

Protocolo P2P para teste de primalidade

Victor Sprengel NUSP: 9298002

Lucas Romão NUSP: 8536214

Cesar Cano NUSP: 8536169

O protocolo

- O protocolo foi criado a partir do zero, com base nas necessidades do projeto e discussões do grupo.
- O teste de primalidade feito pelos peers é feito por sucessivas divisões, até a raiz quadrada do número de entrada.
- O líder distribui intervalos de números para cada um dos peers testarem a divisão, até que todos números foram testados ou algum peer achou um divisor e sabemos que o número é composto.

Comandos

HELLO: comando feito no início de uma conexão de um peer com o outro.

HI: resposta de um peer comum dada ao HELLO. Indica que a conexão foi reconhecida.

OBEY a1 a2: resposta dada ao HELLO de um peer que se conectou, com o número que está sendo processado e o IP do líder

ex: OBEY 472149 192.168.0.12

Comandos

CHUNK i1 i2 i3: comando que o líder envia para os outros peers com intervalos de números a serem testados como divisores do primo em processamento. i1 é o início do intervalo de números, i2 é o fim, i3 é o último número que foi testado como divisor com sucesso, para, caso esse peer se torne o líder, ele possa continuar o processamento.

Os peers respondem com COMPOSITE se algum dos números do intervalo for divisor ou UNKNOWN se naquele intervalo nenhum número é divisor.

ex: CHUNK 2 10000 2

Comandos

VOTE ip: comando que os peers enviam com o seu voto, onde ip é o IP do peer que escolheram como novo líder.

Após receber o voto, é passada a resposta RECEIVED.

Comandos

PING: manda para os peers com os quais está conectado e espera a resposta

PONG, se não receber, considera que o peer se desconectou.

Comandos

DONE: comando usado pelo líder para indicar a todos os demais que a computação chegou ao fim, o líder aguarda a resposta XOXO de cada uma das conexões ativas para finalizar o seu próprio processo.

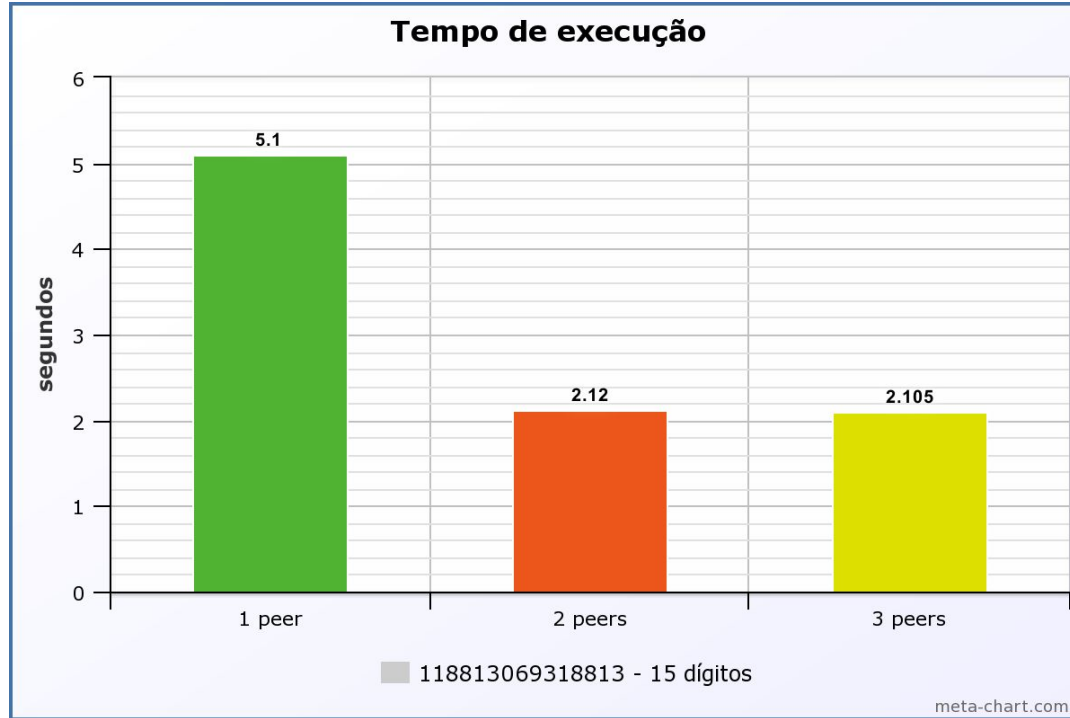
Experimentos

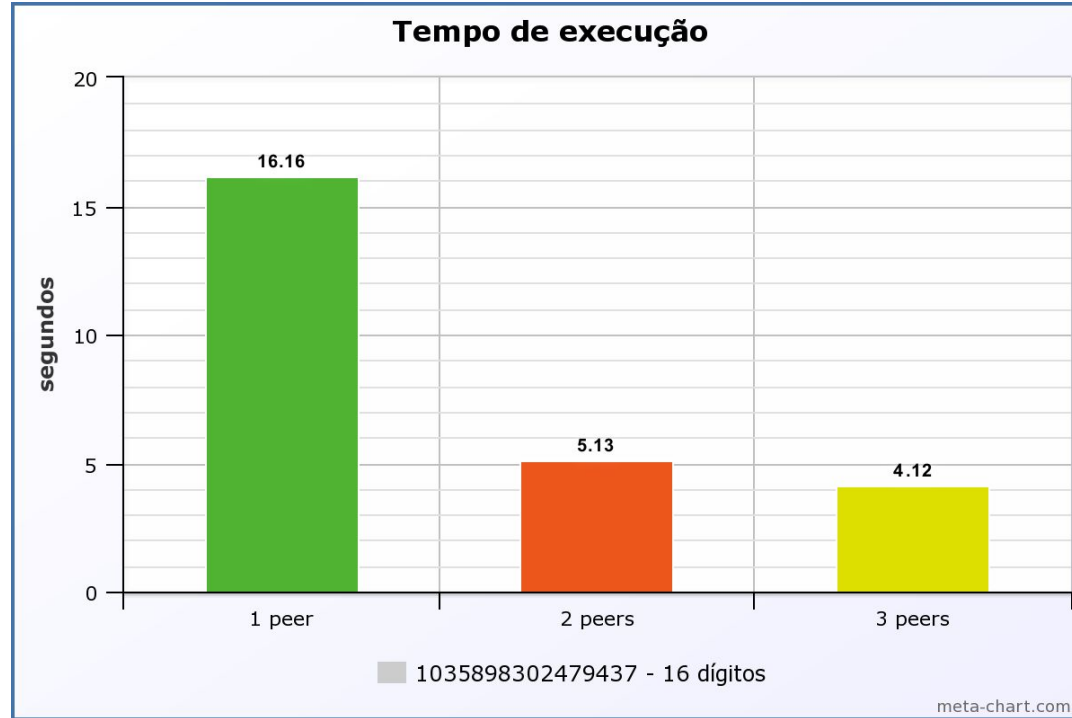
- Os experimentos foram rodados com 2 VMs e uma máquina física.
- Máquinas virtuais:
 - Debian 64 bits
 - 1.5GB RAM
- Máquina onde foi rodado o *peer* líder (sem eleição nos experimentos):
 - 6GB RAM
 - processador i7-4510U @ 2.00GHz (2 cores físicos)

Desempenho (tempo de execução)

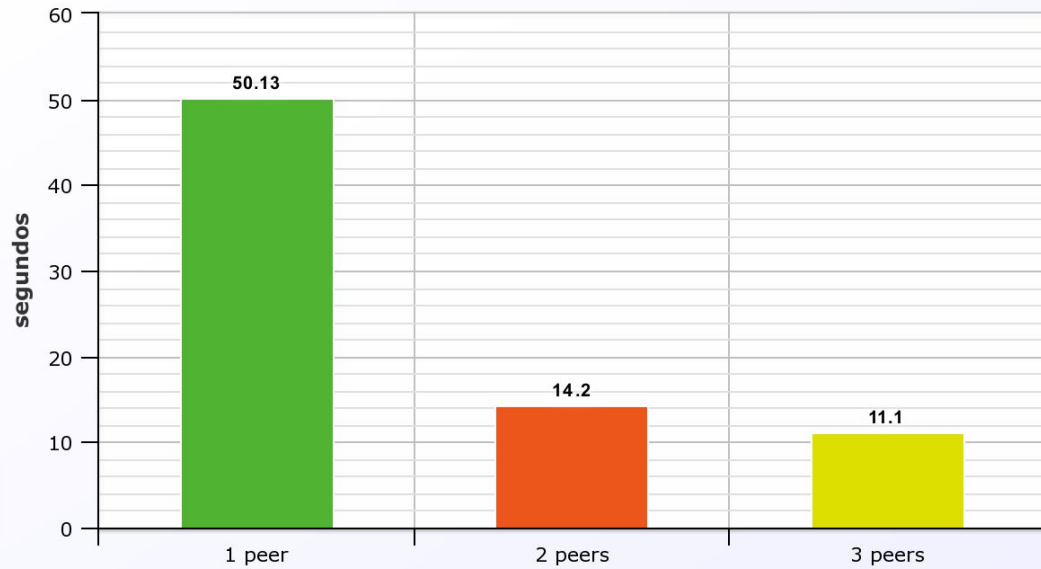
- Para obter os dados de tempo de execução foi utilizado o comando `perf stat` no terminal.
- Os gráficos foram gerados para primos de 15, 16 e 17 dígitos e, para cada caso desses, é mostrada a diferença de tempo de execução quando se tem um único peer, dois peers ou três peers no processamento do primo.

Gráficos de tempo de execução





Tempo de execução

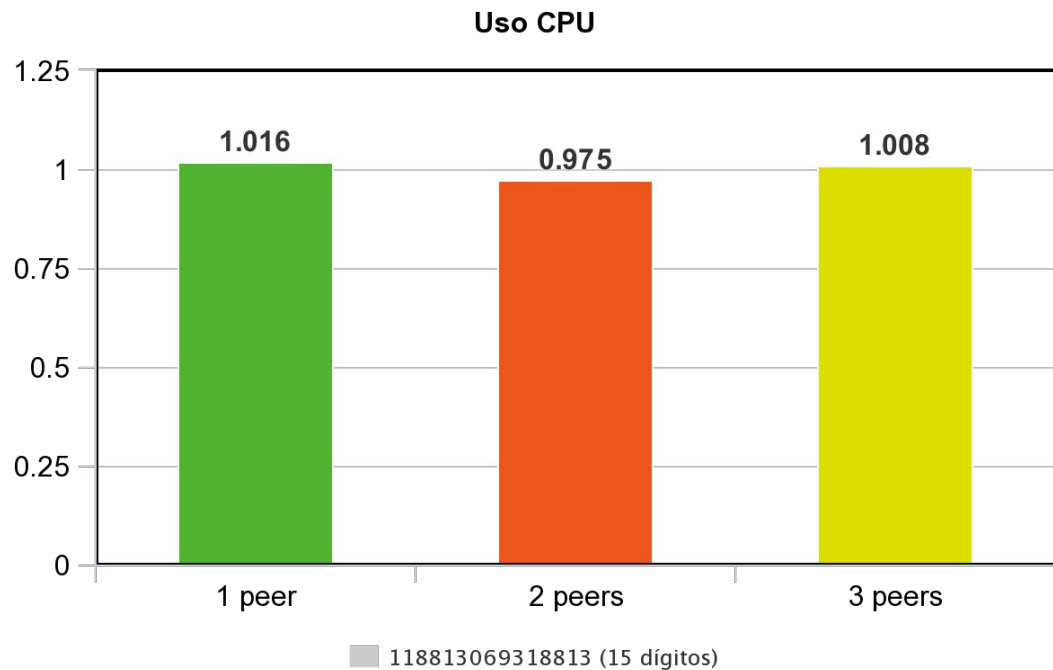


10768252274445931 - 17 dígitos

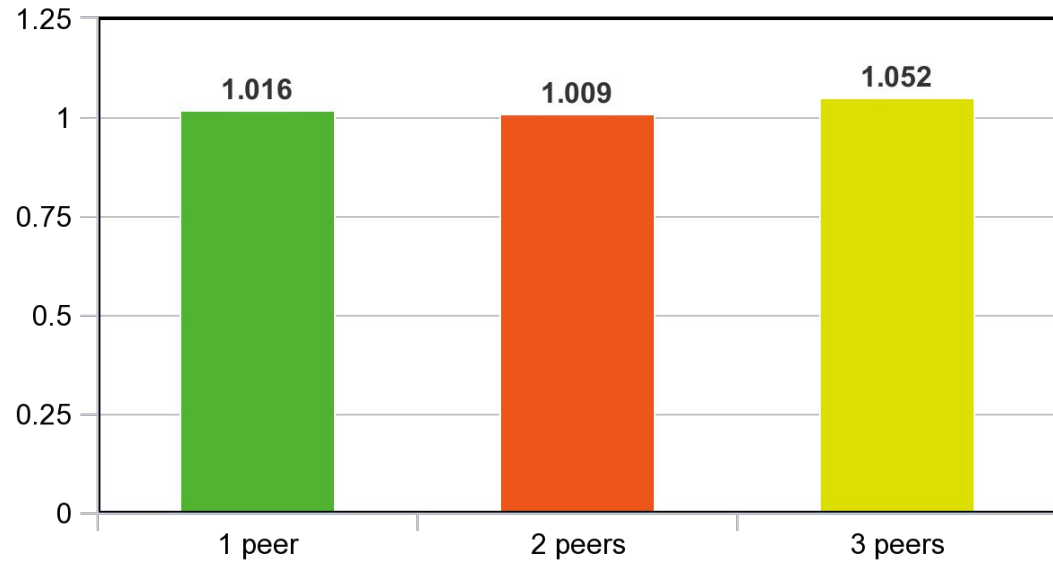
Uso da CPU

- Para obter os dados de tempo de execução foi utilizado o comando `perf stat` no terminal. O número usado nos gráficos é a porcentagem da CPU usada para o processo.
- Os gráficos foram gerados para primos de 15, 16 e 17 dígitos e, para cada caso desses, é mostrada a diferença de uso de CPU quando se tem um único peer, dois peers ou três peers no processamento.

Gráficos - uso de CPU

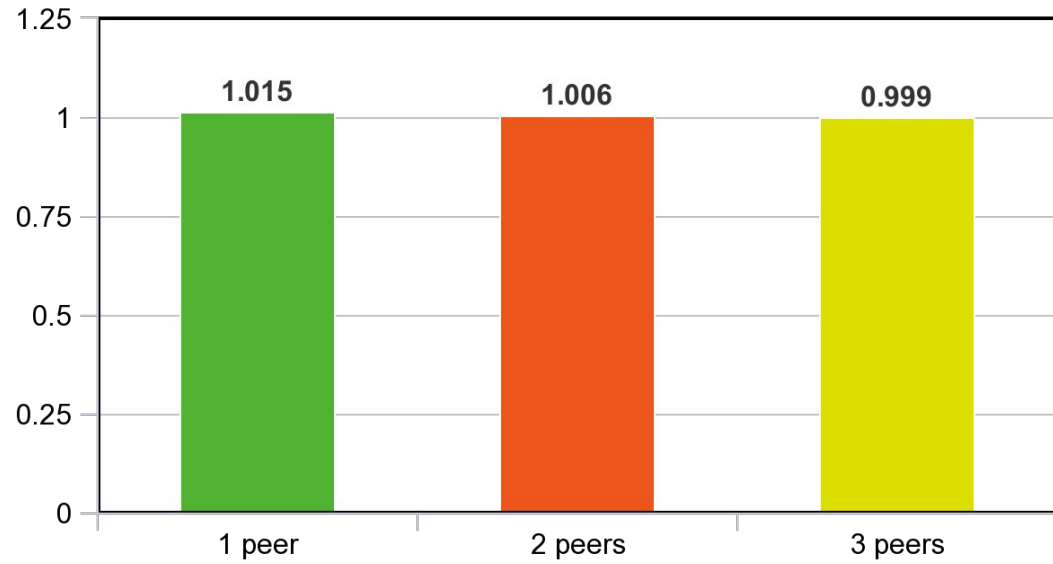


Uso CPU



10358983302479437 (16 dígitos)

Uso CPU

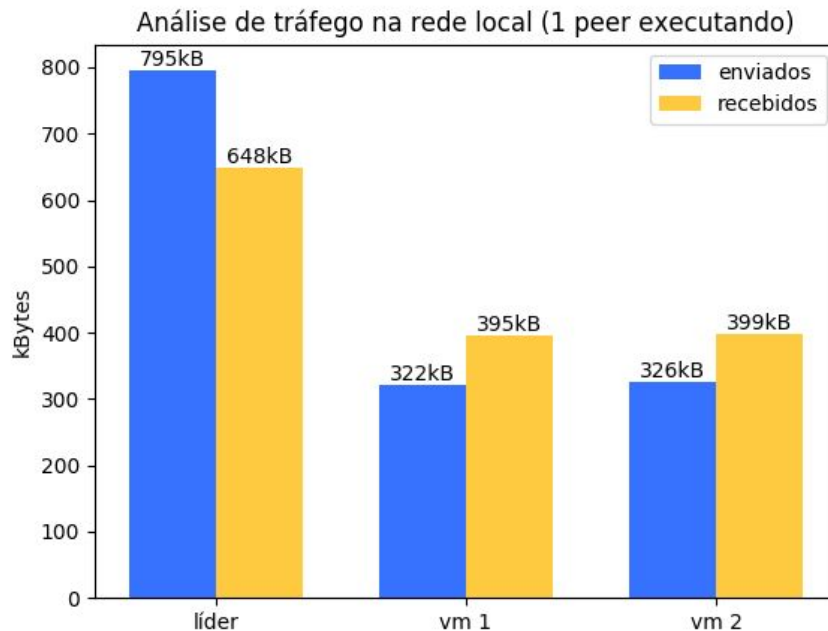


10768252274446931 (17 dígitos)

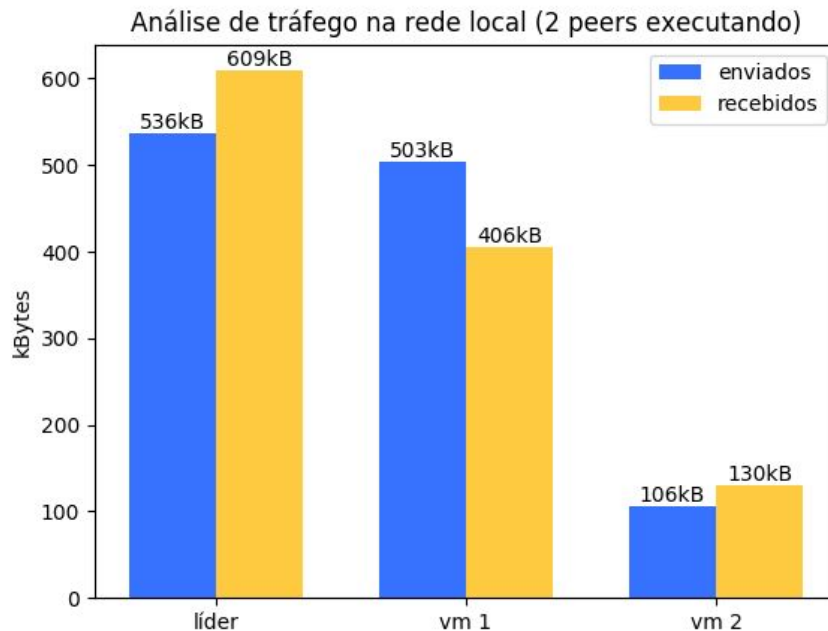
Uso da rede

- Os experimentos de uso da rede foram feitos utilizando Wireshark para capturar pacotes com filtro por protocolo TCP e com dest/source pertencente às máquinas envolvidas.
- Depois, os dados foram obtidos na aba Statistics -> Endpoints, onde é possível saber quantos pacotes/bytes foram trocados por cada IP que aparece como dst ou src na lista de pacotes capturados.
- Finalmente, foram gerados os gráficos considerando todas máquinas ligadas (mesmo quando não estavam executando a aplicação, pois pacotes de tentativa de conexão eram recebidos)

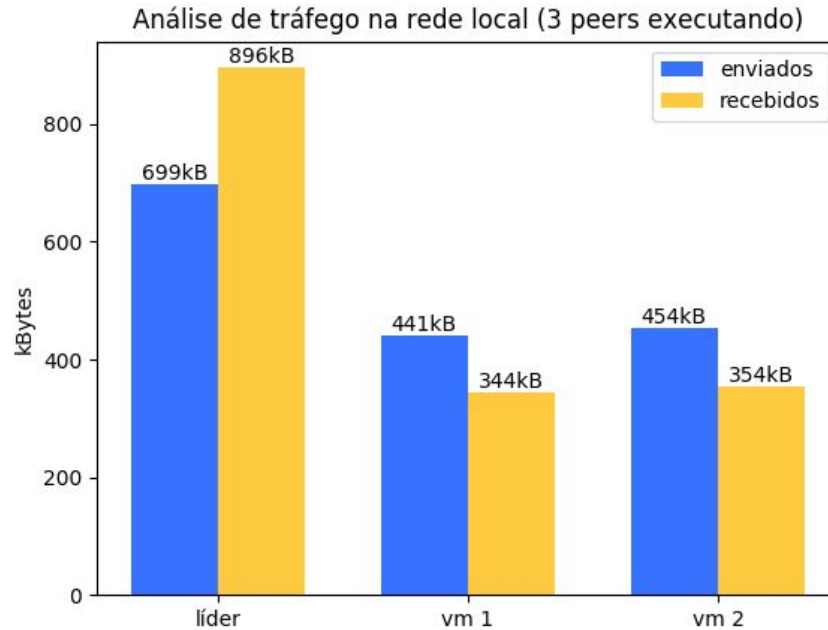
Apenas líder executando processo



Líder e VM 1 executando o processo



Todos executando o processo



Conclusões

- É possível ver um ganho de performance significativo em relação a tempo de execução quando se tem dois peers, e quando em três o ganho não é tão grande, o que pode ser explicado por um overhead maior das comunicações TCP
- O uso de CPU não mostrou alterações significativas, usando sempre em torno de 1%.
- Os testes de rede mostram que mesmo não executando o processo, há troca de pacotes nas tentativas de conexão. Também é possível ver o aumento de dados enviados e recebidos pelo líder de acordo com o número de peers.

Obrigado