INSTITUTO DE INFORMÁTICA
Universidade Federal de Goiás

Um Estudo de Ferramentas de Monitoramento de Sistemas

Matheus Messias Valadão Barbosa

Orientador: Prof. Bruno Oliveira Silvestre

INSTITUTO DE INFORMÁTICA
UFG

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Introdução

- A observabilidade é um ramo importante da computação que tem como objetivo a visualização e a análise de sistemas.
- Nuvem e microsserviços
- Ela é dividida em três pilares
  - Monitoramento/Métricas
  - Logs
  - Traces (Rastreio)
- Nosso foco será no monitoramento

Introdução

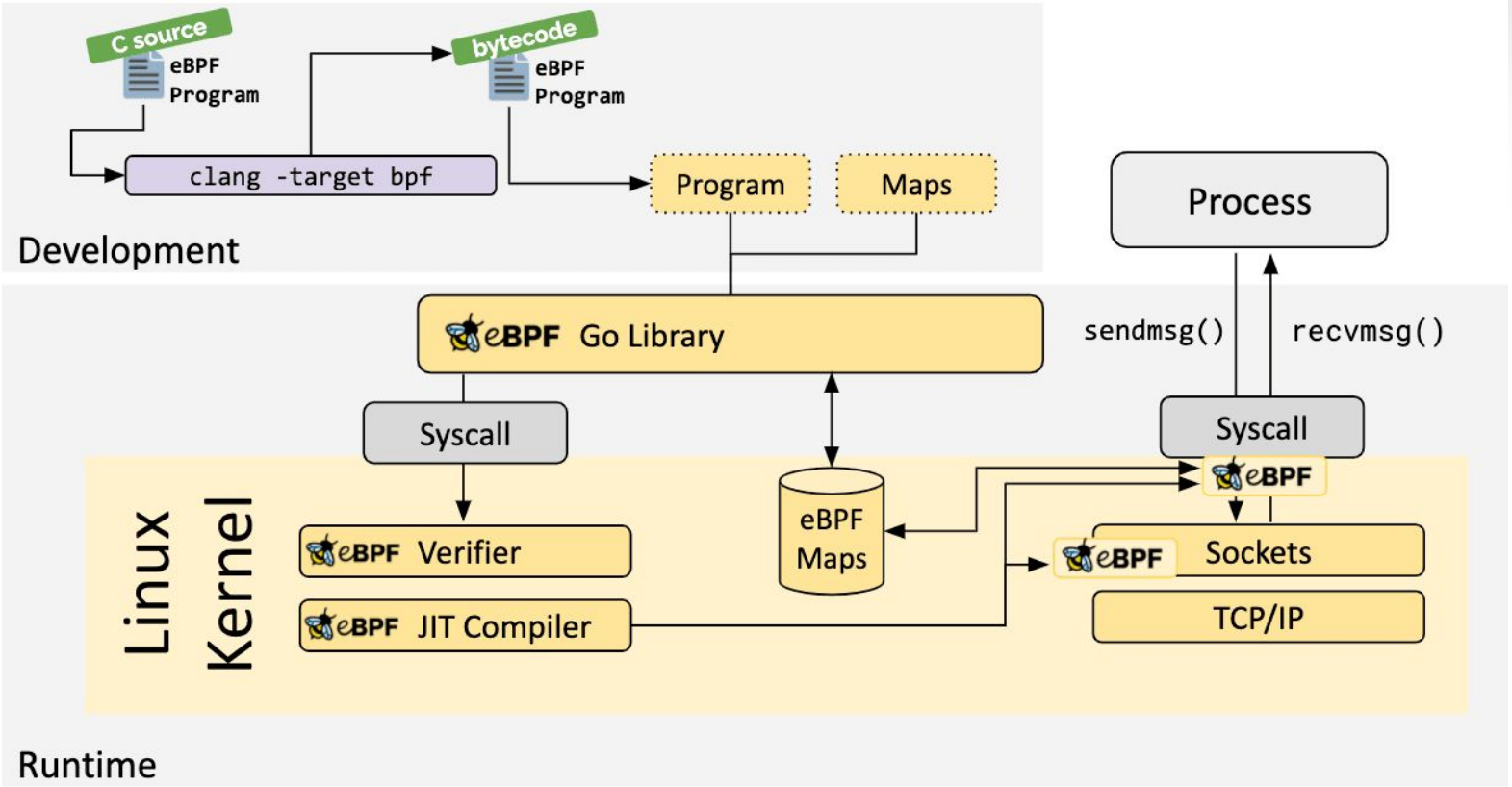
- Neste trabalho foi realizado:
  - Investigação sobre o eBPF
  - Levantamento de ferramentas de monitoramento
  - Estudo de caso de ferramentas de monitoramento

eBPF

eBPF

- O eBPF é um framework que permite criar programas que serão executados diretamente no kernel
- Permite a comunicação direta entre o kernel e o userspace
- Permite vincular programas a syscalls do kernel, para serem executados assim que uma chamada dessas syscalls for realizada
- Utilizada para filtragem de pacotes de rede, monitoramento e segurança

eBPF - Runtime



eBPF - Como criar programas

- Programas eBPF podem ser escritos em C e compilados via LLVM, carregados e vinculados pelo bpftool
- O BCC é um framework que permite criar programas eBPF em que serão executados com Python ou Lua
- O bpftrace permite criar programas simples (on-liners) para execução e verificação rápida, ou programas mais complexos baseados em AWK e C

```
$ sudo bpftrace -e ‘BEGIN{printf("Hello, worlds!\n"); exit();}’
```

Ferramentas

Ferramentas de Monitoramento

- CPU
- GPU
- Energia
- Rede

Ferramentas de Monitoramento - CPU

- sar
- pidstat
- runqlat
- runqlen
- flamegraph

Ferramentas de Monitoramento - CPU

- pidstat

```
Média: UID PID %usr %system %guest %wait %CPU CPU Command
Média: 0 951 0,30 0,30 0,00 0,30 0,60 - Xorg
Média: 1000 1520 42,51 2,59 0,00 49,80 45,11 - qterminal
Média: 0 2406 0,00 17,66 0,00 4,79 17,66 - kworker/u8:1
Média: 112 2489 2,89 0,10 0,00 1,30 2,99 - sddm-greeter
Média: 0 2856 0,00 11,98 0,00 2,69 11,98 - kworker/u8:0
Média: 0 2872 0,00 16,97 0,00 4,89 16,97 - kworker/u8:3
Média: 1000 2891 0,00 35,03 0,00 47,31 35,03 - yes
Média: 0 2895 66,27 1,90 0,00 31,74 68,16 - stress
Média: 0 2896 62,77 2,20 0,00 35,03 64,97 - stress
```

Ferramentas de Monitoramento - CPU

- runqlat

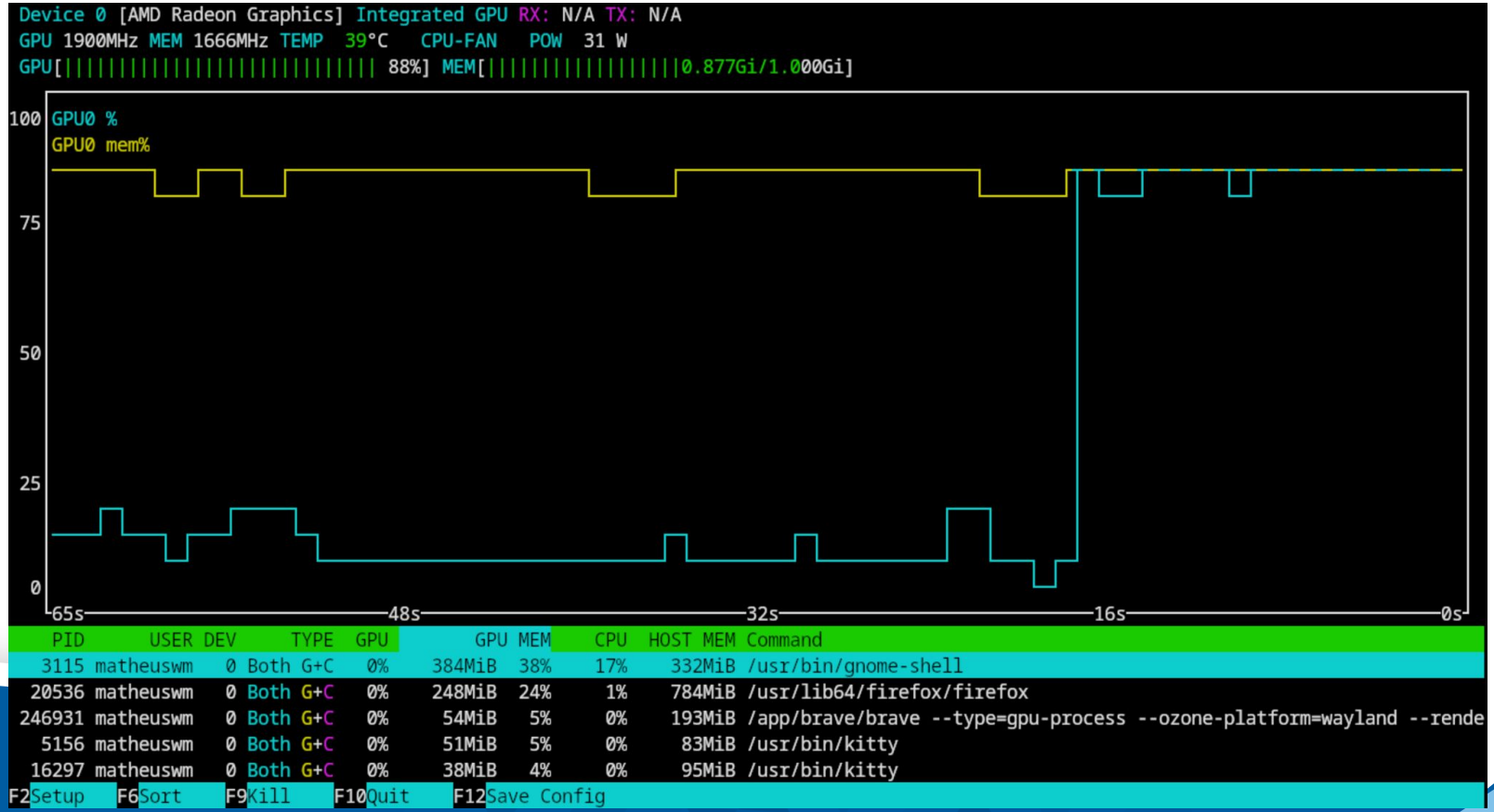
```
Tracing run queue latency... Hit Ctrl-C to end.

     usecs               : count     distribution
         0 -> 1          : 59    |                                        |
         2 -> 3          : 78    |                                        |
         4 -> 7          : 40    |                                        |
         8 -> 15         : 11    |                                        |
        16 -> 31         : 10    |                                        |
        32 -> 63         : 2     |                                        |
        64 -> 127        : 3     |                                        |
       128 -> 255        : 1     |                                        |
       256 -> 511        : 1     |                                        |
       512 -> 1023       : 2     |                                        |
      1024 -> 2047       : 0     |                                        |
      2048 -> 4095       : 107   |                                        |
      4096 -> 8191       : 6573  |****************************************|
      8192 -> 16383      : 25    |                                        |
```

Ferramentas de Monitoramento - GPU

- gpustat
- nvidia-smi
- nvtop

Ferramentas de Monitoramento - GPU

- nvtop



Ferramentas de Monitoramento - Energia

- powerstat
- powertop

Ferramentas de Monitoramento - Energia

- powertop

```
Summary: 834,0 wakeups/s, 0,0 GPU ops/s, 0,0 VFS ops/sec and
542,8% CPU use
Power est. Usage Events/s Category Description
5.36 W 989,3 ms/s 1,0 Process  [PID 288340] stress -c 6
5.15 W 949,0 ms/s 0,25 Process [PID 288344] stress -c 6
4.90 W 904,3 ms/s 1,9 Process [PID 288345] stress -c 6
4.90 W 903,5 ms/s 0,4 Process [PID 288343] stress -c 6
4.76 W 877,5 ms/s 0,05 Process [PID 288341] stress -c 6
3.63 W 670,2 ms/s 1,6 Process [PID 288342] stress -c 6
204 mW 37,6 ms/ 13,4 Process [PID 287898] /opt/google/chrome
```

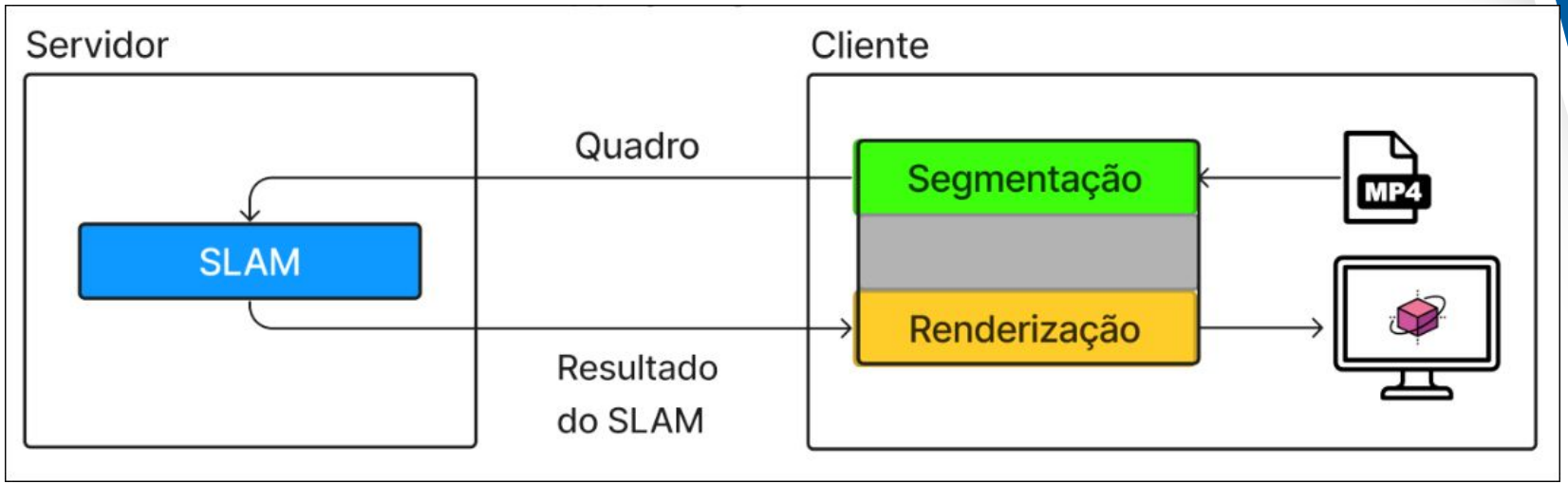
Ferramentas de Monitoramento - Rede

- sar
- tcpretrans
- tcplife
- tcpwin

Estudo de Caso

Estudo de Caso

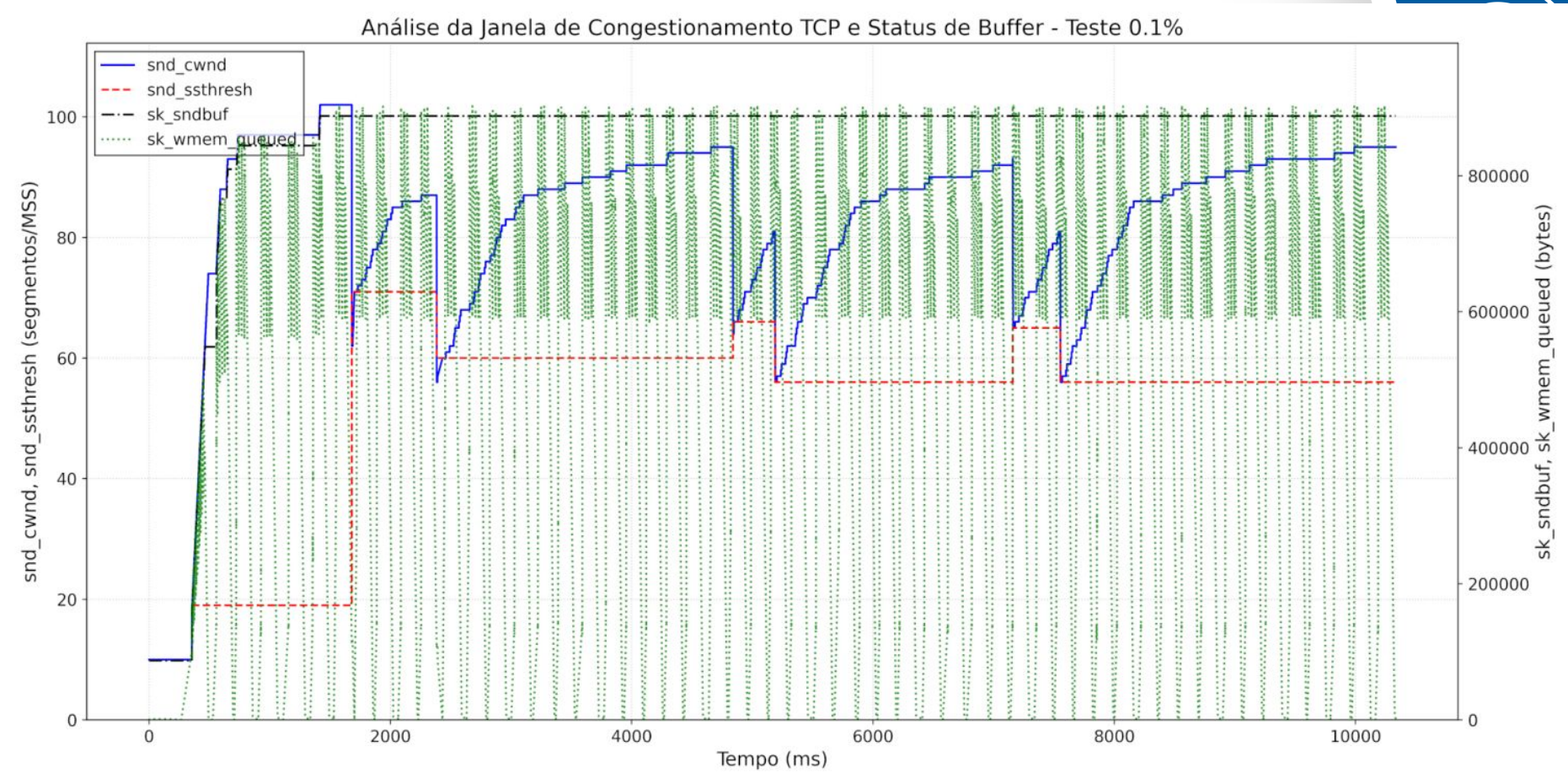
- **Aplicação**



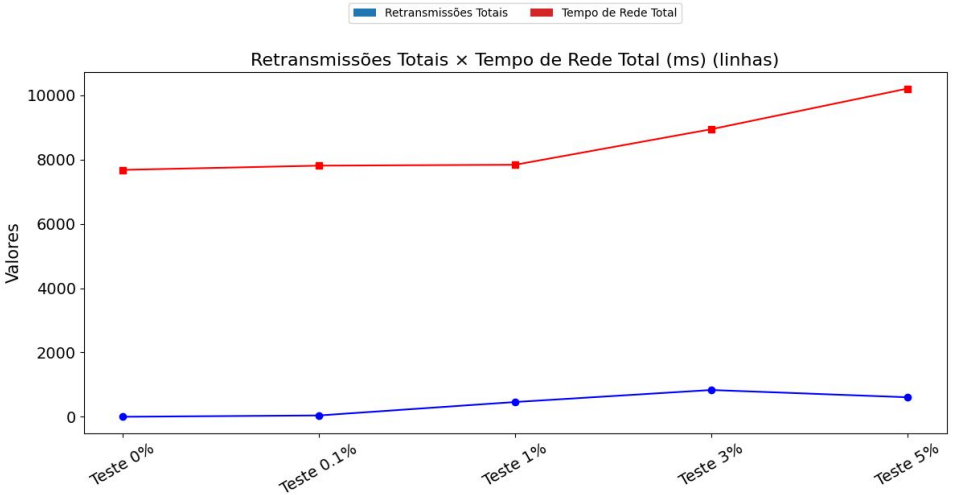
Fonte: PIRES, M.; ESPINDOLA, G. N.; RODRIGUES, K. C.; CARDOSO, K. V.; VERDI, F. L.; SILVESTRE, B.; CORREA, S. L. Uma avaliação empírica sobre o uso de webassembly para descarga de funções de realidade aumentada, 2024

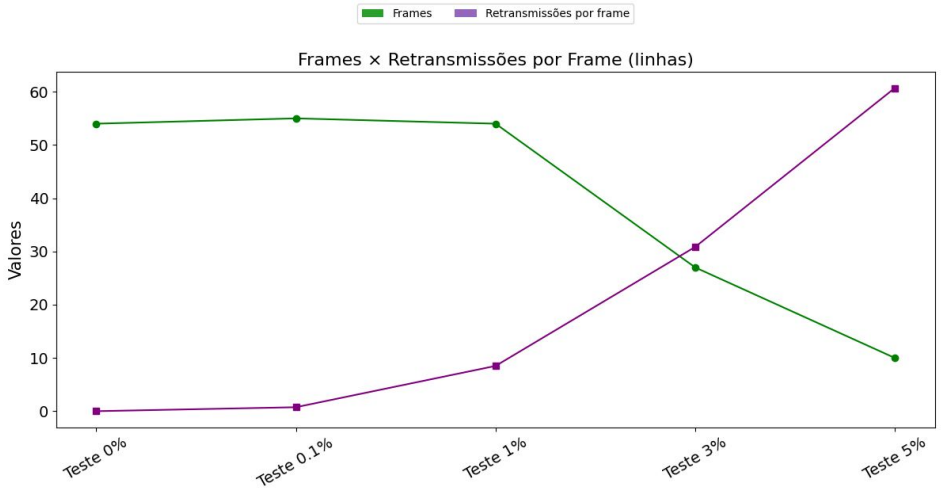
Estudo de Caso

- **Cenário do Teste**
  - O cliente e o servidor foram conectados via cabo (back-to-back)
  - O cliente sofreu com perdas de pacotes ocasionadas pelo traffic control
  - Os teste foram executados no cliente

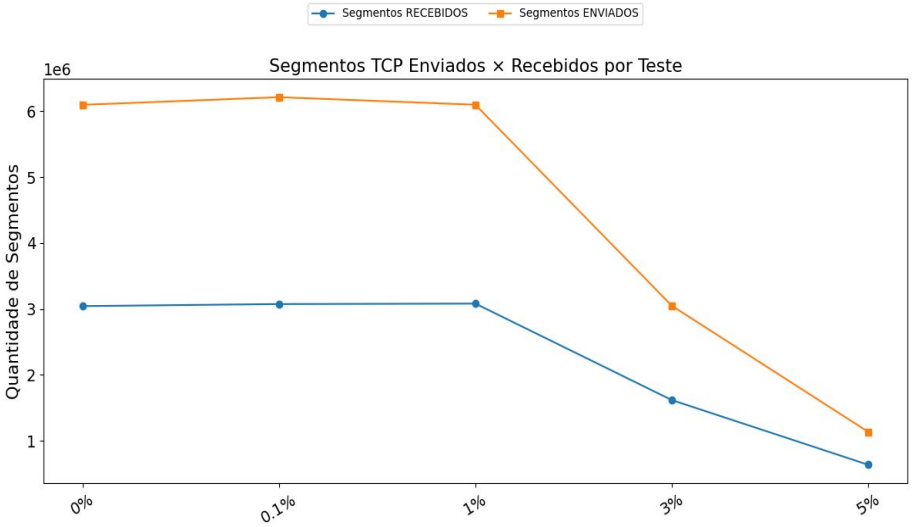

Análise da Janela de Congestionamento TCP e Status de Buffer - Teste 0.1%
snd_cwnd
snd_ssthresh
sk_sndbuf
sk_wmem_queued
snd_cwnd, snd_ssthresh (segmentos/MSS)
sk_sndbuf, sk_wmem_queued (bytes)
Tempo (ms)
0
20
40
60
80
100
0
200000
400000
600000
800000
0
2000
4000
6000
8000
10000

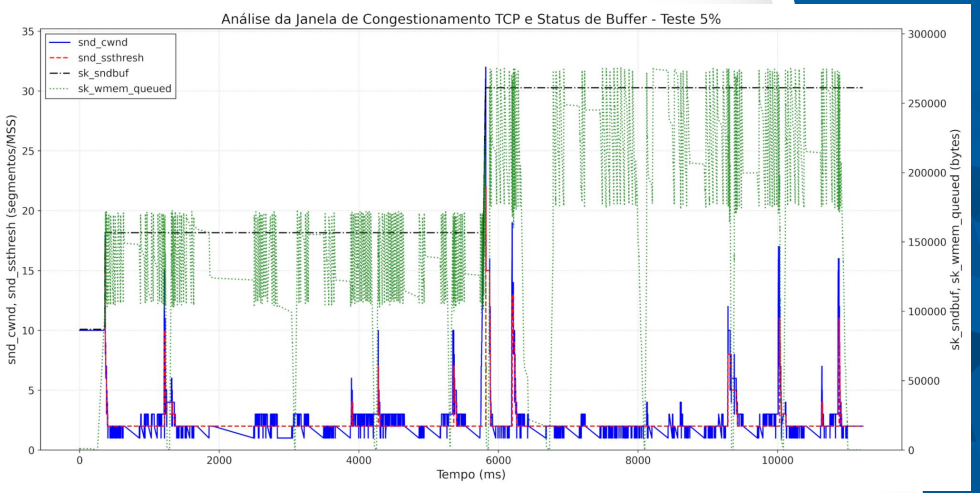
Estudo de Caso - Comparação





Estudo de Caso - Comparação





Conclusão

Conclusão

- O eBPF se mostrou bem flexível podendo monitorar o sistema em baixo-nível, com métricas precisas, com baixo impacto no sistema. Porém para o monitoramento mais específicos surge a necessidade de criar as suas próprias ferramentas
- As ferramentas prontas se mostram interessantes por oferecerem várias métricas, que podem satisfazer a necessidade do monitoramento

Conclusão

- As ferramentas utilizadas no estudo de caso permitiram a análise do tráfego de rede, com a quantidade de segmentos enviados e recebidos e a análise do comportamento da janela de congestionamento.
- Como trabalhos futuros proponho uma exploração mais profunda da programação com eBPF e da criação de ferramentas próprias