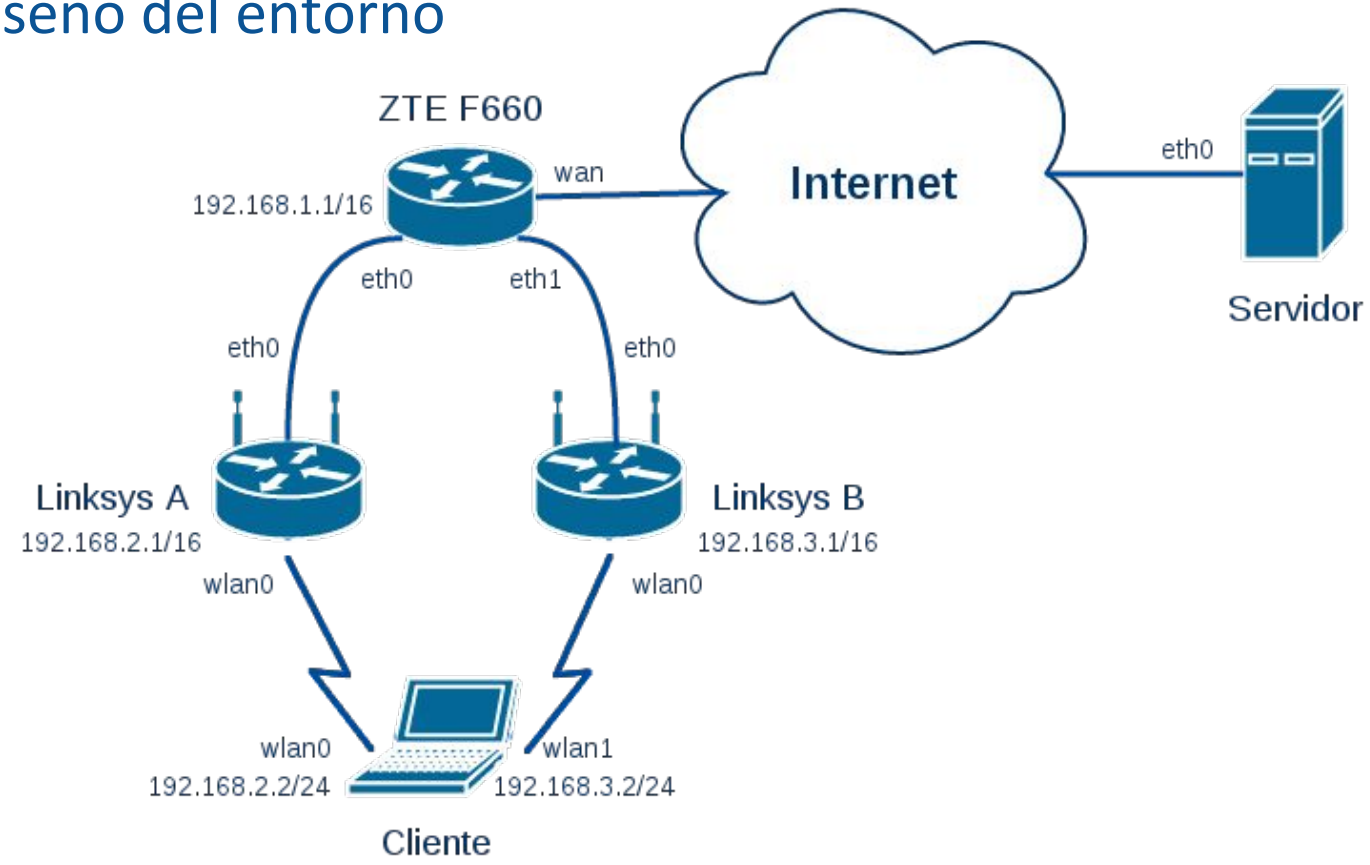
- Ventajas
  - Aumento de throughput
  - Optimización de recursos de red
  - Reducción de pérdidas
  - Minimización de downtime ante pérdida de conexión

- Problemas
  - Elementos de red intermedios (NATs, proxys, firewalls)

3 Mediciones y resultados

3. Mediciones y resultados

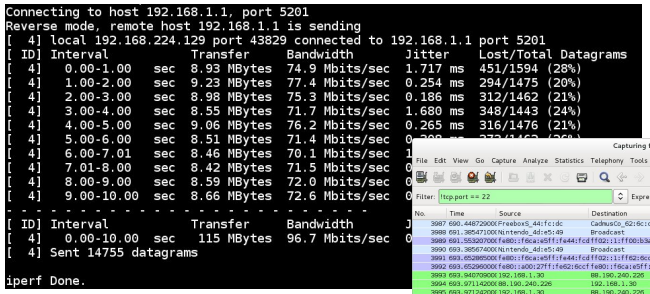
3.1 Diseño del entorno

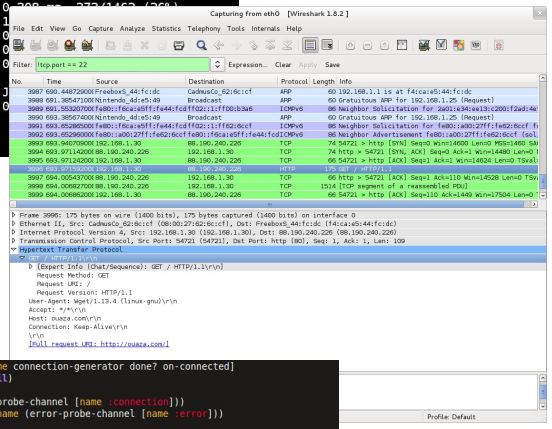


3. Mediciones y resultados

3.2 Herramientas

- iperf3
- hping3
- Wireshark
- iproute
- Netimpair
- Herramientas propias en Python y Bash






```
(defn- connection-loop [name connection-generator done? on-connected]
  (let [connection (atom nil)
        delay (atom 0)
        probe (when name (probe-channel [name :connection]))
        error-probe (when name (error-probe-channel [name :error]))
        trace #(when probe
                 (enqueue probe
                   (assoc %
                     :timestamp (System/currentTimeMillis))))]
    (run-pipeline nil
      {:error-handler (fn [err]
                        (when error-probe (enqueue error-probe err))
                        (error @connection err)
                        (swap! delay incr-delay)
                        (restart))}

      ;; check if we want to try to reconnect
      (fn [ ]
        (if @done?
          (do
            (reset! connection :lamina/deactivated!)
            (complete nil))
          ;; if so, reset the connection to a fresh result-channel
          (do
            (reset! connection (result-channel))
            nil)))

      ;; wait the proscribed duration
      (wait-stage @delay)
```

3. Mediciones y resultados

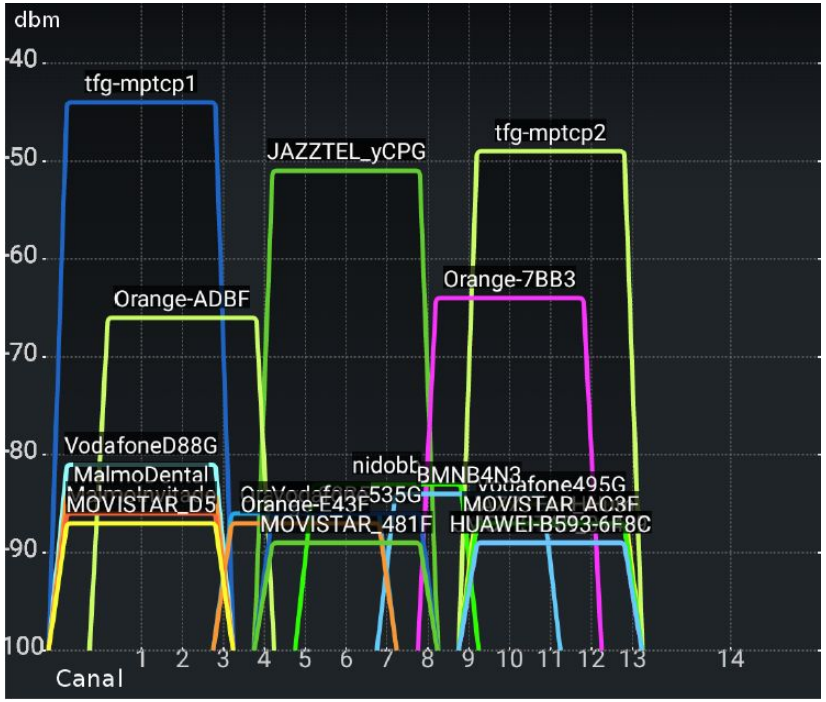
3.3 Metodología

- Métricas
  - Throughput y delay: máximo, mínimo, media, mediana.
- Dos interfaces inalámbricas
  - *wlan0* y *wlan1*
  - Canales 1 y 11
  - 802.11 b y 802.11 g
- 12 mediciones
  - 20 segundos
  - Eliminación de valores extremos
- Medición de tráfico de bajada
- Configuración de interfaces:
  - *wlan0* con TCP
  - *wlan0* con MPTCP
  - *wlan1* con TCP
  - *wlan1* con MPTCP
  - Ambas con MPTCP

3. Mediciones y resultados

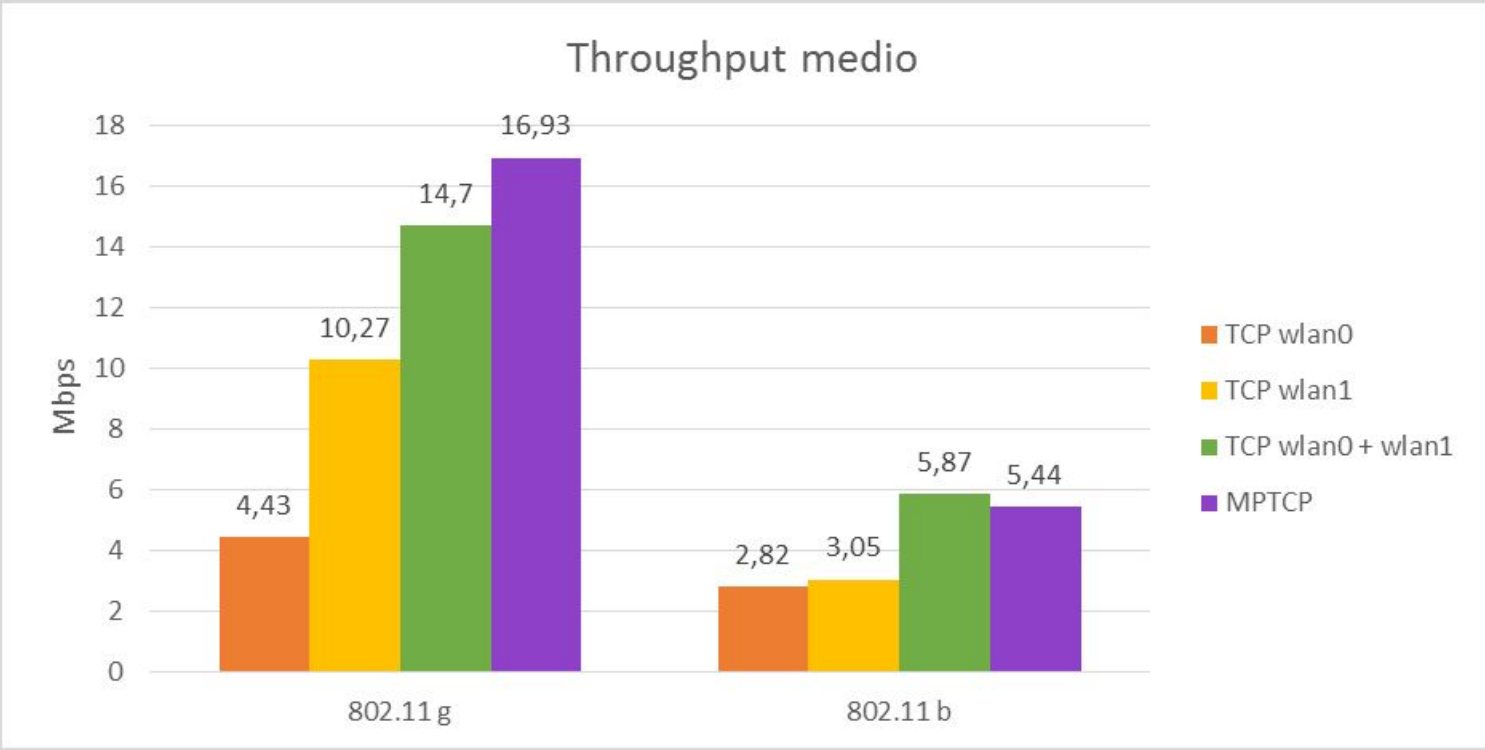
3.4 Pruebas iniciales

- 802.11 g
  - Interferencias
- 802.11 a
  - Inestable
- 802.11 b
  - Menor variabilidad



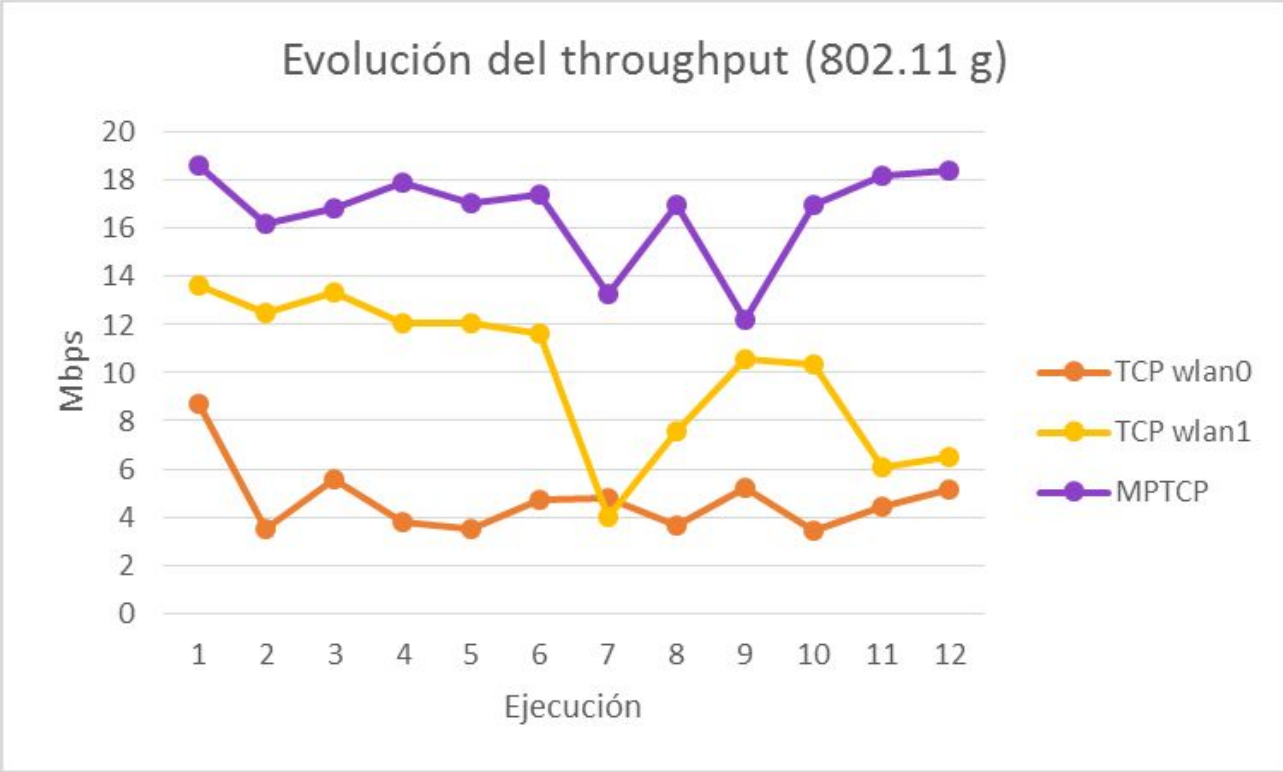
3. Mediciones y resultados

3.5 Resultados



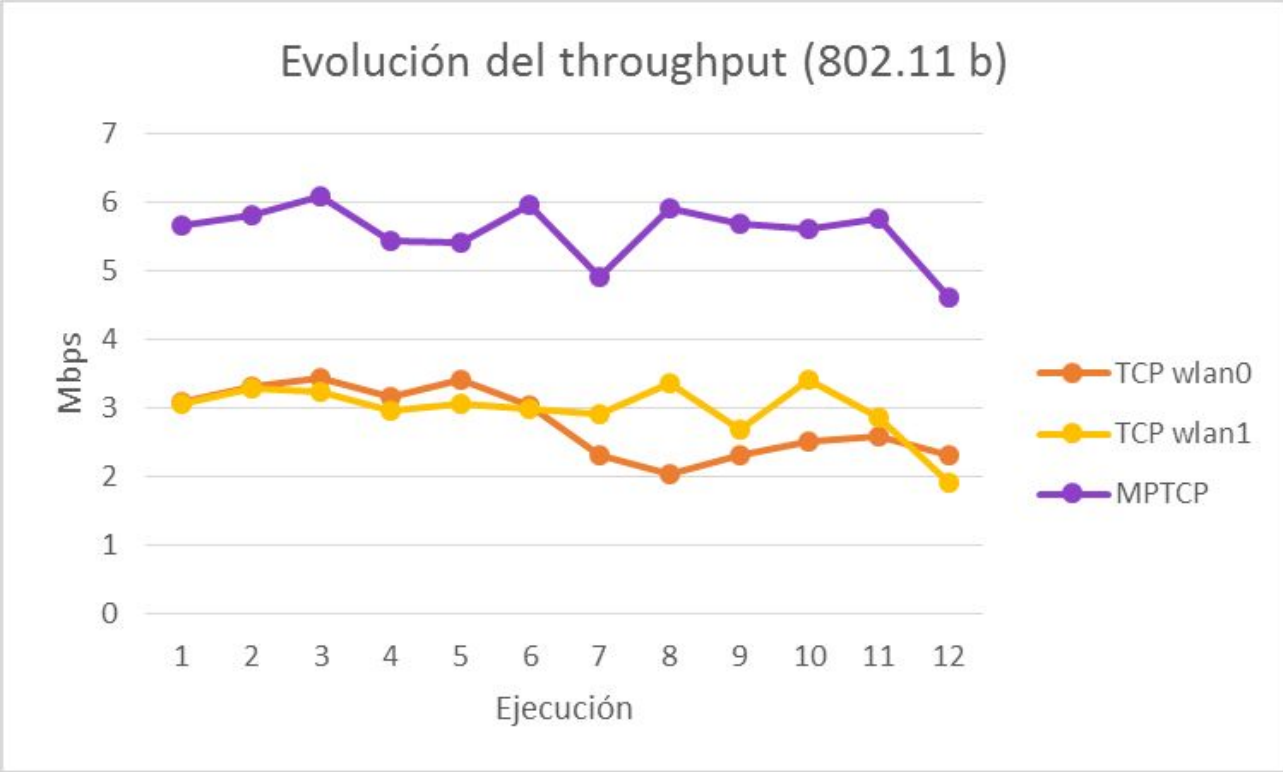
3. Mediciones y resultados

3.5 Resultados



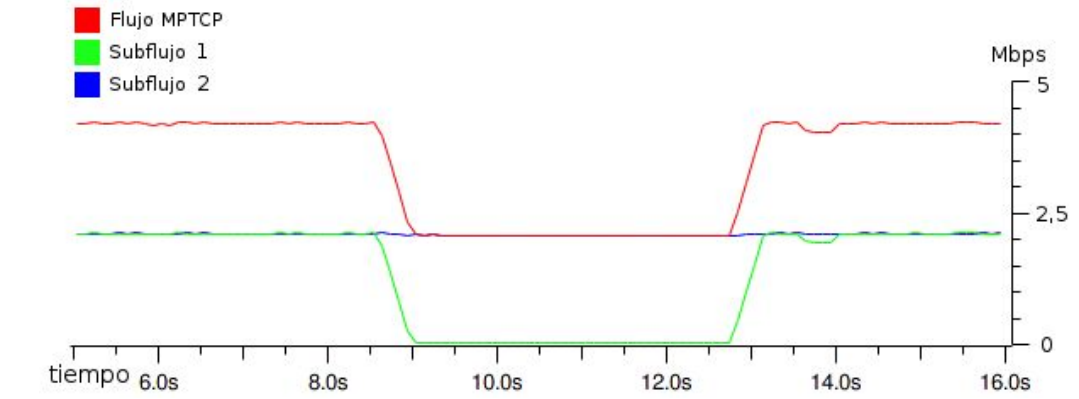
3. Mediciones y resultados

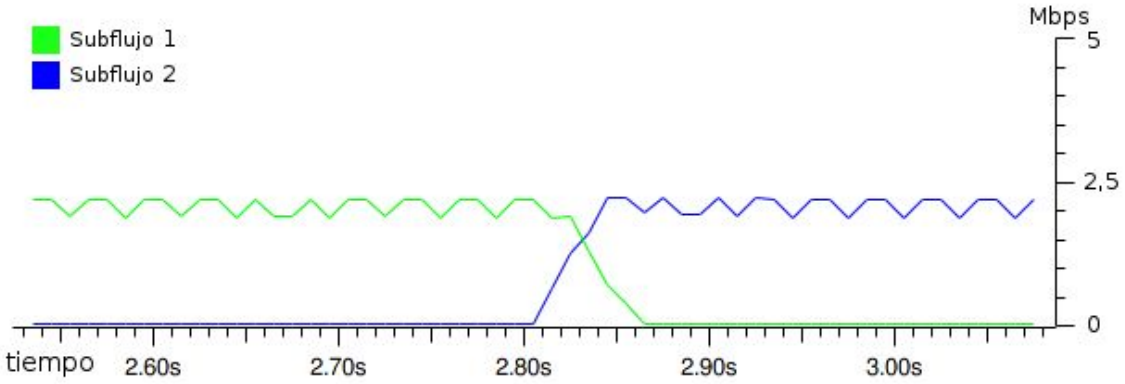
3.5 Resultados



3. Mediciones y resultados

3.5 Resultados





4 Conclusiones

4. Conclusiones

4.1 Conclusiones

- MPTCP supone una mejora real en entornos con múltiples interfaces
- Flujo de datos estable ante pérdidas de conexión
- Implementación para Linux robusta y en continua mejora

Todos los objetivos propuestos han sido cumplidos

4. Conclusiones

4.2 Trabajo futuro

- Casos de uso de movilidad
- Estudio de otras implementaciones de MPTCP
- Análisis de seguridad del protocolo
- Estudio de otras funcionalidades

¿Preguntas?