Universidade Federal de Santa Catarina
Departamento de Informática e Estatística
Ciências da Computação
INE5415 - Redes de Computadores I
Professor: Carlos Becker Westphall

Aluno: Marco Antônio Machado de Arruda - 20200415

Trabalho Prático 2
Monitoramento de Rede, SNMP, UDP, TCP e WireShark

Florianópolis – SC, 17 de Julho de 2022

Resumo

Durante a realização deste trabalho utilizamos das informações obtidas durante as aulas para a avaliação e compreendimento dos conceitos obtidos por meio das ferramentas de redes. Aquela escolhida para o uso do gerenciamento de redes foi o PRTG, na qual utilizamos o teste grátis de 30 dias durante o período de obtenção dos dados, num período mínimo de 2 horas por dia durante 3 dias seguidos, que foram avaliados em diferentes dispositivos conectados a uma rede local. Também avaliamos os protocolos ARP, SNMP, TCP e UDP a partir da captura de pacotes efetuada pelo software WireShark.

Índice

1. Introdução

No mundo atual, a vasta disponibilidade de redes e internet praticamente em todos os locais, traz consigo uma grande comodidade à população, que pode desfrutar de entretenimento, realizar tarefas, comunicação, entre diversas outras atividades em qualquer momento do seu dia a dia, contudo, não é de conhecimento de todos os perigos que tal interligação pode oferecer. Nosso trabalho tem por objetivo explicitar alguns pontos acerca do gerenciamento de redes, principalmente sobre o monitoramento em pacotes como SNMP, que recebemos a todo o momento, ao mesmo passo que compreendemos seus protocolos e ligações entre diferentes máquinas. Buscamos também avaliar discrepâncias no comportamento de gráficos monitorados e o que podem indicar, também alertar sobre casos como picos na rede quando não sabemos uma causa por trás.

2. Ferramenta de Gerência

Para a realização do trabalho escolhemos o PRTG (Paesller Router Traffic Grapher) Network Monitor, é uma ferramenta de gerência de redes sem agentes, então não foram necessárias instalações em diferentes dispositivos, apenas algumas alterações em comunicações e identificações nas redes entre o computador utilizado e o notebook durante nosso trabalho, ele monitora através de sensores, inseridos a partir de uma máquina com acesso ao software na rede, permite também a criação de gráficos com os dados coletados. A ferramenta apresenta assinaturas pagas mas também um modo gratuito onde podemos utilizar até 100 sensores em dispositivos, junto de algumas outras limitações, ela seria suficiente para a análise e recolhimento dos pacotes obtidos para a nossa avaliação, mas acabamos por utilizar a versão trial, de 30 dias grátis sem limitações.

Utilizamos também o Wireshark, que é um analisador de pacotes da rede, organizado por protocolos e temporalmente, sua principal utilização se dá aplicando filtros e analisando as partes desejadas separadamente devido a sua captura detalhada.

3. Topologia de Rede

Foi utilizada uma rede doméstica, provida pela empresa Olé Telecom. Nela conectamos um Notebook Acer Aspire 3, um computador com acesso cabeado a rede e dois smartphones, um Motorola One Action e um Samsung Galaxy M52.

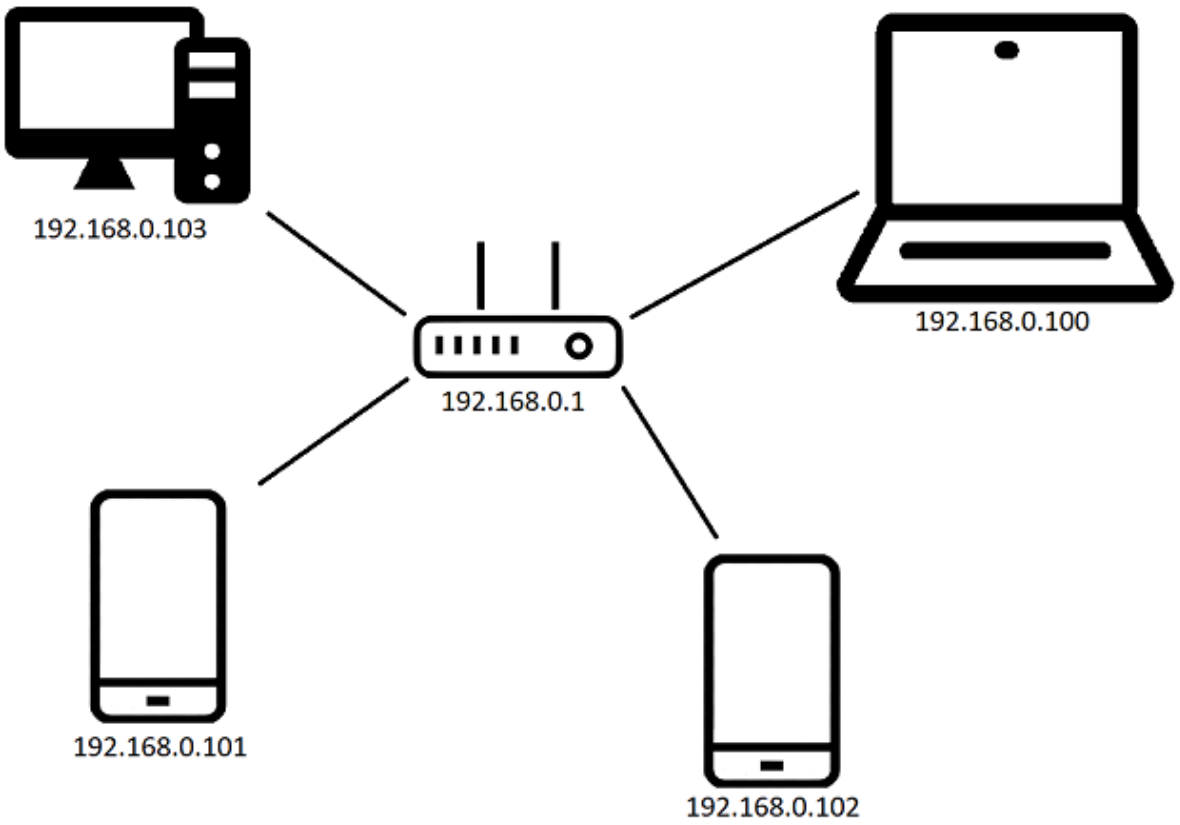

Imagem 1: Representação da conexão na rede local

Os IPs representados na imagem:

Desktop - 192.168.0.103

Notebook - 192.168.0.100

Motorola - 192.168.0.102

Samsung - 192.168.0.101

Modem - 192.168.0.1

4. Componentes da Rede

- Modem Wireless Gigabit Dual Band AC1200
    - Modelo: EC 220-G5
    - Frequência de transmissão: 2,4 GHz e 5 GHz

- Notebook Acer
    - Modelo: Aspire 3
    - Processador: i3 - 1005G1 @ 1.2 Ghz
    - Memória RAM: 8 GB
    - Sistema Operacional: Windows 11
    - Adaptador de rede Wireless: Qualcomm Atheros QCA9377

- Desktop
    - Processador: i5-9400F CPU @ 2.90GHz
    - Memória RAM: 16 GB
    - Sistema Operacional: Windows 11
    - Adaptador de Rede Ethernet: Realtek PCIe GbE Family Controller

- Motorola One Action
    - Processador: 4x2.2 GHz ARM Cortex-A73+ 4x 1.6 GHz ARM Cortex-A53
    - Sistema Operacional: Android 11

- Samsung M52
    - Processador: 1x 2.4 GHz Kryo 670 Prime + 3x 2.2 GHz Kryo 670 Gold + 4x 1.9 GHz Kryo 670 Silver
    - Sistema Operacional: Android 11

5. Medições Realizadas

Durante 3 dias por mais de 2 horas mas com interrupções mantivemos sensores nos dispositivos recém citados, dentre eles analisaremos 5, valendo a pena ressaltar que nos smartphones apenas o sensor de ping foi inserido enquanto que no notebook e no desktop os demais foram implementados.

5.1 Ping

Ping foi implementado nos 5 dispositivos, aqui podemos praticamente ignorar o ping em relação ao desktop, visto que é o ping efetuado da máquina para ela própria, em geral, ping envia mensagens seguindo o ICMP (Internet Control Message Protocol), utilizado para medir o atraso em relação à máquina na rede e também para verificar a simples existência de um aparelho conectado.

Aqui podemos notar o quão rápida é a resposta dada do modem diretamente para o desktop, e quão lento os celulares foram, prestando atenção na escala que não é a mesma para todos, podemos notar o quanto o celular Motorola variou suas respostas, e de forma contrária o Samsung manteve uma resposta com pouca alteração; também vemos que o notebook utilizando uma ligação via Wi-Fi assim como os celulares respondeu significativamente mais rápido.

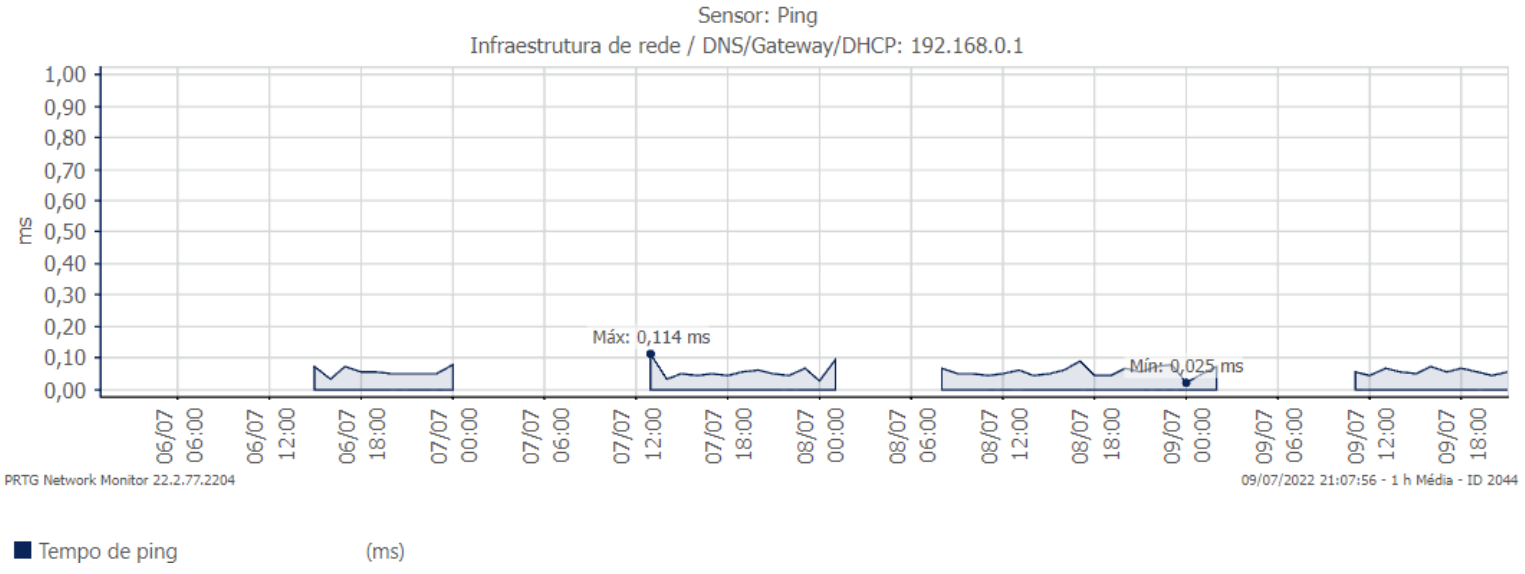

Gráfico 1: Monitoramento do sensor Ping em relação ao Modem (192.168.0.1)

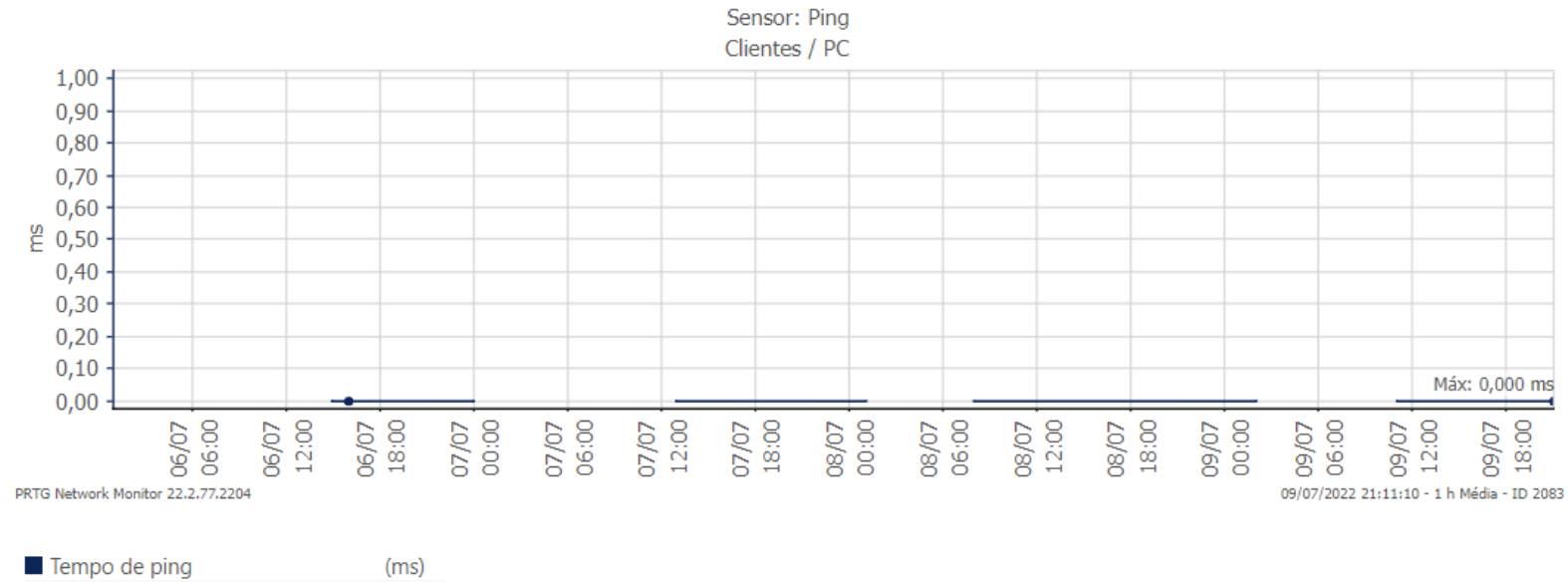

Gráfico 2: Monitoramento do sensor Ping em relação ao Desktop (192.168.0.103)

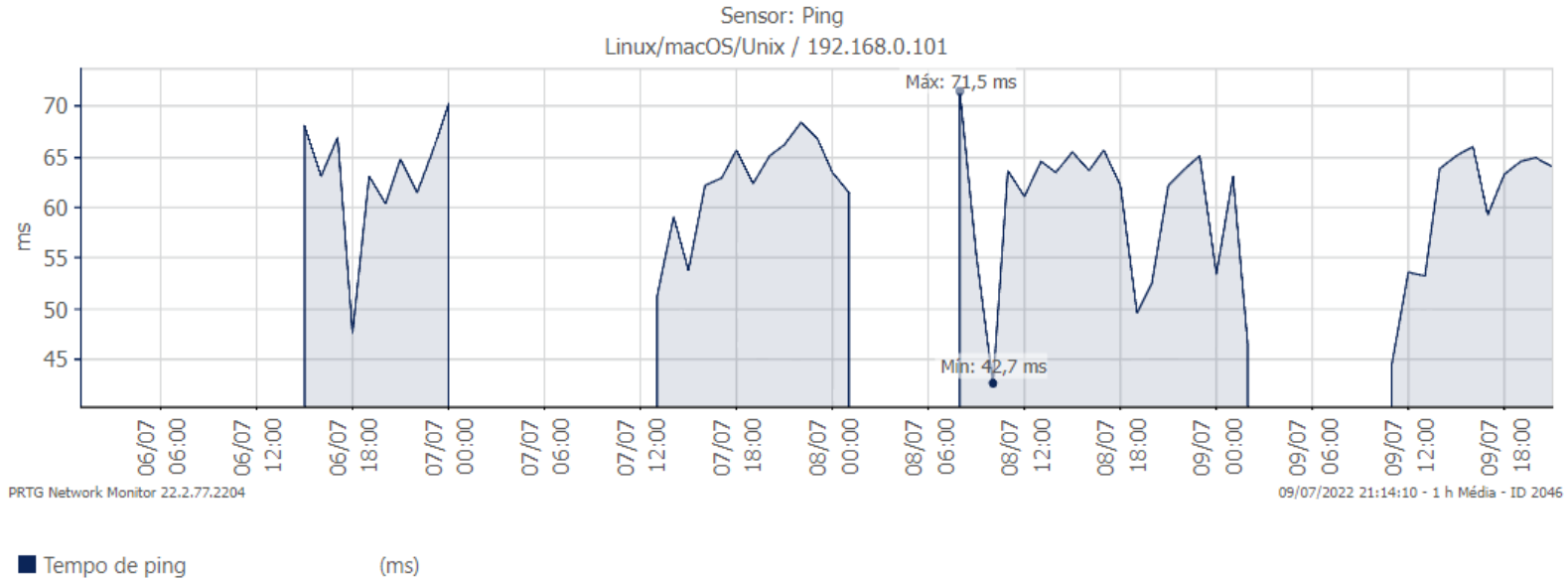

Gráfico 3: Monitoramento do sensor Ping em relação ao Samsung (192.168.0.101)

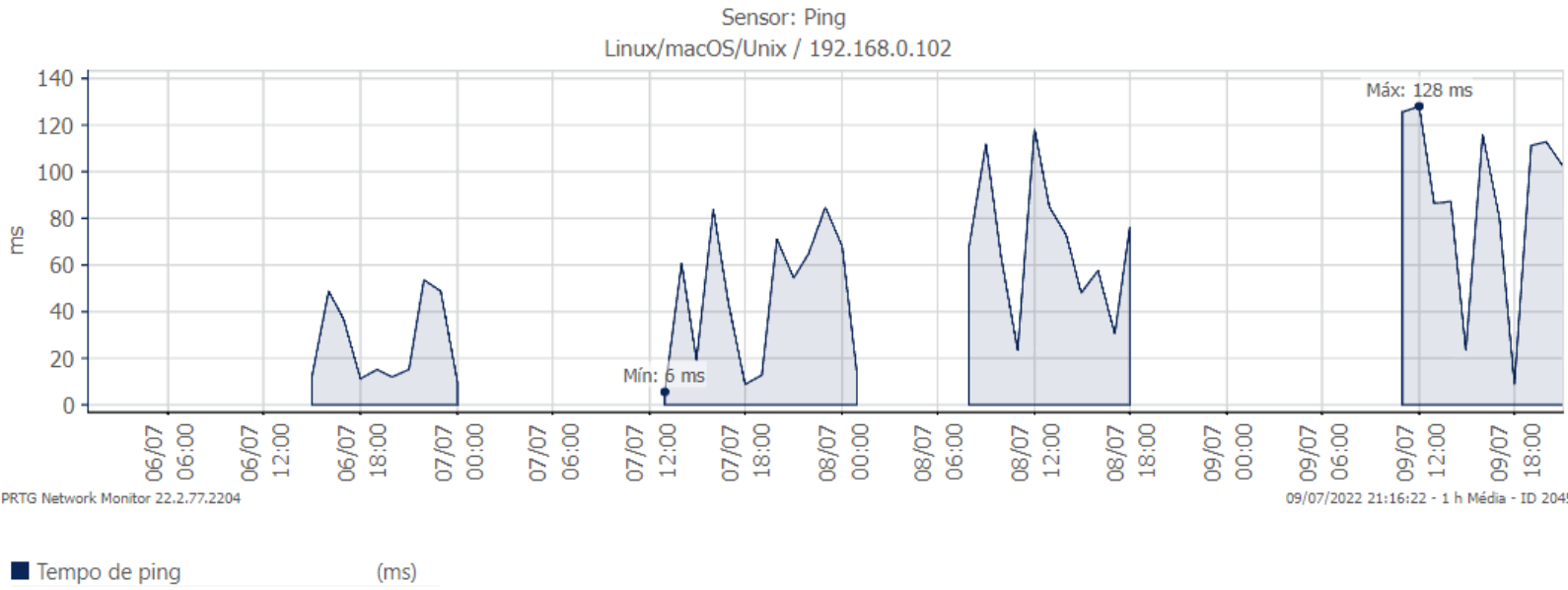

Gráfico 4: Monitoramento do sensor Ping em relação ao Motorola (192.168.0.102)

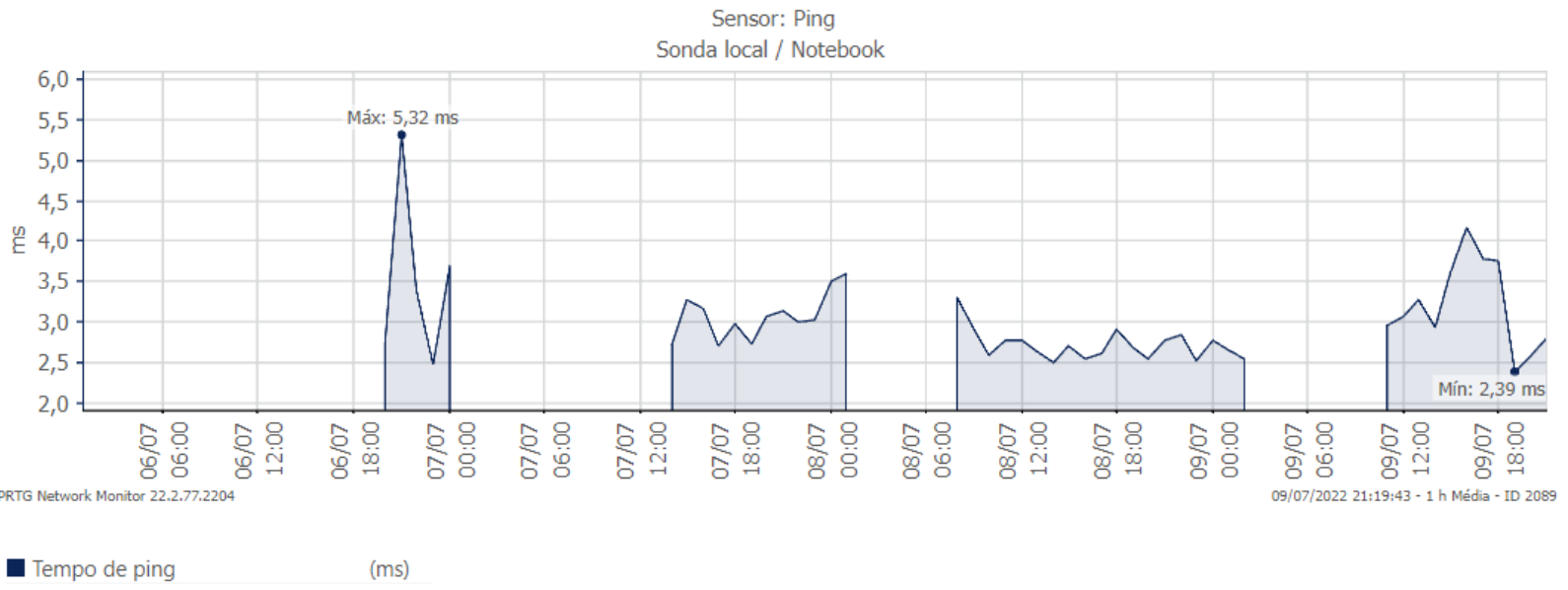

Gráfico 5: Monitoramento do sensor Ping em relação ao Notebook (192.168.0.100)

5.2 Common SaaS Check

O sensor Common SaaS Check monitora alguns Software as a Service, como já vistos em aula, são uma forma de distribuir softwares baseados em clouds, neste gráfico, vamos notar baseados na conexão da sonda local a disponibilidade de alguns serviços.

Interessante notar alguns picos apresentados principalmente pelo Bing, foram pontos esporádicos que algumas vezes tiveram um atraso total de praticamente 1,5 segundo, interessante também notar algo como "3 linhas" com tempos de atraso diferentes entre as quais os serviços se dividem, sendo a mais de baixo contendo o GitHub e o Office, a intermediária (em geral) contendo o Youtube, Google Apps (bem inconstante) e o Facebook, já a mais inconstante contendo o restante, sendo eles o Salesforce, Twitter, Bing e o Dropbox, todos esses com variações bem perceptíveis.

Tentei encontrar alguma correlação entre o atraso e as Clouds que disponibilizaram o serviço mas não encontrei provas para isso, por exemplo, tanto o Facebook quanto o DropBox são hosteados por uma Cloud da Amazon.

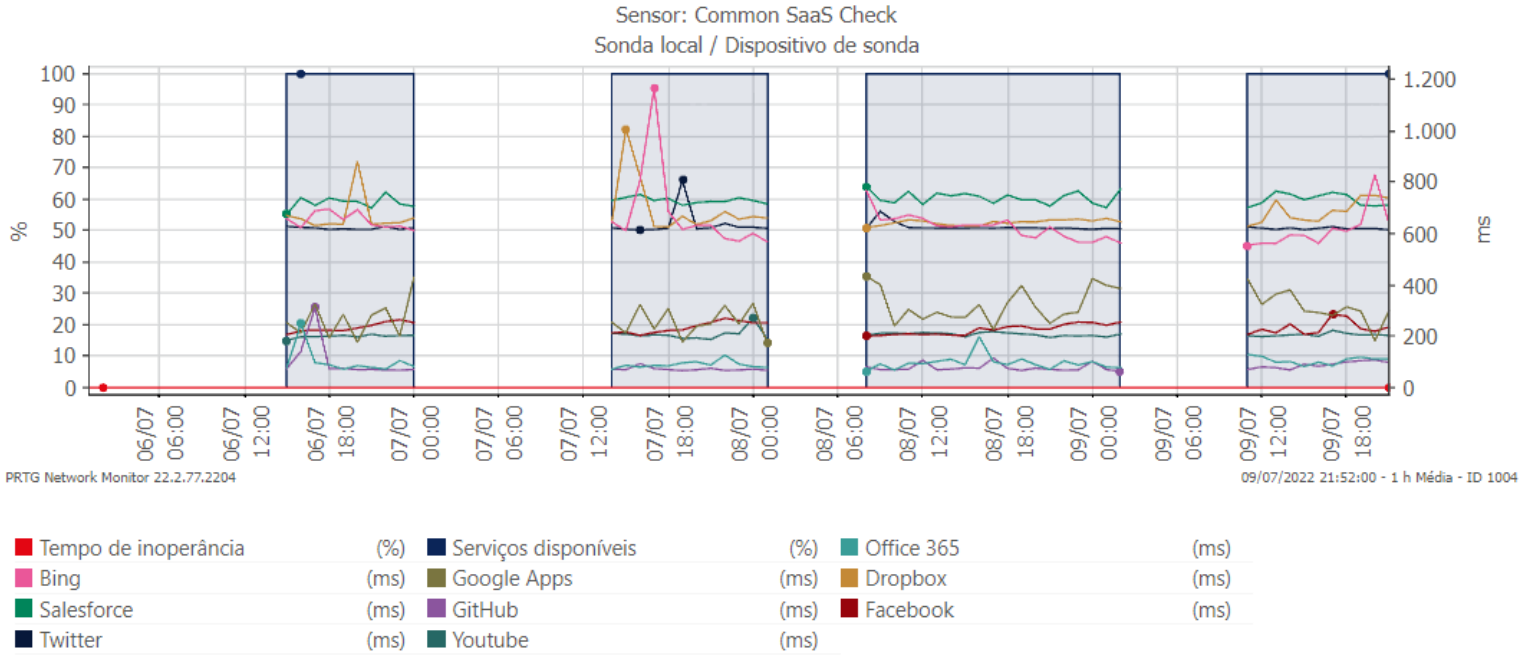

Gráfico 6: Monitoramento do sensor Common SaaS Check na sonda local

5.3 Ethernet / Wi-Fi Traffic

Tanto o sensor de tráfego de Ethernet (desktop) quanto o de Wi-Fi monitoram a largura de banda pelo usando o protocolo SNMP, aqui foram feitos alguns testes básicos, principalmente utilizando o computador.

Dia 6 das 19h até as 22h deixamos abertas 9 livestreams simultâneamente (144p 30fps) no notebook (Wi-Fi traffic) às 22:30 pusemos uma delas em 1080p 60 fps, aquí podemos notar a imensa diferença no tráfego de saída só por conta de uma alteração na qualidade gráfica (ver gráfico 9 para melhor detalhamento).

Dia 7 às 22h jogamos CSGO, um jogo online com servidores localizados em São Paulo, num modo de 5 contra 5, podemos notar um pico relativamente alto quando comparado com o restante do período naquele dia (ver gráfico 10 para melhor detalhamento).

Em geral o tráfego foi bem alternado, momentos em que se utilizava muito o computador, outros que ficavam em descanso, algumas vezes somente com livestreams abertas, outras com pesquisas e até alguns pequenos downloads juntos. Acabei não deixando downloads por tempo o suficiente para que tenham influenciado significativamente o gráfico, mas vale ressaltar que observando a análise em tempo real do Ethernet Traffic, quando mantemos um download por mais tempo chegamos em torno de 400 Mbit/s (ver gráfico 11 para melhor detalhamento).

Vale ressaltar também alguns momentos de perda de conexão que foram capturados pelo PRTG, no gráfico por conta do intervalo de tempo ser muito grande nessa escala de três dias parece ter sido algo maior do que realmente foi, mas na verdade o sensor de tráfego Wi-Fi capturou apenas 2,591% (1h 8min) de rede inoperante que ainda é um tempo relativamente alto e que pessoalmente acabei nem reparando que fiquei sem internet no notebook, principalmente por conta de que no computador (Ethernet traffic) não foram encontradas quedas de internet, ela manteve-se em 0% de inoperância.

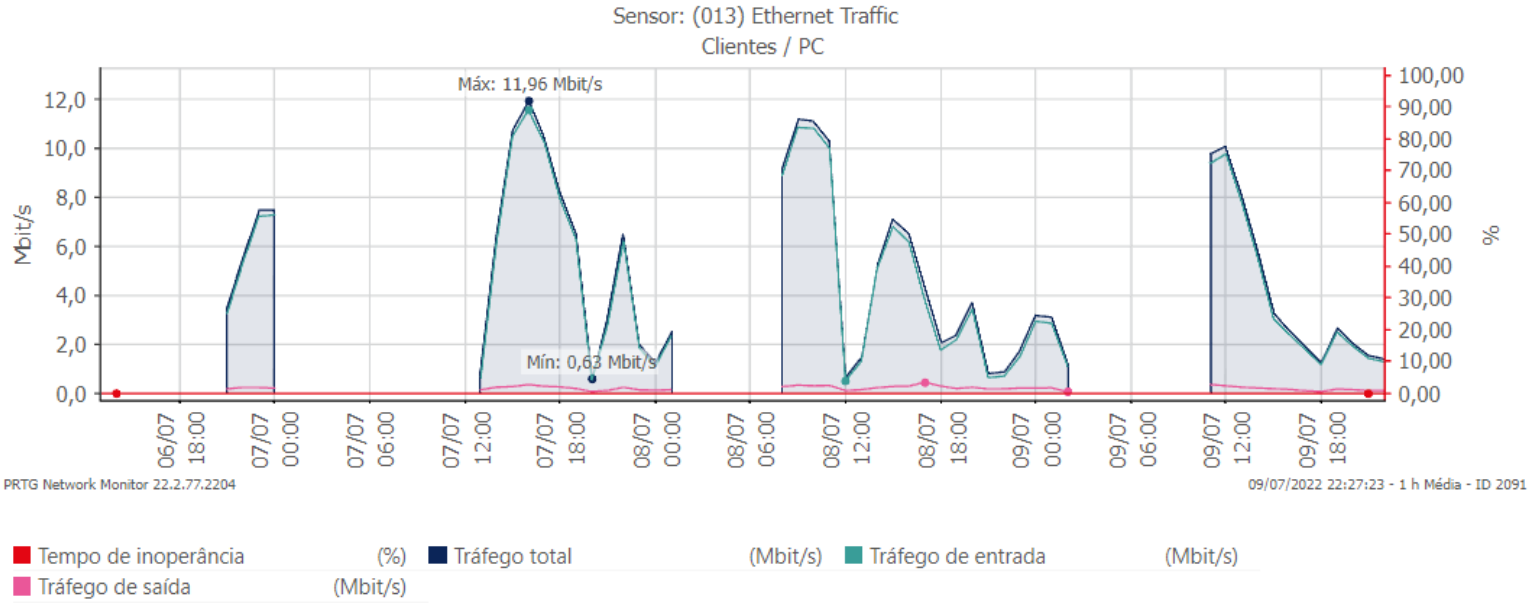

Gráfico 7: Monitoramento do sensor Ethernet Traffic

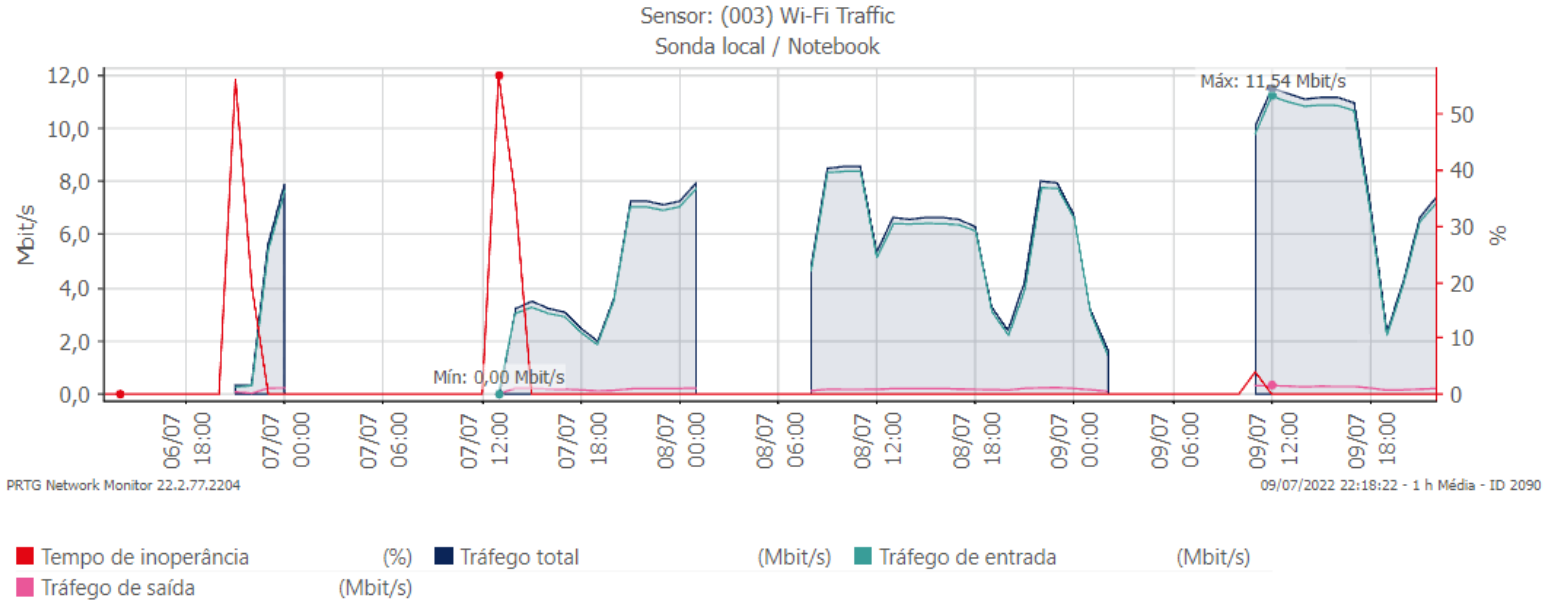

Gráfico 8: Monitoramento do sensor Wi-Fi Traffic

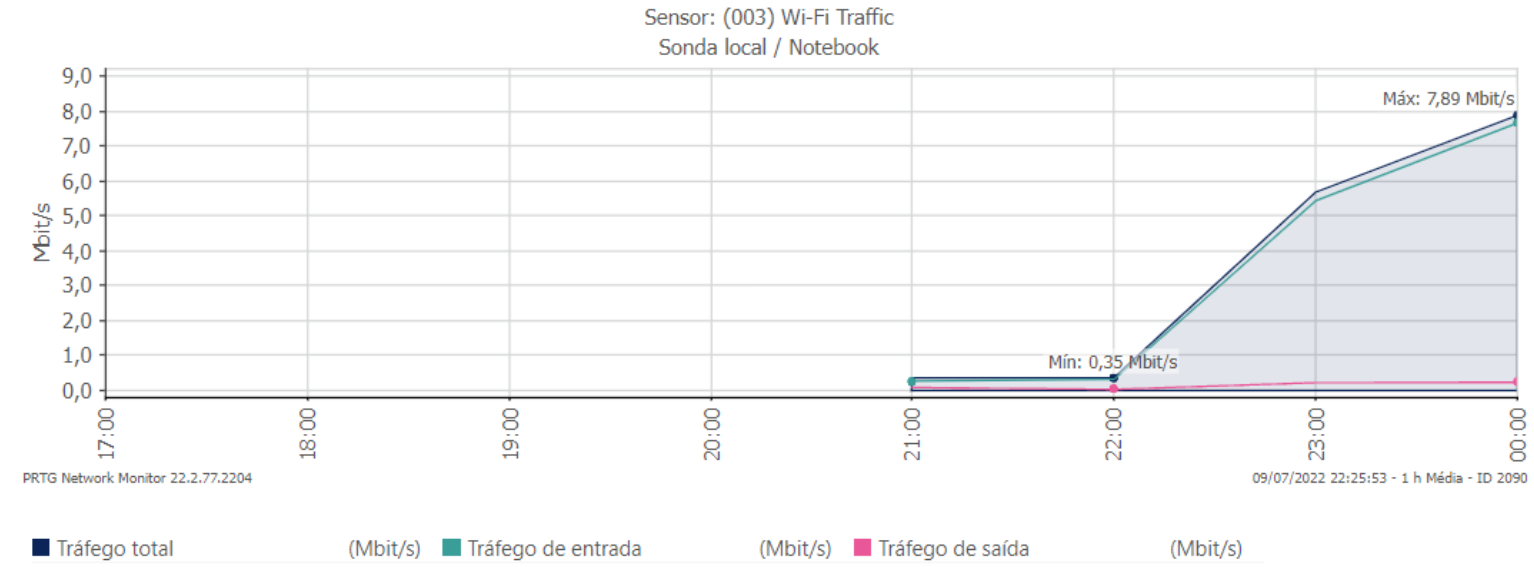

Gráfico 9: Monitoramento Wi-Fi traffic, dia 6 das 17h até 00h

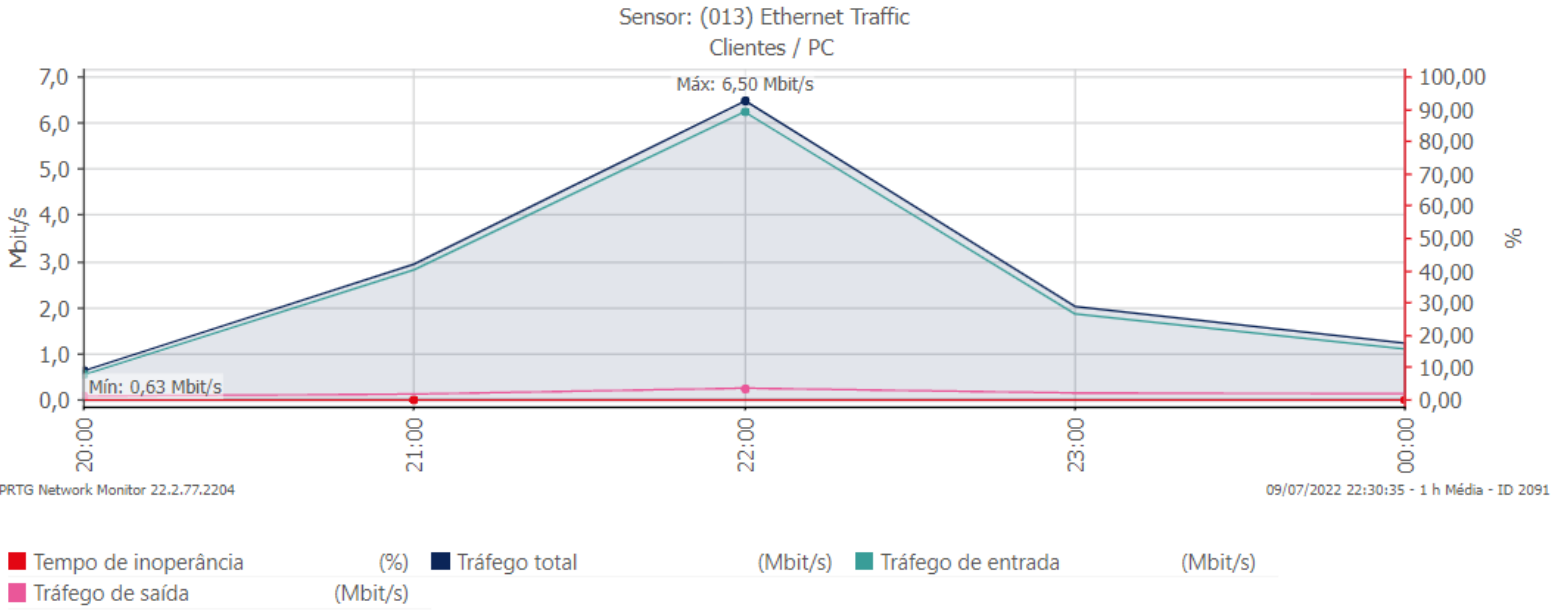

Gráfico 10: Monitoramento Ethernet traffic, dia 7 das 20h até 0h

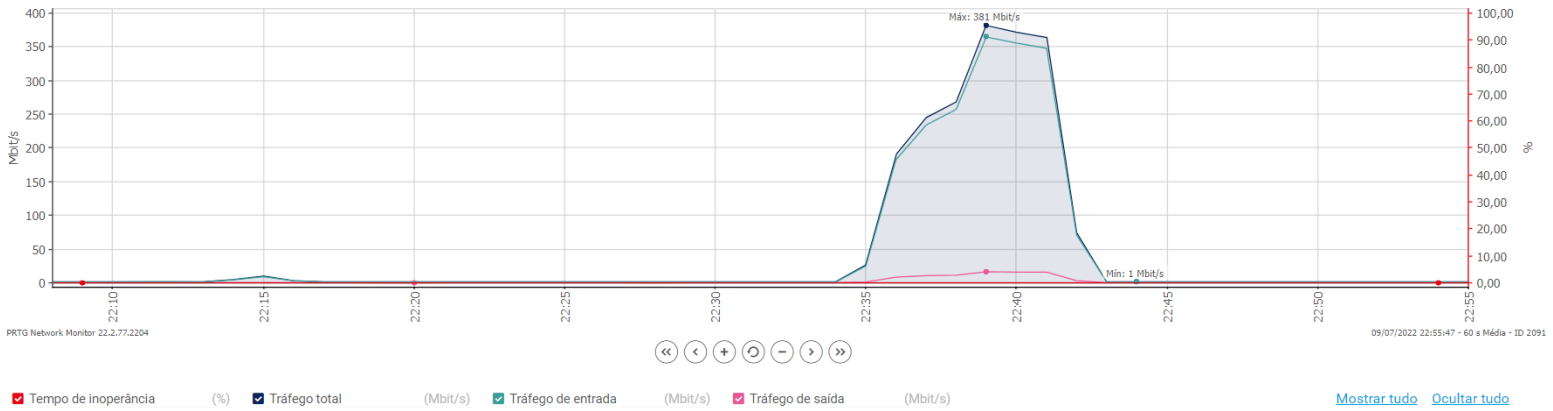

Gráfico 11: Monitoramento Ethernet traffic, tempo real, exemplo download

5.4 HTTP

Utilizamos um sensor HTTP que monitora a disponibilidade de um site na internet, resolvi escolher monitorar o site Twitch.tv, tanto no computador por conta da Ethernet quanto no notebook por conta da Wi-Fi, mas com isso podemos notar que mesmo com o tempo de resposta se mantendo de forma bem constante, fomos capazes de perceber alguns valores de pico extremamente altos quando observamos os dados em tempo real, que por fim acabaram alterando significativamente uma parte dos dados encontrados (ver gráficos 14 e 15 para observar um dos maiores picos capturados)

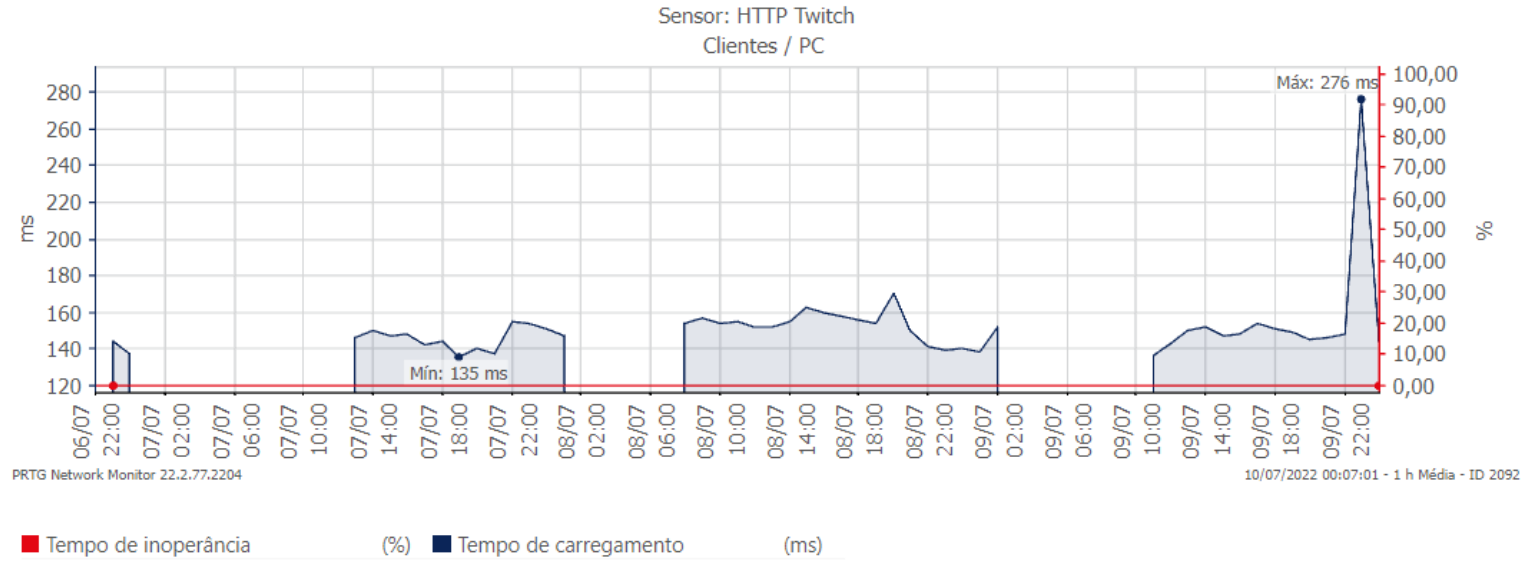

Gráfico 12: Monitoramento HTTP Twitch PC

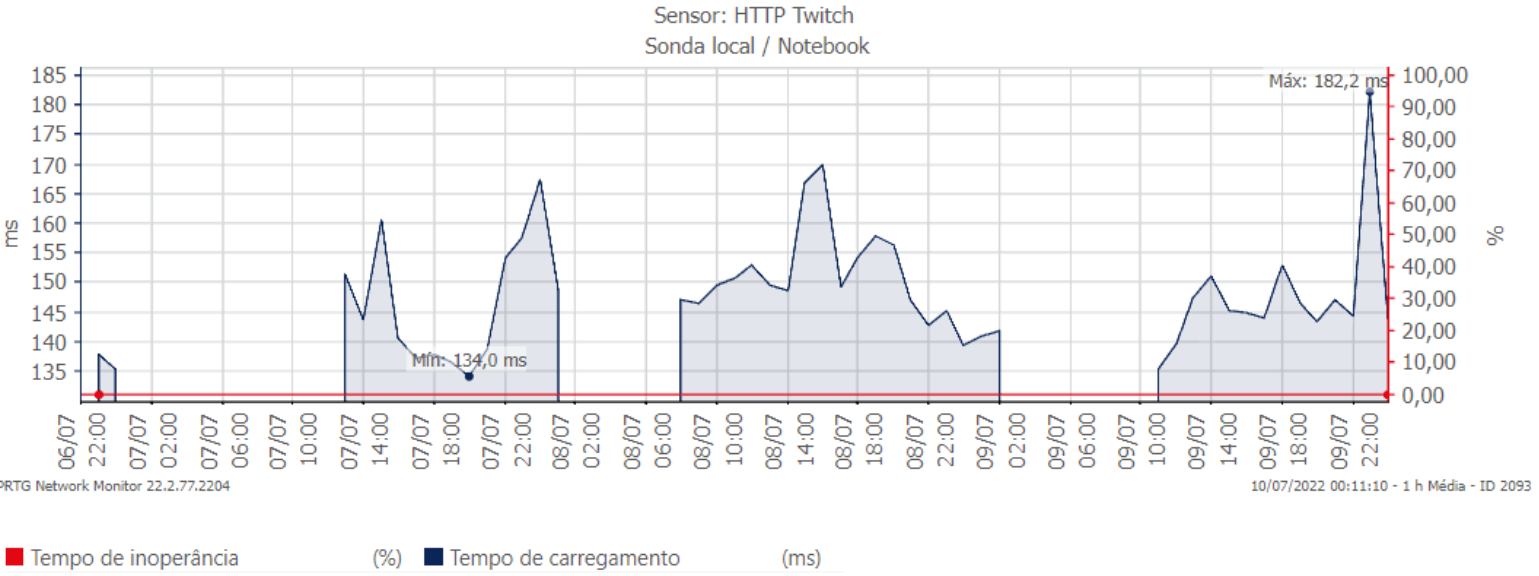

Gráfico 13: Monitoramento HTTP Twitch Notebook

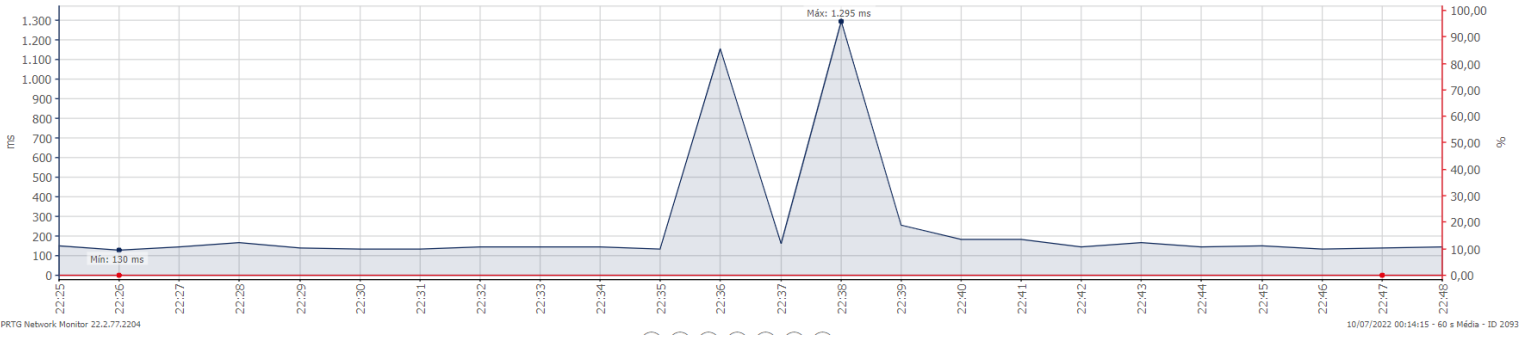

Gráfico 14: HTTP Twitch PC tempo real, pico percebido

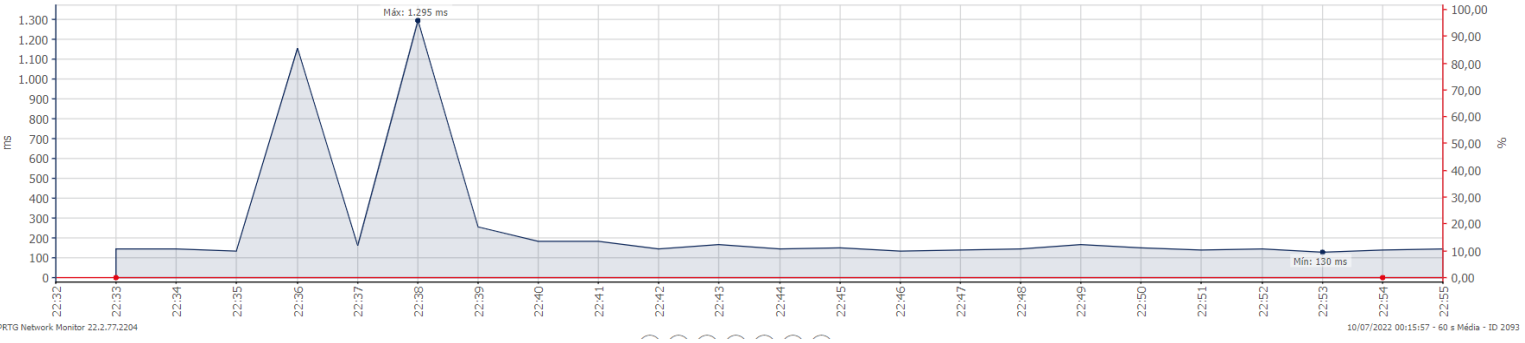

Gráfico 15: HTTP Twitch Notebook tempo real, pico percebido

5.5 Carga CPU (SNMP)

Este sensor monitora a carga da CPU com protocolos SNMP, podemos notar que nos gráficos obtidos o do computador apresenta uma maior variação, isso se dá por conta de que além de em alguns momentos ele estar com livestreams abertas assim como o notebook, algumas outras horas utilizei para fazer pesquisas e até mesmo jogar como dito anteriormente na análise, já a performance do notebook mesmo que mais amena se dá um pouco turva provavelmente por conta do hardware mais fraco encontrado nele visto que não o utilizamos para algo além das livestreams.

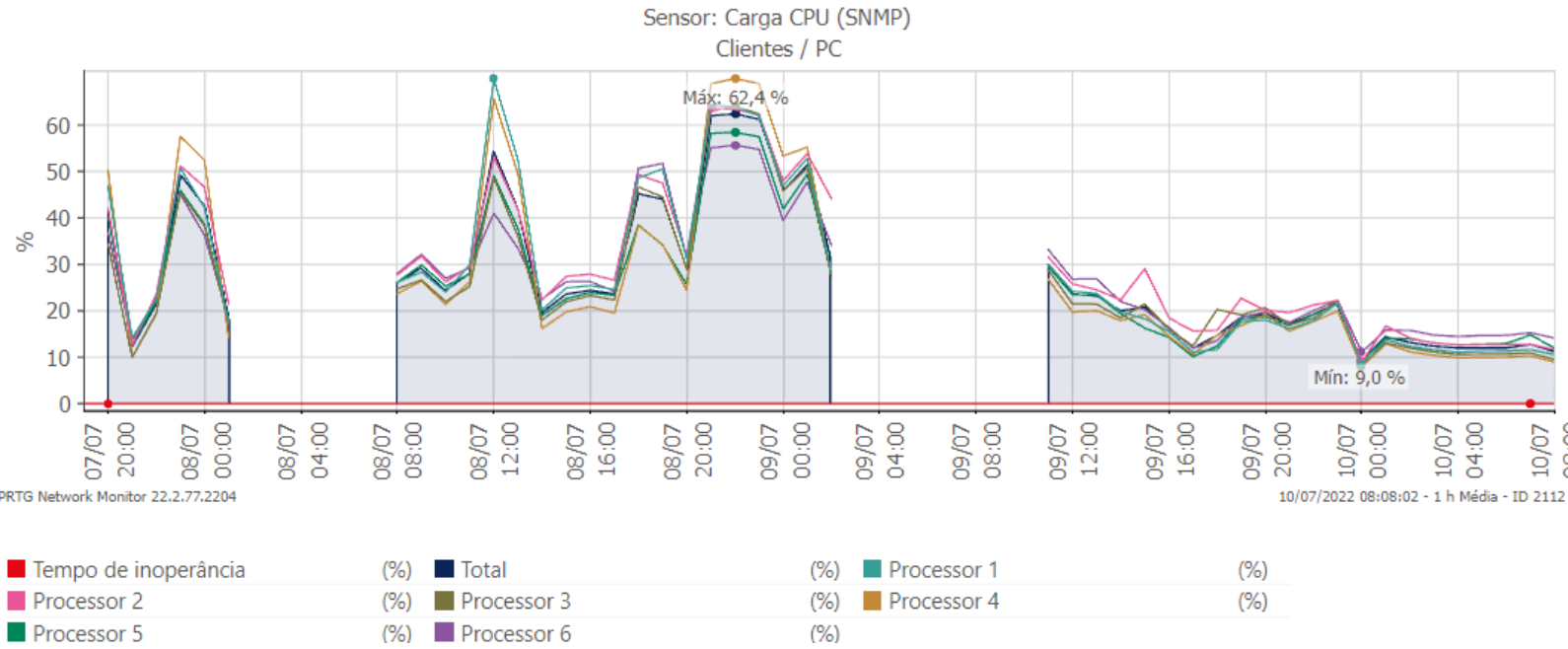

Gráfico 16: Monitoramento Carga CPU (SNMP) PC

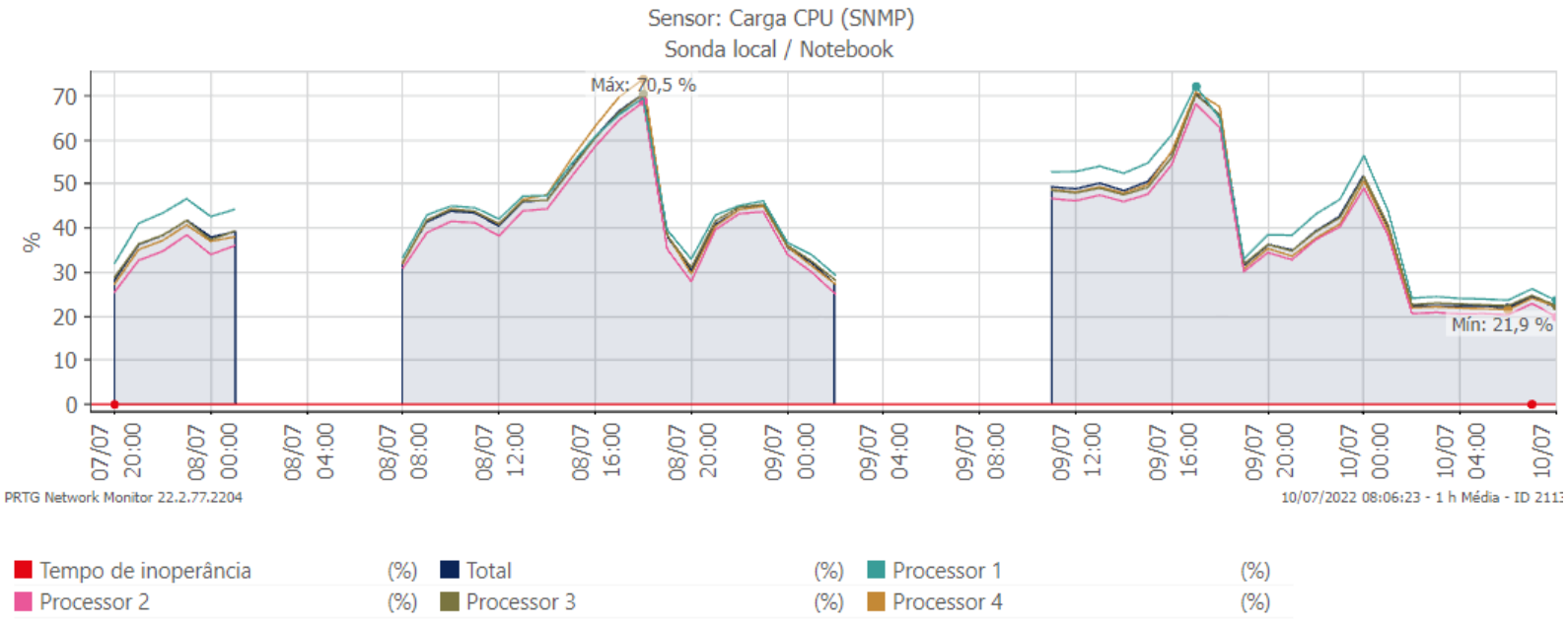

Gráfico 17: Monitoramento Carga CPU (SNMP) Notebook

6. Wireshark

Utilizamos o Wireshark para capturar alguns pacotes e analisarmos a ocorrência de alguns protocolos como ARP e SNMP, bem como ações de gerência de rede. As medições não foram efetuadas durante todos os dias, apenas algumas vezes isoladas, umas ao dia 9, outras ao dia 10/07/2022.

6.1 ARP

O protocolo Arp é um padrão utilizado para a conversão de endereços da camada de rede (IPv4) para a camada de enlace (MAC address). (Em parênteses nossos casos de estudo).

Inicialmente gostaria de ressaltar que a utilização do comando diretamente no cmd não é necessária para observarmos a ocorrência do protocolo SNMP, as tabelas armazenadas nos dispositivos apresentam a ligação IP address - MAC address - TTL, onde a última indica o Time-To-Live que quando acaba faz com que o protocolo ARP entre em ação novamente para aquele endereço IP, em nosso caso apenas utilizamos para que pudéssemos observar alguns outros fatores tais quais o broadcast durante a identificação inicial de valores na rede.

Primeiramente utilizamos o comando "netsh interface ip delete arpcache" diretamente no cmd para que o cache armazenado pelo protocolo ARP até o momento fosse excluído, logo após a execução do comando, o computador (192.168.0.103) faz um broadcast na rede (destino ff:ff:ff:ff:ff:ff) a todos conectados, ele quer receber as informações dos IP's e armazená-las, primeiramente pergunta sobre quem tem o IP 192.168.0.1, que é o padrão do roteador, e então é respondido com o endereço MAC. Após o primeiro passo completo o computador pergunta: "quem tem o IP 192.168.0.101 (Galaxy M52)?" quatro vezes
seguidas, sempre fazendo broadcast na rede, o IP com final 101 ainda não enviou resposta mas o computador já começou a perguntar para outro ip qual o MAC conectado (192.168.0.100), e nisso ele é respondido com outra pergunta, o IP do notebook (finalizado em 100) pergunta exatamente o oposto; que o computador responda para ele quem está conectado no IP 192.168.0.103, então assim como quer obter sua resposta, o computador envia seu MAC (a8:a1:59:09:29:4e) como para o notebook e logo em seguida recebe a resposta sobre o MAC do mesmo (80:30:49:3d:95:4f). Para os próximos passos, os celulares não fazem a pergunta sobre o endereço do computador, mas assim que fosse necessário enviar algum dado teriam de fazê-la antes, entretanto todos enviam suas respostas à pergunta efetuada.

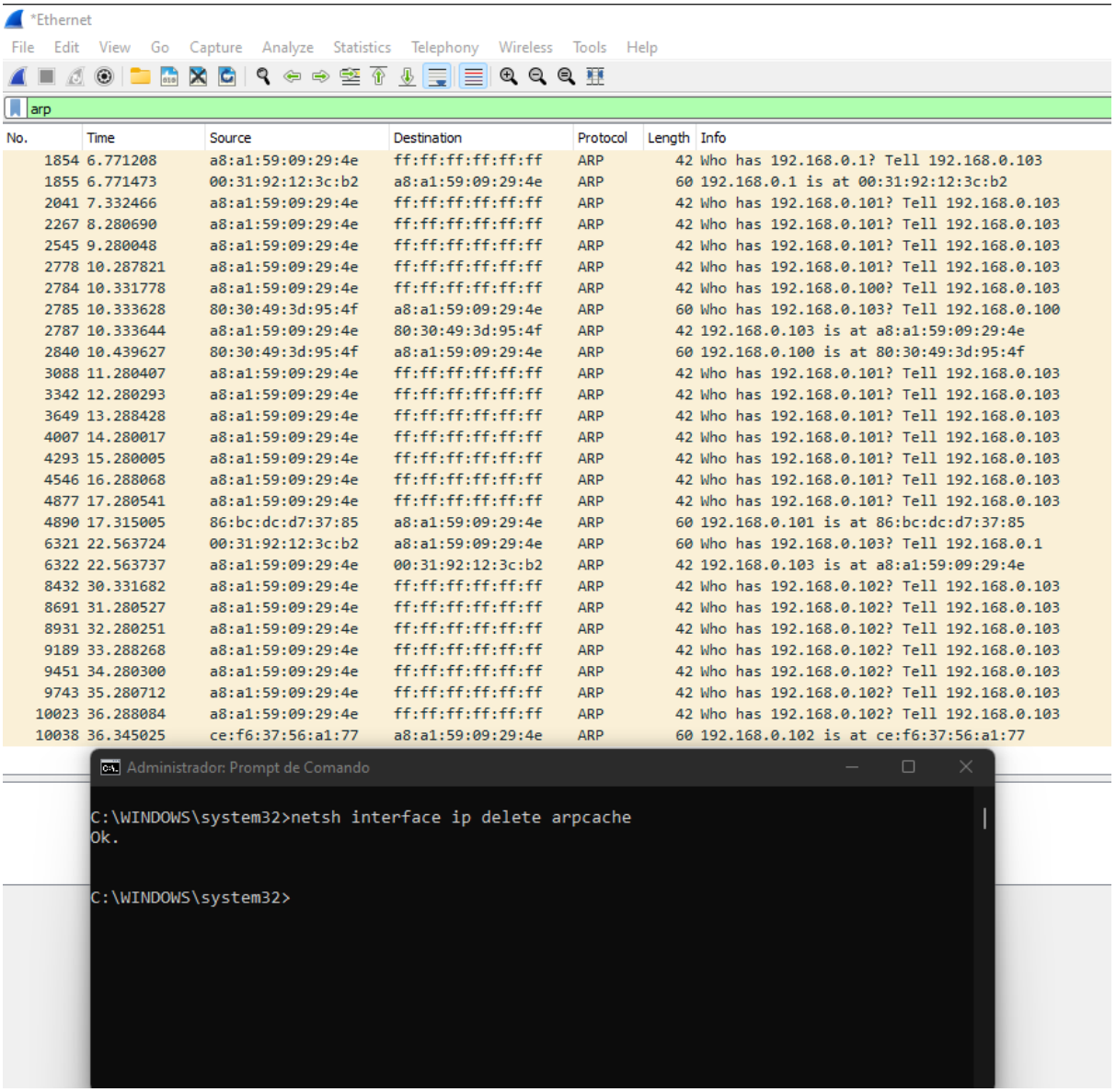

Imagem 2: Filtro “arp” utilizado na medição do Wireshark

6.2 SNMP

“O protocolo SNMP é utilizado para transmitir informações e comandos entre uma entidade gerente e uma agente executando em nome dessa entidade dentro de um dispositivo de gerência de rede” (KUROSE e ROSS, 2017).”

Aqui podemos observar algumas ocorrências de pedidos e respostas de dados por conta de nosso programa gerente (encontrado no computador 192.168.0.103), mesmo que o PRTG não obrigue a instalação de agentes, podemos observar claramente que o notebook

(192.168.0.100) está funcionando como tal, por outro lado, nosso gerente, encontrado no desktop, faz requisições por meio de Get-Request e GetBulkRequest, o primeiro pede ao agente que recupere o valor de uma lista de variáveis e/ou sua descrição, já o segundo funciona de forma parecida, tendo o papel de agir como vários pedidos de Get-NextRequest, mas fazendo a requisição de todos os valores encontrados em folhas a partir de um certo local especificado. Como podemos notar, o processo de gerência apenas tem a permissão de ler os dados do notebook para que possamos monitorá-los, ele não utiliza Set's em nenhum momento.

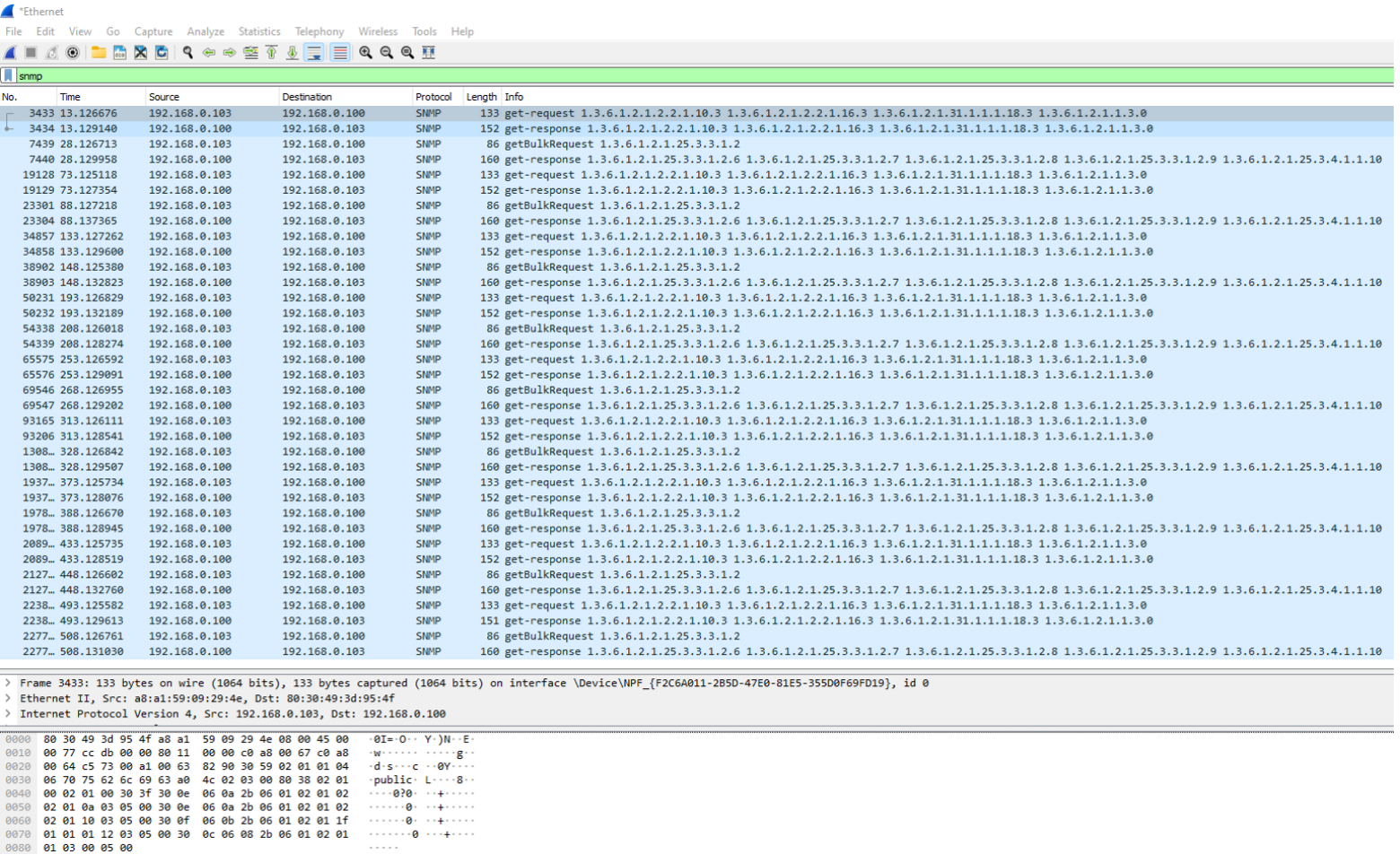

Imagem 3: Filtro “snmp” utilizado nos dados adquiridos via Wireshark

6.3 TCP

Primeiramente buscamos um site que ainda funcionasse com o antigo protocolo HTTP visto que a grande maioria deles hoje em dia utiliza HTTPS na camada de aplicação por conta da maior segurança oferecida. Durante a busca encontramos o http://www.quadrix.org.br, site que utilizamos para avaliar os critérios requisitados.

6.3.1 Criação da Conexão

Sabemos que o protocolo TCP atua na camada de rede, ele precisa estabelecer a ligação entre o computador e o site pretendido antes de qualquer Get ou outro pedido que HTTP possa pedir. TCP utiliza um método de three-way handshake para estabelecer a conexão, conseguimos notar isso com o ativamento de algumas flags nas três primeiras comunicações. Na primeira mensagem, o cliente (computador) precisa emitir um aviso de conexão, simbolizado com a flag SYN (ainda sem dados), então o servidor responde com outra mensagem também sem dados mas com as flags SYN, ACK ativadas, funcionando como uma confirmação da conexão, então o cliente, mais uma vez sem dados responde , agora com um ACK para simbolizar o recebimento das informações do servidor e então a conexão está inicializada.

Durante este estágio, é importante notarmos alguns passos para comparar posteriormente com o UDP, aqui podemos ver que o protocolo TCP busca manter a “Confiabilidade”, isto é, garantir que o destino e a origem estão corretos, é “Orientado a Conexões” pois precisa estabelecer uma antes de ocorrer a comunicação efetiva de dados e mantém o “Controle de Fluxo”, ou seja, um equipamento envia um pacote e aguarda uma resposta dizendo que o pacote chegou íntegro.

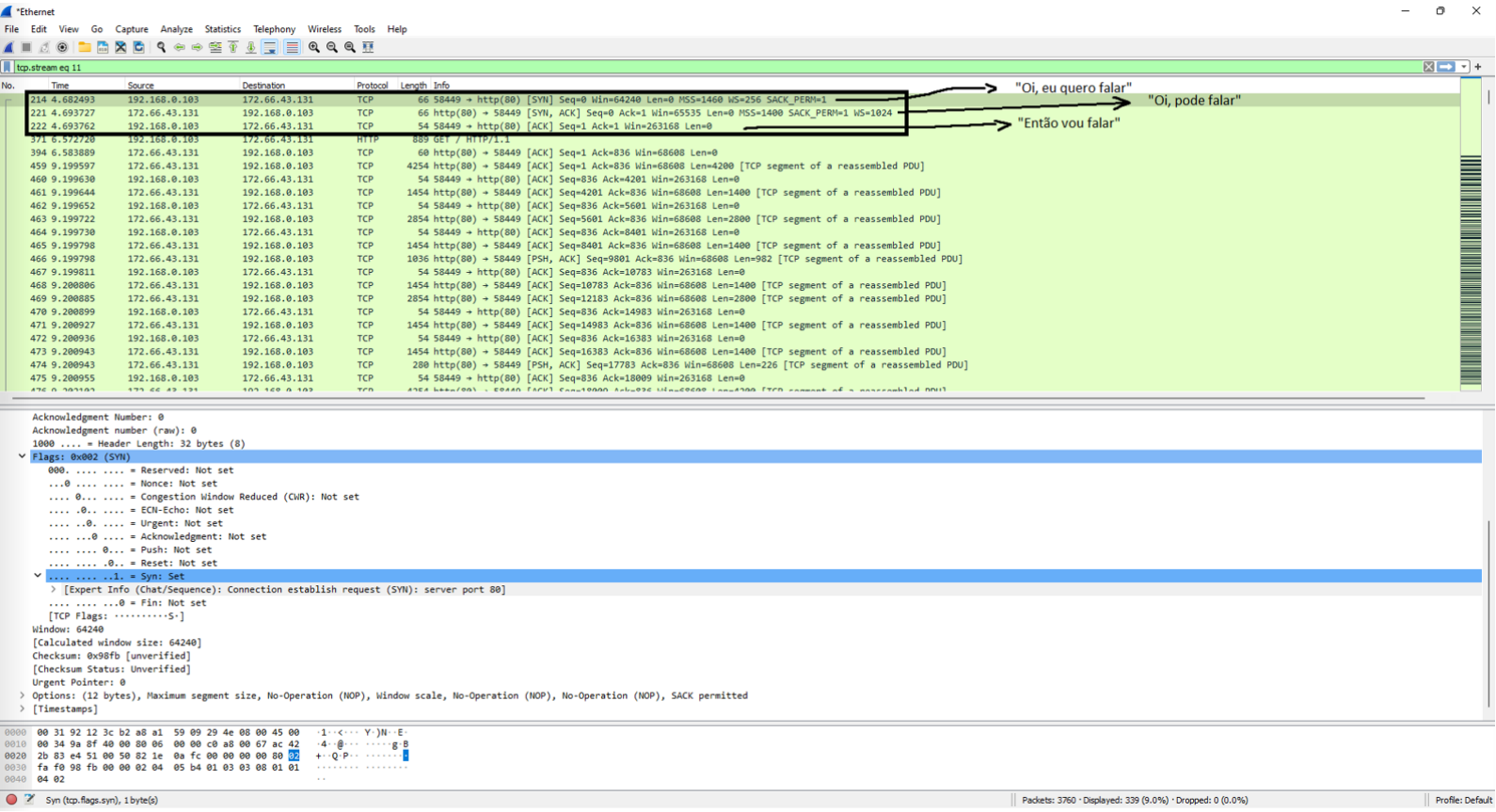

Imagem 4: Filtro "tcp.stream eq 11" utilizado nos dados adquiridos via Wireshark, analisando o estabelecimento da conexão

6.3.2 Transferência de dados

Após a inicialização da conexão ter ocorrido com sucesso, os pacotes contendo dados podem começar suas transmissões. Podemos notar na Imagem 5 que o protocolo HTTP inicialmente pede pelo URI (Uniform Resource Identifier) completo do host, isto é, seu identificador único com protocolo, localização e nome do recurso. Interessante notar também que o HTTP estipula um frame que irá manter a resposta (indicar se tudo chegou certo ou não), esse frame é mantido no futuro e reservado para uma resposta, que em casos afirmativos tem ou código 200 ou 206 (no caso de nossa print 200).

Analisando mais adentro do protocolo TCP podemos ver que em cada frame enviado temos indicadores de qual será o próximo bloco, sendo assim mostramos outra propriedade

do TCP, ele garante a “Entrega Ordenada”, mesmo que ocorra um atraso na rede os pacotes ao chegarem no destino podem ser remontados na ordem correta por conta dos identificadores. (Ver imagem 6)

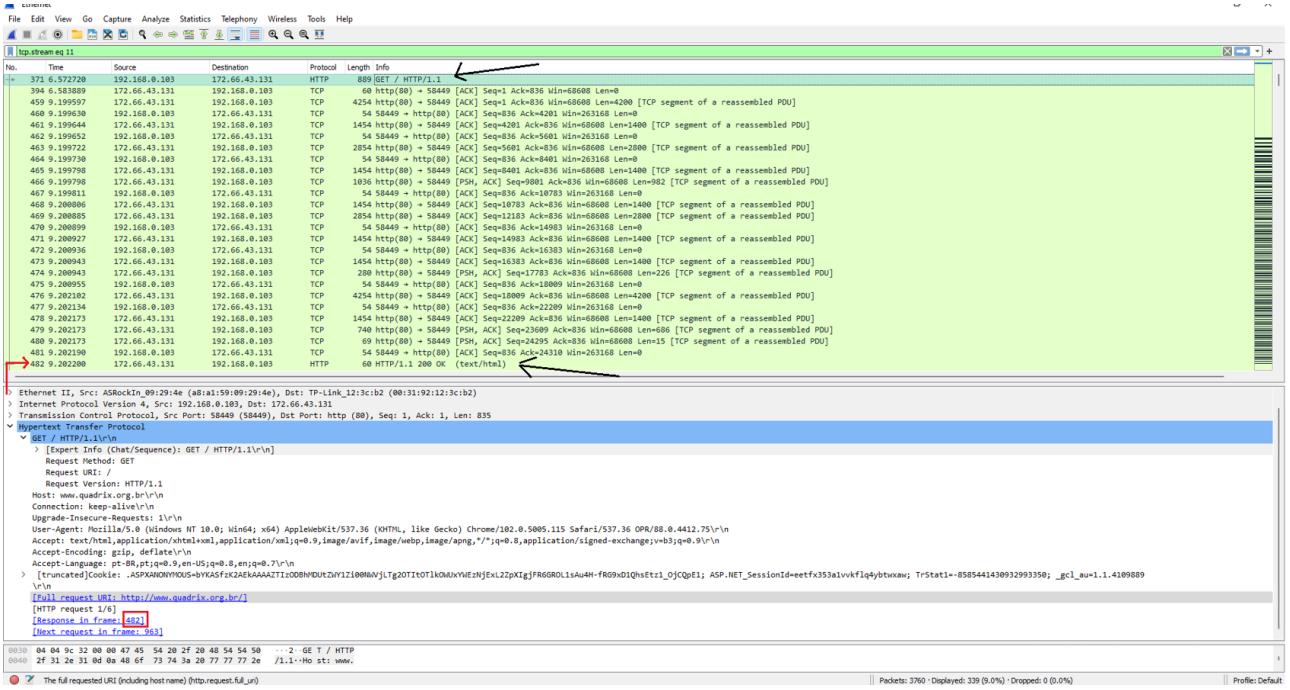

Imagem 5: Filtro “tcp.stream eq 11” utilizado nos dados adquiridos via Wireshark analisando transferência de dados

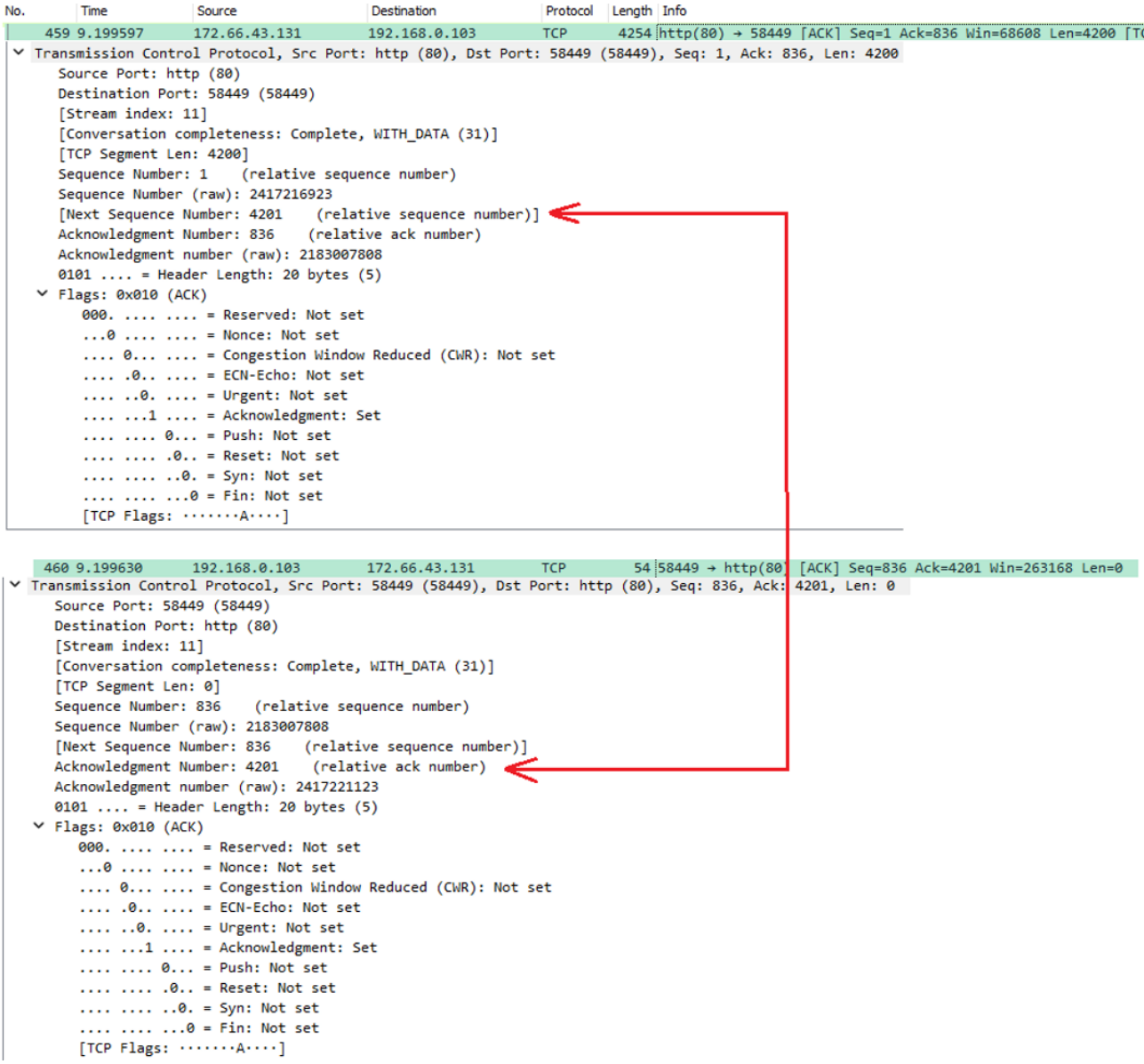

Imagem 6: Comparação entre 2 frames seguidos exemplificando a ordenação

6.3.3 Liberação de Conexão

Enquanto estava no site ao final, notamos que o TCP identificou vários envios de pacotes fora de ordem, e logo após isso resolvemos fechar o site, fizemos um Get imediatamente antes de fechar mas não demos tempo para que ele buscasse arquivos. TCP utiliza de um four-way handshake para sua finalização, embora tenhamos capturado uma variação desse four-way, comumente ele se dá da seguinte maneira: primeiro passo uma flag FIN é enviada pelo cliente, indicando que se quer finalizar a conexão, então é respondido

pelo servidor com um ACK, confirmando, logo após isso o servidor então envia um FIN e o cliente responde com um ACK dando por encerrada a conexão. Em nosso caso encontramos uma variante, mas também pertencente ao protocolo TCP, que funciona da seguinte maneira: o cliente envia diretamente um FIN, ACK ao servidor, que responde com outro FIN, ACK e então para que o computador garanta ao servidor que recebeu a mensagem envia um último ACK.

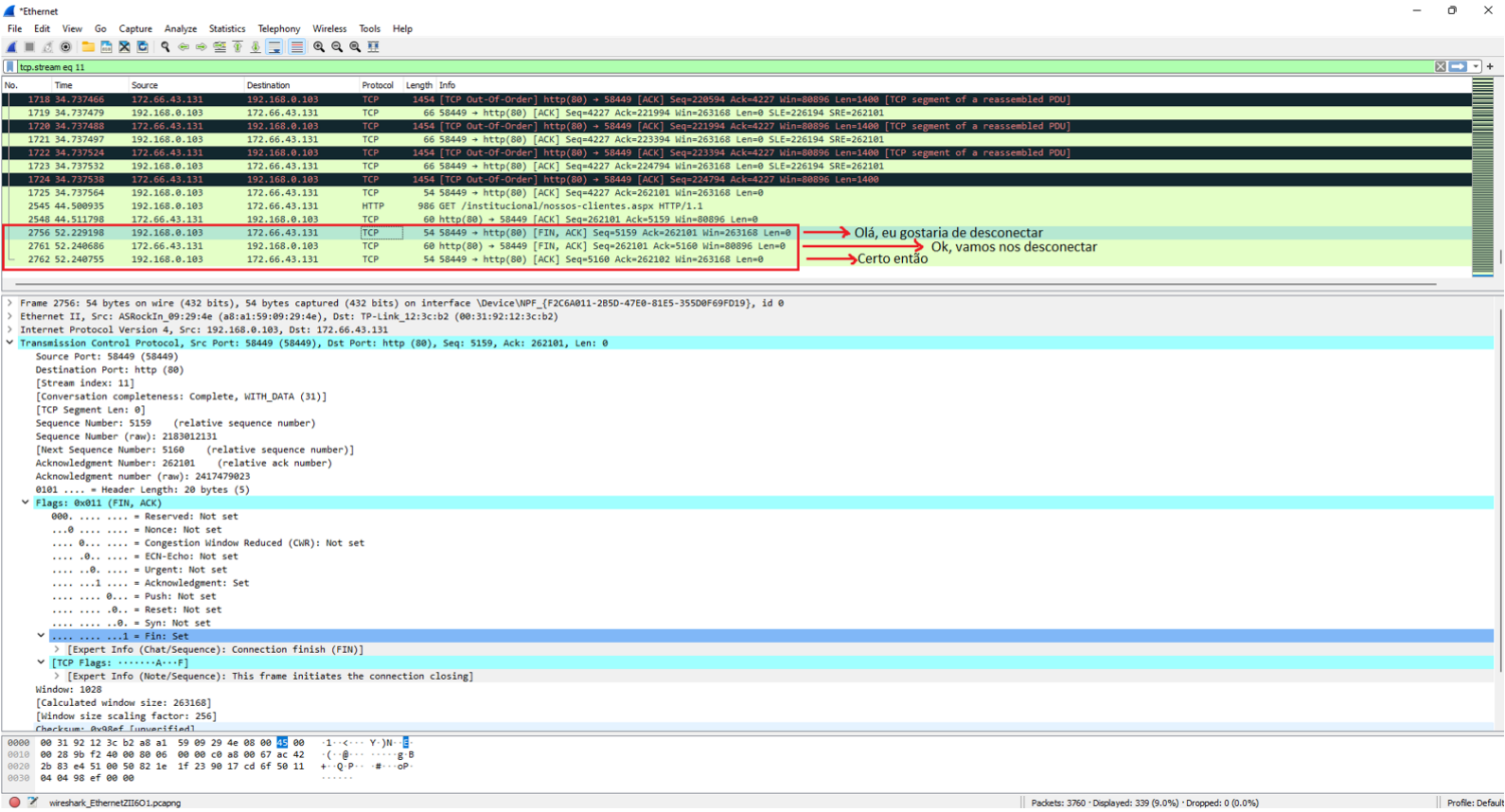

Imagem 7: Filtro "tcp.stream eq 11" utilizado nos dados adquiridos via Wireshark analisando liberação de conexão

6.5 UDP

UDP (User Datagram Protocol) agora que já vimos como o TCP é podemos afirmar, é um protocolo muito mais simples e com menores campos, UDP deixa para que a camada de aplicação peça a retransmissão de dados caso necessária, sendo assim muito útil para programas em tempo real, como por exemplo videochamadas onde não se pretende pedir a retransmissão um frame perdido, por conta desse fator ele é considerado mais eficiente do

que o TCP para necessidades instantâneas, também pelo fato de que diferentemente do anterior, ele não ser “orientado a conexões”, isto é não há a necessidade de um “handshake” antes do envio de dados, o que pode nos levar a outro rumo, como por exemplo a inundação de pacotes UDP em ataques DDOS cujo campo não será abordado neste trabalho.

Para exemplificar o UDP enquanto o Wireshark monitorava a rede acabei retirando a rede Ethernet, e reinserindo-a fazendo assim com que ocorresse a emissão de protocolos DHCP (protocolo que oferece configuração dinâmica de IP’s, máscaras de sub-rede, gateway padrão, etc), que operam da seguinte maneira: Quando um computador se conecta a rede o host/cliente DHCP envia um pacote UDP em broadcast com um pedido DHCP para a porta 67, então espera sua resposta na porta 68. Neste caso, não faz sentido precisar estabelecer uma ligação TCP com handshaking tanto para conexão quanto para liberação para realizarmos um simples pedido de endereço IP.

Podemos ver os campos do cabeçalho UDP que são apenas 4, Porta de origem, Porta de destino, Comprimento da mensagem, Checksum (para “garantir” a integridade dos dados).

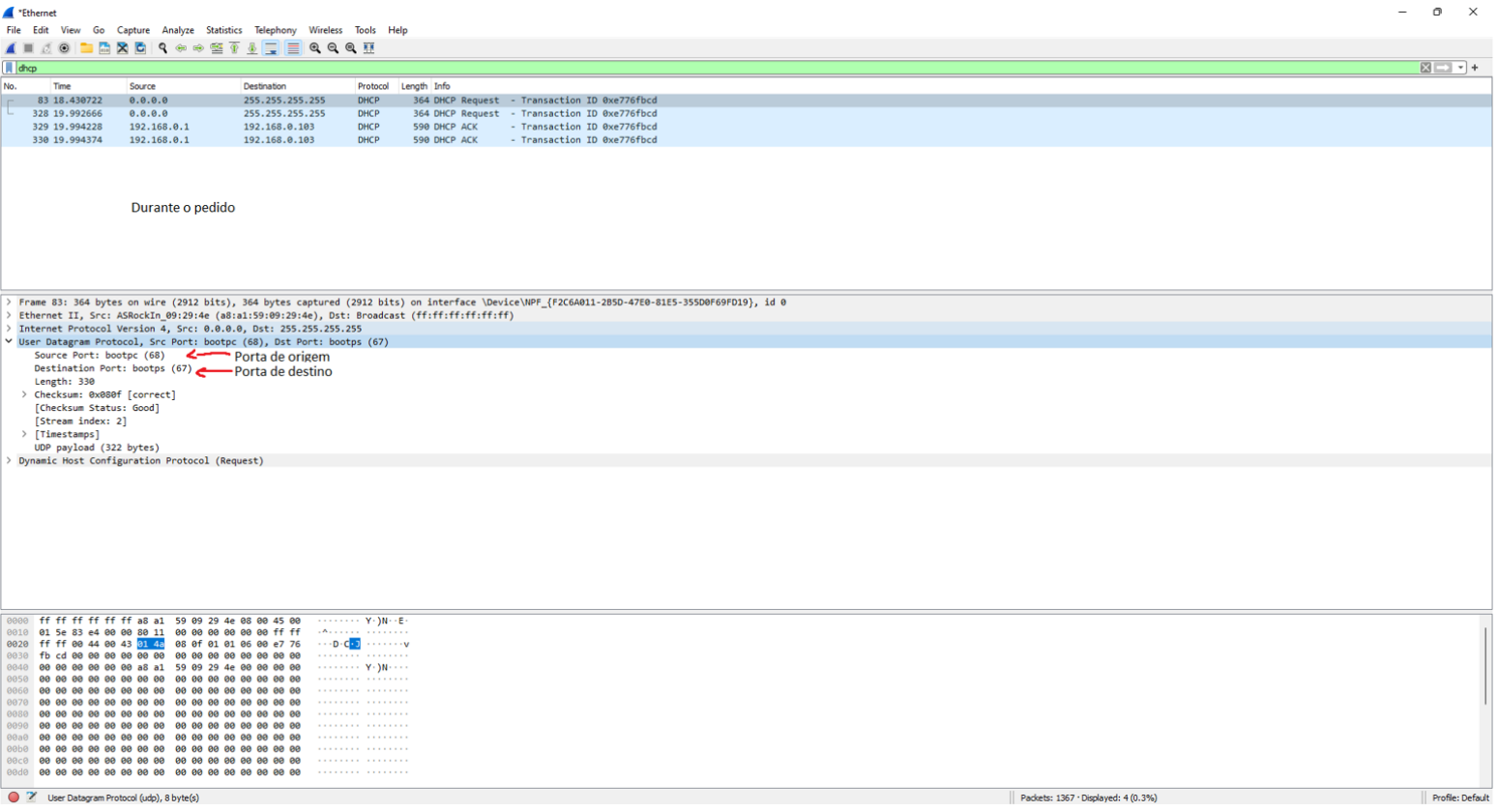

Imagem 8: Filtro “dhcp” utilizado nos dados adquiridos via Wireshark analisando o Request DHCP com UDP

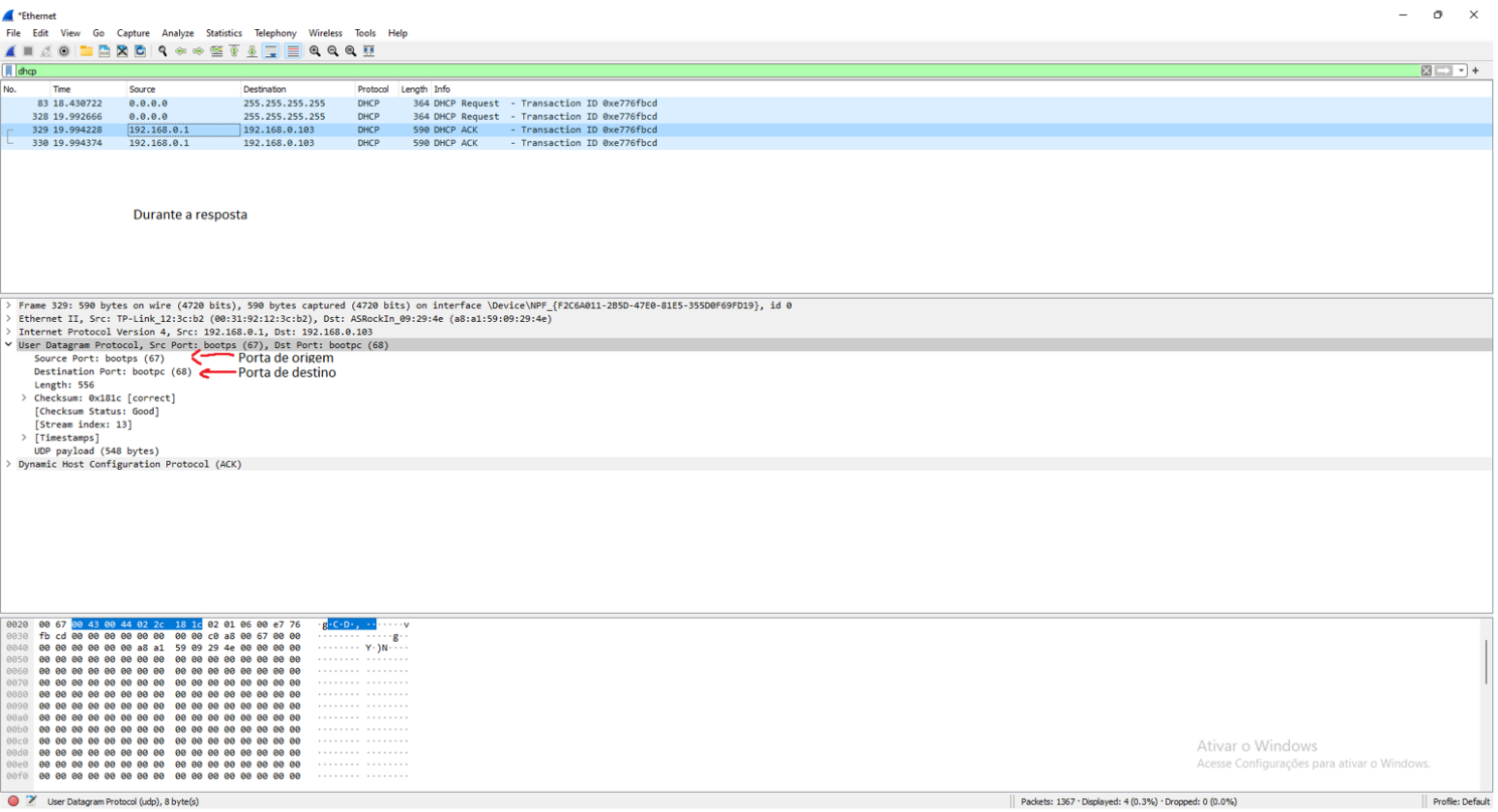

Imagem 9: Filtro “dhcp” utilizado nos dados adquiridos via Wireshark analisando resposta com UDP

7 Conclusão

Para a realização do trabalho, além da busca pelas informações já passadas em sala de aula, foram necessárias pesquisas para o melhor compreendimento dos protocolos de redes e quais suas camadas de atuações, bem como diferentes protocolos para melhores exemplificações tiveram de ser completamente entendidos. Ao mesmo passo que a visualização do protocolo ARP em prática serviu para dirimir alguns pontos de dúvidas anteriores.

O monitoramento de rede permitiu uma avaliação e facilitou a visualização do tráfego na rede e algumas variáveis que acabam influenciando determinados sensores, em nosso caso por ter usado o PRTG, também, para ser capaz de monitorar o notebook durante a execução do trabalho foi necessário algumas atualizações de configurações da rede de mesma forma a instalação de dois serviços nativos do Windows acerca de SNMP, os quais não vêm instalados a priori foram necessárias.

8 Referências

KUROSE, James F.; ROSS, Keith W. Computer Networking: A Top-Down Approach. 6a edição. Harlow: Pearson Education Limited, 2013.

DYNAMIC HOST CONFIGURATION PROTOCOL. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Dynamic_Host_Configuration_Protocol&oldid=59637441>. Acesso em: 10 jul. 2022.

PAESSLER, PRTG Network Monitor v22. Acesso em: 6 de Julho de 2022.

ATAQUE UDP FLOOD. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2018. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Ataque_UDP_flood&oldid=52556447>. Acesso em: 10 jul. 2022.

TCP/IP. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2022. Disponível em: <https://pt.wikipedia.org/w/index.php?title=TCP/IP&oldid=63882068>. Acesso em: 8 jul. 2022.

STASHCHUK, Bogdan. Analyzing UDP in Wireshark. Youtube, 3 de out. de 2021, Disponível em: https://www.youtube.com/watch?v=ToLao3kSBFA

ADDRESS RESOLUTION PROTOCOL. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2020. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Address_Resolution_Protocol&oldid=59829718>. Acesso em: 9 nov. 2022.