|  |
| --- |

**UTILIZANDO FILTROS NO WIRESHARK**

Profa. Raiane Santos

* criado em 1998 por **Gerald Combs;**
* já foi chamado de **Ethereal**;
* Semelhante ao tcpdump;
* classificado **como a melhor ferramenta de segurança de rede;**
* reconhece 1038 protocolos;

1. **Introdução**

Como visto anteriormente, o Wireshark é um software analisador de pacotes de tráfego de redes de computadores. Ele analisa os protocolos permitindo que você capture o tráfego da rede online através de sua interface e, também analise esses dados, inclusive de forma oflline.

Os sniffers (farejadores) são amplamente utilizados por administradores de redes com a finalidade de traçar um perfil de tráfego da rede e futuramente identificar possíveis conexões suspeitas.

Você lembra da tríade de segurança **(confidencialidade, integridade e disponibilidade)** já conhecida por nós? Então, o que ela teria a ver com Wireshark e análise de pacotes? Bem, de início já nos referimos ao princípio da confidencialidade (garantia de que apenas as pessoas explicitamente autorizadas podem ter acesso à informação), pressupondo que normalmente ninguém utilizaria um sniffer na sua rede e que o seu uso seria apenas quando explicitamente autorizado, o que em determinadas situações pode não ocorrer.

1. **Sistemas para Detecção e Prevenção de Invasões**

Vocês certamente já ouviu falar do programa Snort, certo? Então, é um programa notadamente fácil de obter e instalar, mas um pouco mais complexo na aplicação. As dificuldades encontradas, por exemplo, no Snort são dificuldades enfrentadas diariamente por profissionais de segurança que lidam com IPS e IDS.

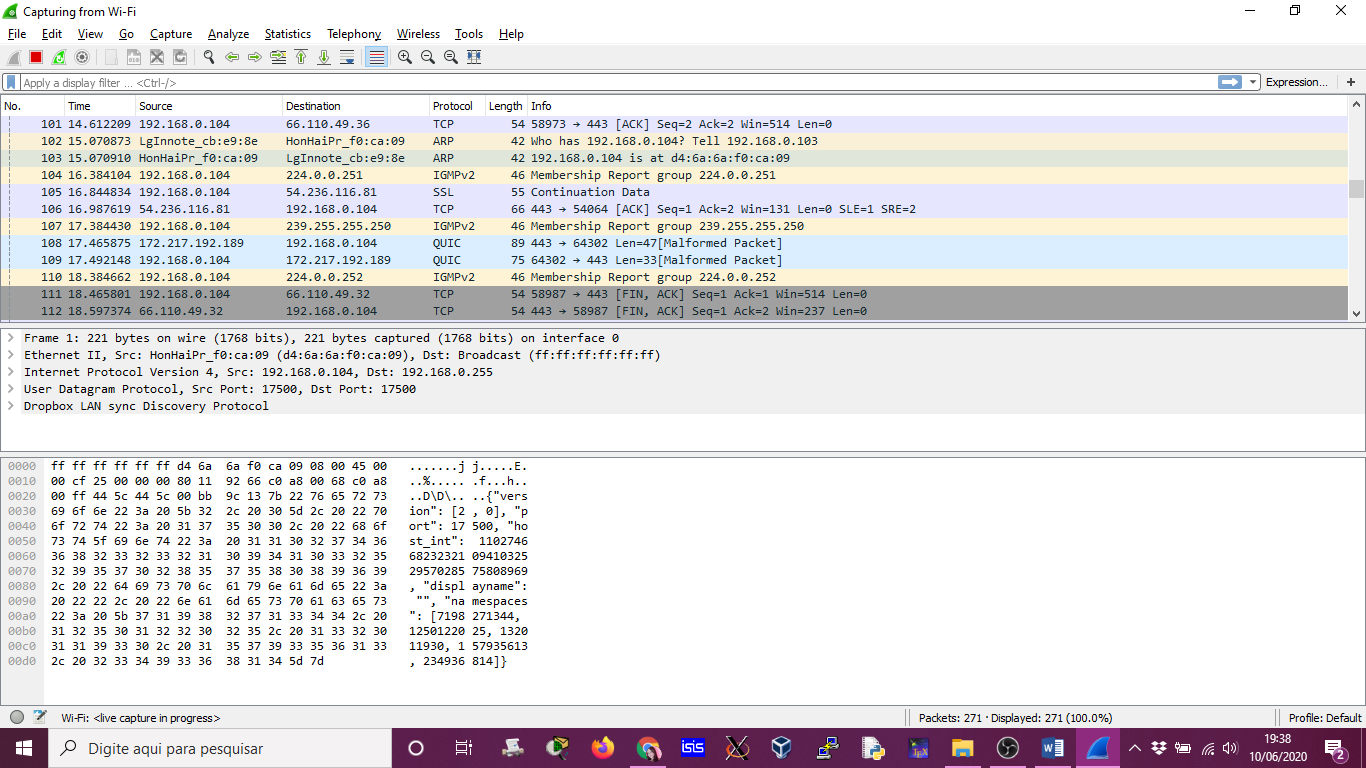
E agora você se pergunta o que é um IDS/IPS. Basicamente IDS (Intrusion Detection System) são sistemas utilizados para a detecção de intrusões em um sistema, de forma breve, apenas sinaliza para o administrador que algo anômalo/ruim foi detectado na rede. Essa detecção normalmente é realizada com base no que o IDS já conhece, por exemplo, você treinaria o sistema para reconhecer ataques do tipo botnet, ou ainda para reconhecer o tráfego notadamente “normal”, tão logo, tudo que difere do que o sistema entende por normal, ele classifica como anômalo/ataque. Enquanto os IPS (Intrusion Prevention System) são sistemas responsáveis por tomar uma ação após a detecção de algo ruim na rede. Por exemplo, em um possível ataque o IPS contra-ataca.

;)

E a pergunta que não quer calar: **Wireshark pode ser considerado um IDS?** 

SIM!!!

O Wireshark pode funcionar como um IDS baseado em assinatura, que detectará o que você desejar nos pacotes capturados, desde que tulizando os filtros corretos. Ele ainda poderia funcionar como observador do comportamento da rede.



Origem do tráfego

Destino do tráfego

Protocolo

Informações adicionais

Tamanho do pacote

Ordem do pacote

Ah, e antes que vocês me perguntem, o Wireshark pode ser manipulado por *malwares* presentes na captura? Ou ainda malwares como rootkits podem manipular os dados capturados, alterando a captura?

A resposta é Não.

1. **Análise de Pacotes**

Como já detalhado anteriormente é muito importante conhecer as definições do modelos em camadas Modelo OSI e Arquitetura TCP/IP, logo que essas camadas representam os detalhes da captura.

Então, vamos lá?!

**Figura 1**. Informações do Pacote.

A captura dos pacotes podem ocorrer de modo promíscuo (quando a placa de rede aceita todos os pacotes chegam à interface) ou modo monitor (permite realizar captura de todo o tráfego, de todos os pontos de acesso). Na Figura 1 é possível observar as informações preliminares que a captura nos apresenta. Veja,

**[nº]**  número do pacote no arquivo de captura;

**[Time]** registro de data e hora do pacote;

**[Source (origem)]** informando o endereço da origem do tráfego;

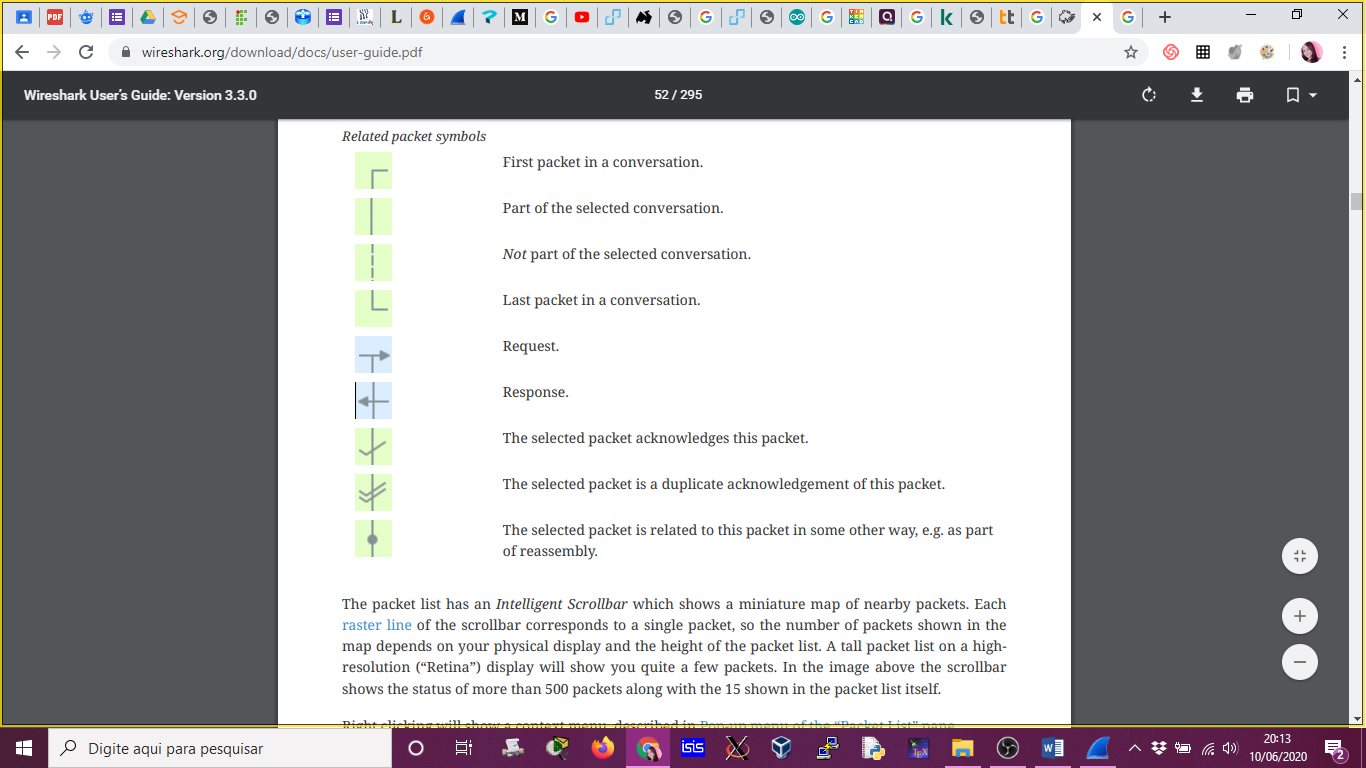
**[Destination (destino)]** o endereço para onde este pacote está indo;

**[Protocol]** protocolo utilizado;

**[Length]** comprimento do pacote;

**[Info]** informações adicionais do pacote.

Se você observar, *durante a seleção dos pacotes*, a primeira coluna mostra como cada pacote está relacionado ao pacote selecionado, de acordo com os símbolos a seguir:



Primeiro pacote da comunicação

Parte da comunicação do pacote selecionado

Não faz parte da comunicação do pacote selecionado

Último pacote da conversa

Request (requisição)

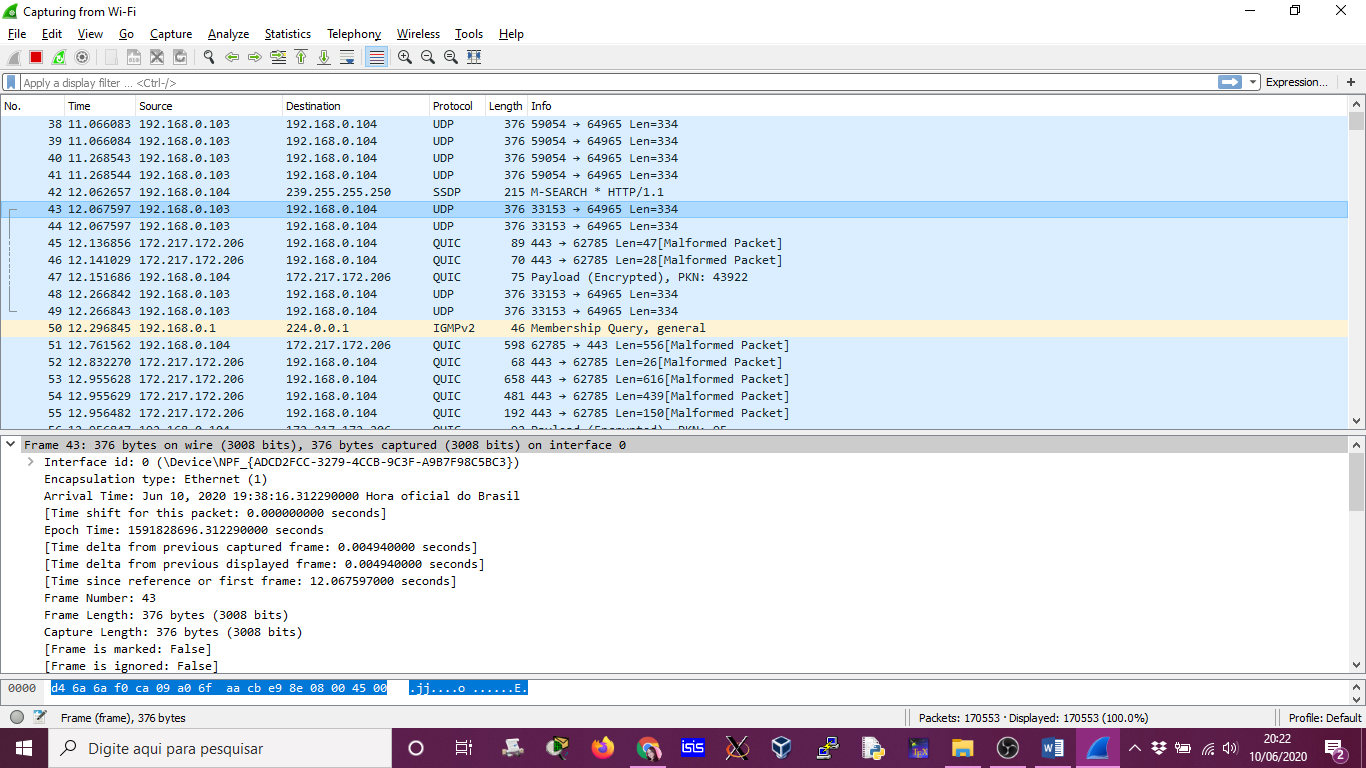
Response (resposta)

Pacote reconhecido pelo pacote selecionado

Pacote selecionado é uma confirmação duplicada desse pacote

Pacote selecionado relaciona-se com este pacote de alguma maneira

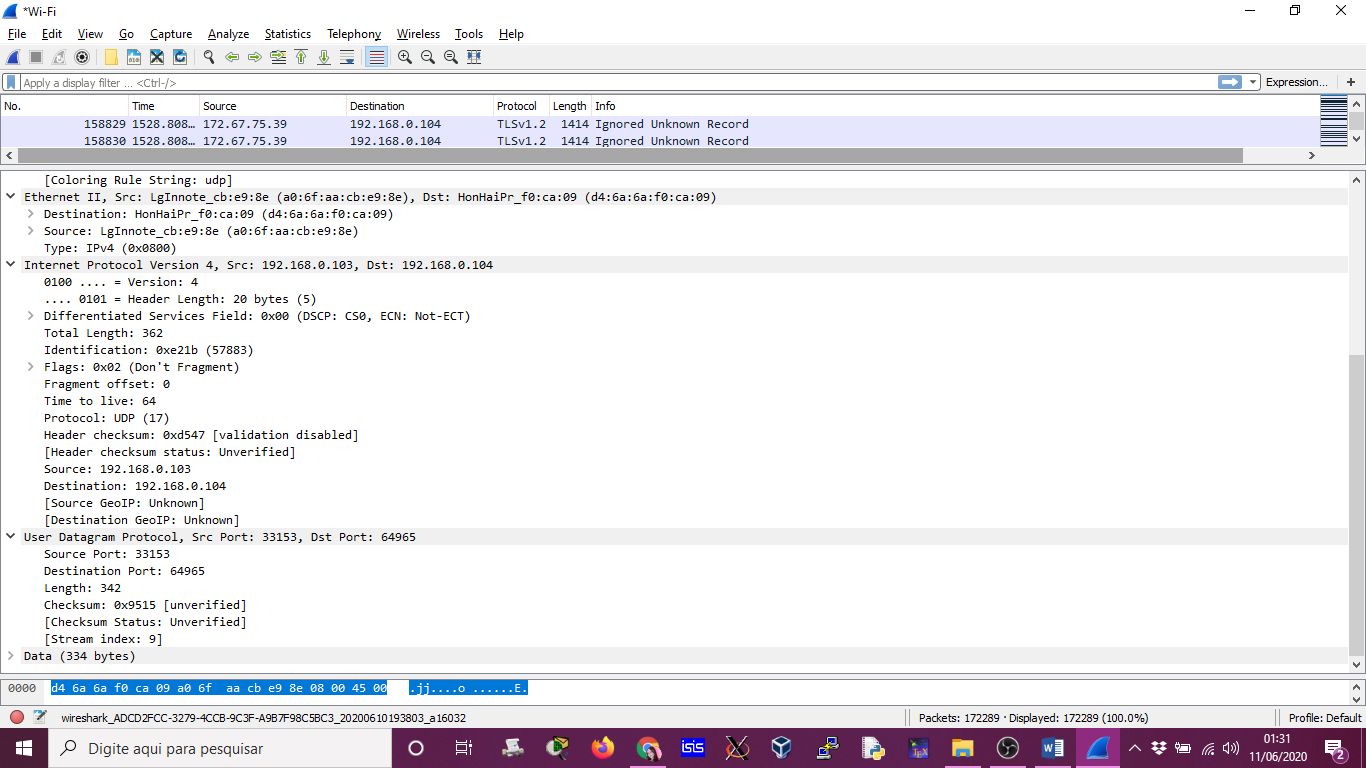
Por exemplo, na Figura 2, ao selecionar o pacote 43, que é um pacote DNS, o Wireshark mostra um símbolo na coluna nº, esse símbolo representa o primeiro pacote da comunicação. Assim como se levarmos em consideração os pacotes 44 a 47, o símbolo apresentado ao lado informado que esses pacotes não tem relação com a comunicação iniciada no pacote selecionado (pacote 43).



**Figura 2**. Informações do Pacote.

1. **Protocolos e portas**

Se navegarmos nas informações das camadas, observa-se o detalhamento da camada de transporte, conforme Figura 3. Em relação aos protocolos e suas respectivas portas, tanto o TCP quanto o UDP são utilizados para transmitir as informações, e para a comunicação se efetivar é necessário uma porta de origem e uma porta de destino.



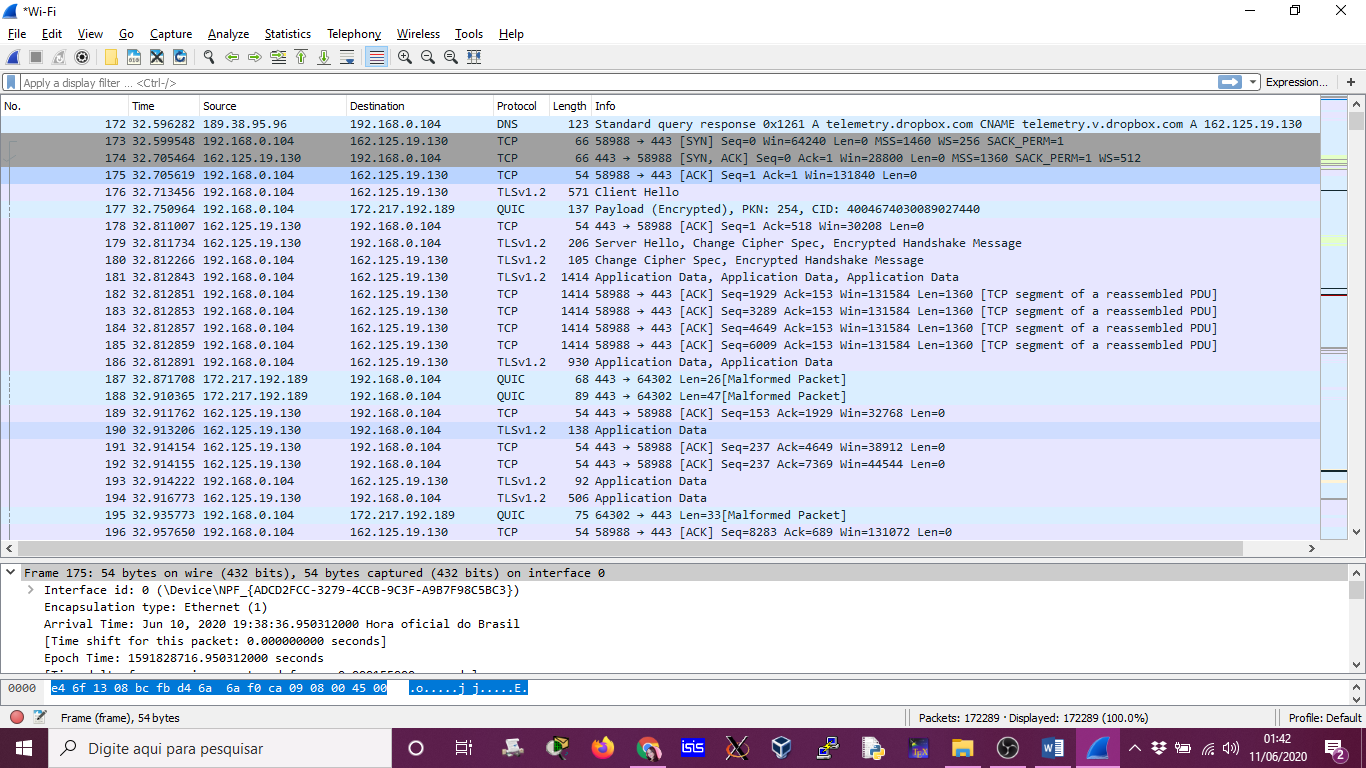
**Figura 3**. Detalhes das Camadas.

Aqui vocês já conhecem o funcionamento do TCP e UDP, mas vamos relembrar alguns pontos e assim facilitar o entendimento da análise realizada no Wireshark.

|  |  |  |
| --- | --- | --- |
|  | TCP | UDP |
| Estabelece comunicação prévia |  |  |
| É mais rápido |  |  |
| É mais leve |  |  |
| Se preocupa com a entrega do pacote |  |  |
| Faz checksums |  |  |
| Não se recupera de um erro |  |  |
| Implementa confirmação de entrega |  |  |

**Tabela 1.** TCP x UDP

Conforme tabela acima o protocolo TCP sempre vai estabelecer uma comunicação antes de enviar qualquer dado. Observe a Figura 4 e note que o TCP estabelece uma comunicação antes do envio propriamente dito. O cliene envia um SYN para abertura da comunicação, e então o servidor responde a requisição com um SYN-ACK e por fim o cliente retornar um ACK.



**Figura 4**. *Handshake* de três vias.

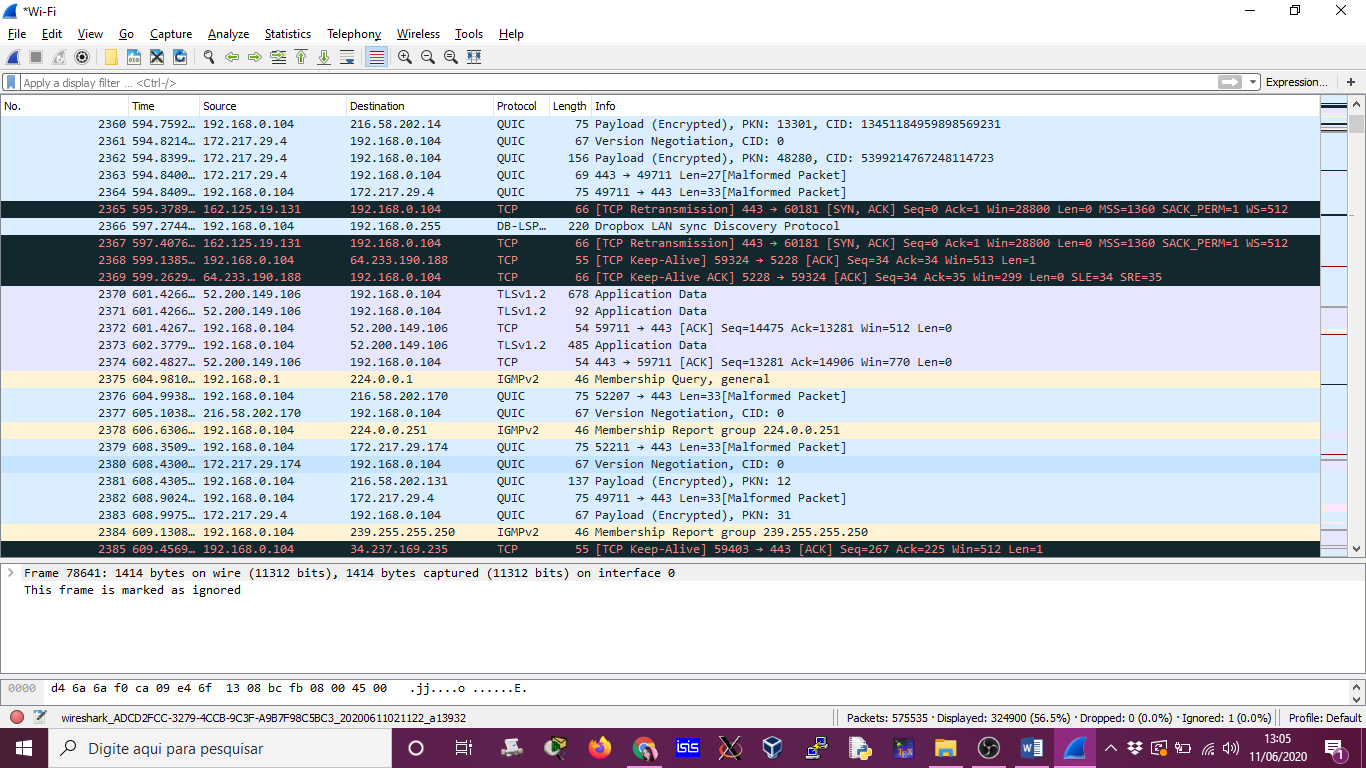
Enquanto que o protocolo UDP é mais para entregas rápidas e sem muitas exigências de segurança e confirmação.



Até aqui tudo bem

Acredito que sim, até porque os protocolos já são cartas conhecida por vocês, mas de qualquer forma, faço aqui os seguintes questionamentos: Qual a importância das portas? E uma aplicação ou serviço utiliza apenas um (TCP ou UDP) ou ambos?

Olhá só, primeiro, normalmente as aplicações/serviços trabalham com apenas um dos dois, porém serviços com DNS pode trabalhar com ambos. E, quanto as portas, veja bem, você possui um endereço para, por exemplo, envio de cartas pelos correios, os protocolos de transporte também possuem um endereço/caixa postal para encaminhamento dos seus dados que chamamos de porta. Por exemplo:



* Requisições HTTP – TCP porta 80
* Consulta DNS – UDP porta 53
* Login HTTPS – TCP porta 443
* Acessando webmail – TCP porta 110
* Enviando um email – TCP porta 25

**Figura 5.** Portas e protocolos.

Veja a Figura 5.

As portas mais conhecidas vão de 0 a 1024 e, a partir de 1025 até 49.141 chamamos de portas registradas e acima de 49.142 são as portas dinâmicas.

1. **Trabalhando com Filtros**

Inicialmente olhando aquele enorme número de pacotes capturados na sua rede, certamente você se perguntou: E agora, por onde começo? O que tudo isso pode me dizer? Como encontrar alguma informação com valor de conhecimento nesse amontoado de informações?

É certo que se você tentar realizar uma análise de tráfego sem um caminho, ou um objetivo, provavelmente ficará perdido por muito tempo. Também se sentirá assim caso não conheça os principais protocolos das camadas do Modelo OSI/ISO e suas respectivas portas. Então, já revise esse conteúdo.

Mas, olha só, o próprio Wireshark também nos possibilita criar filtros (ferramentas poderosa para selecionar determinado tráfego atráves de alguns parâmetros). Vamos conhecer quais são principais filtros que podem ser implementados.

|  |  |  |
| --- | --- | --- |
| **Filtro** | **Função** | **Como aplicar.** |
| http | Filtra por protocolo HTTP | http |
| dns | Filtra por protocolo DNS | dns |
| ftp | Filtra por protocolo FTP | ftp |
| tcp | Filtra por protocolo TCP | tcp |
| udp | Filtra por protocolo UDP | udp |
| arp | Filtra por protocolo ARP | arp |
| tcp.port | Filtrar pela porta TCP | tcp.port==80 |
| udp.port | Filtrar pela porta UDP | udp.port == 53 |
| tcp.srcport | Filtrar pela porta TCP origem | tcp.srcport == 80 |
| tcp.dstport | Filtrar pela porta TCP destino | tcp.dstport == 80 |
| ip.addr | Filtrar por IP | ip.addr == 192.168.0.105 |
| tcp.port == 80 && ip.addr | Filtrar pelo IP e Porta | tcp.port == 80 && ip.addr == 192.168.0.100 |
| ip.addr == IP and ip.addr | Filtrar tráfefgo entre dois dispositivos | ip.addr == 192.168.0.100 and ip.addr == 192.168.0.104 |
| ip.src | Filtrar por IP de Origem | ip.src == 192.168.0.1 |
| ip.dst | Filtrar por IP Destino | ip.dst == 192.168.0.105 |
| !ip.addr | Excluir IP | !ip.addr ==192.168.0.1 |
| ip.addr | Filtrar por Sub rede | ip.addr = 192.168.0.1/24 |
| eth.addr | Filtrar por MAC | eth.addr = 00:40:8f:c5:a6:78 |
| http.request | Filtrar por requisições | http.request |
| http.request.method | Filtrar por requisições GET | http.request.method == "GET" |
| http.request.method | Filtrar por requisições POST | http.request.method == "POST" |
| http.host=="" | Filtrar pelo host do HTTP | http.host=="192.168.0.1" |
| http.request.uri=="/app/login.xhtml" | Filtrar pela URI | http.request.uri=="/app/login.xhtml" |
| Frame contains”( )” | Procurar arquivos por tipo | frame contains "(attachment | pdf)" |
| tcp contains | Procurar por palavras chaves | tcp contains login |
| tcp.flags.syn == 1 and tcp.flags.ack == 0 | Detectara SYN flood | tcp.flags.syn == 1 and tcp.flags.ack == 0 |

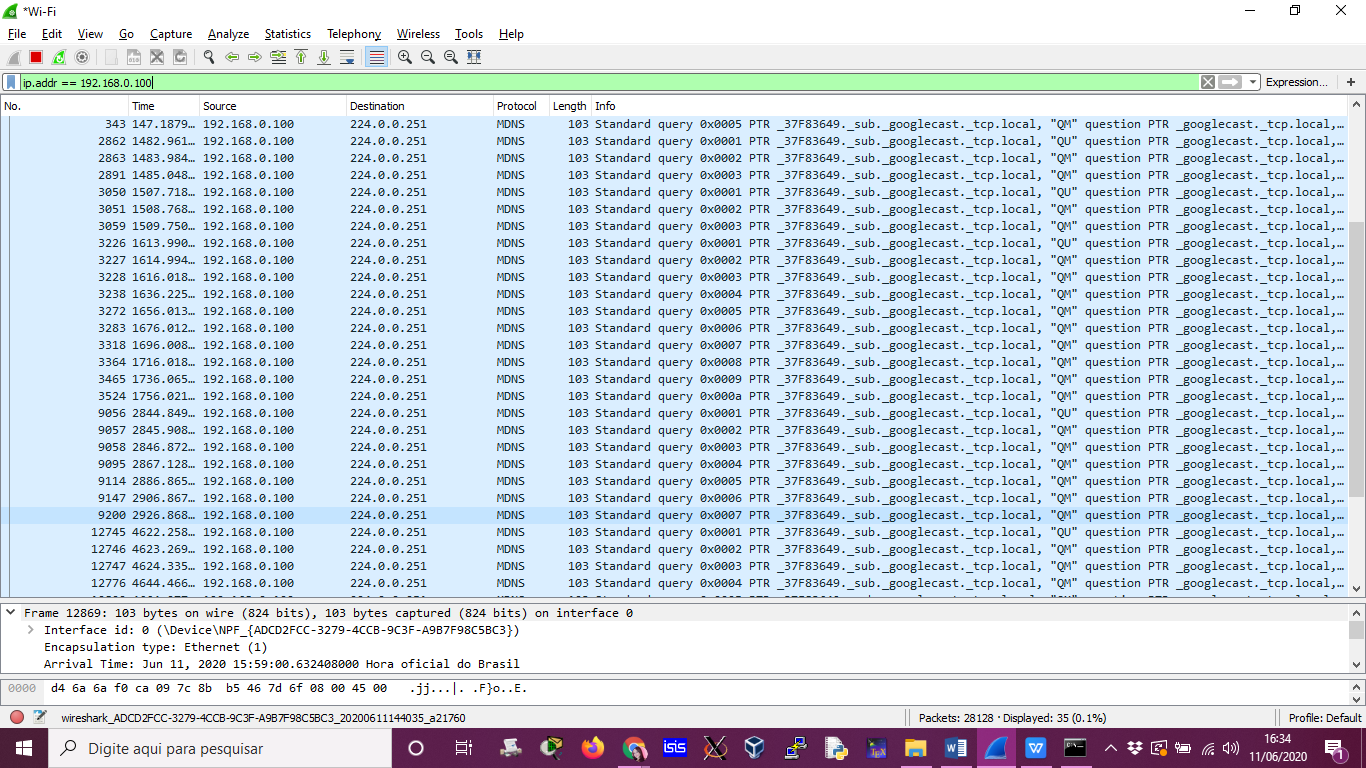
**Tabela 2.** Principais Filtros Wireshark.

Como utilizar os filtros você deve acessar a barra de filtros e digitar a forma de aplicação, por exemplo ip.addr == 192.168.0.100.

Aplicar filtros de exibição

Selecionar filtros salvos.

Limpar

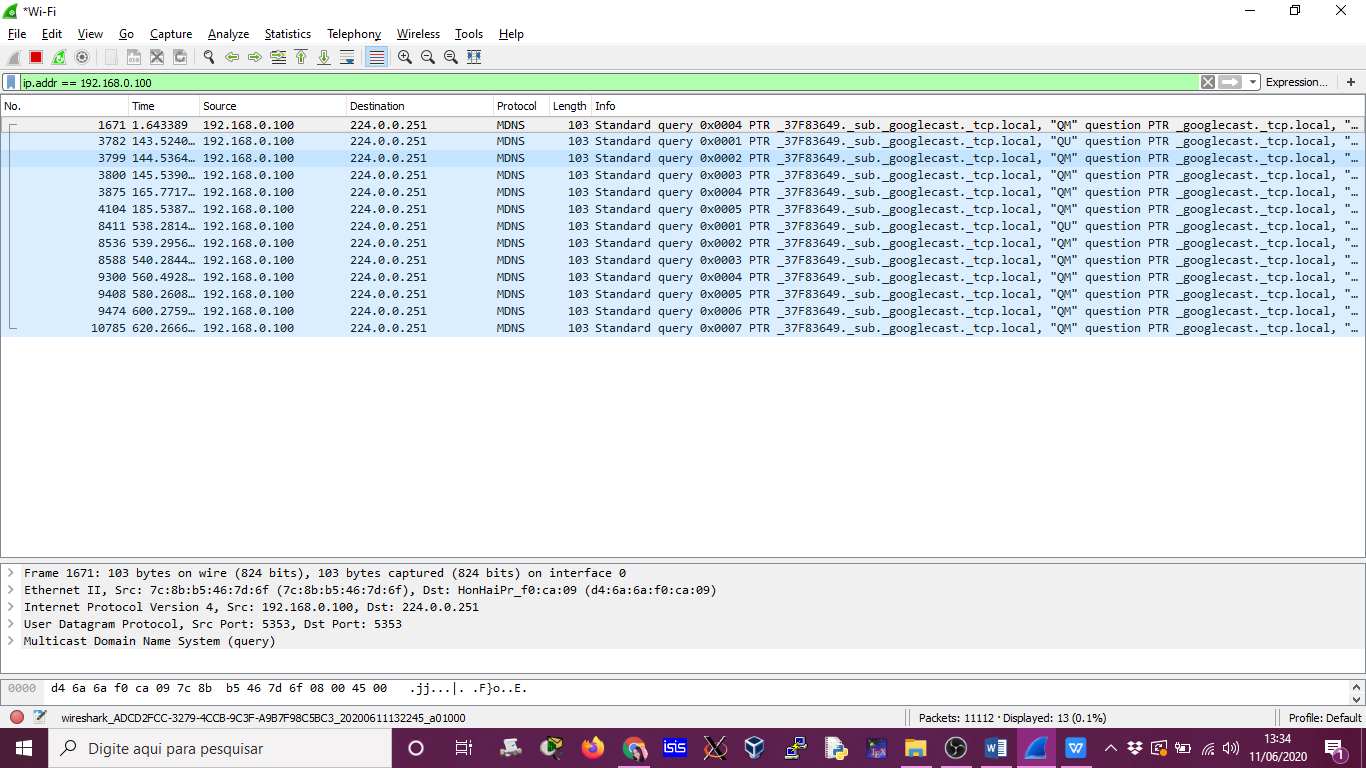


Filtro

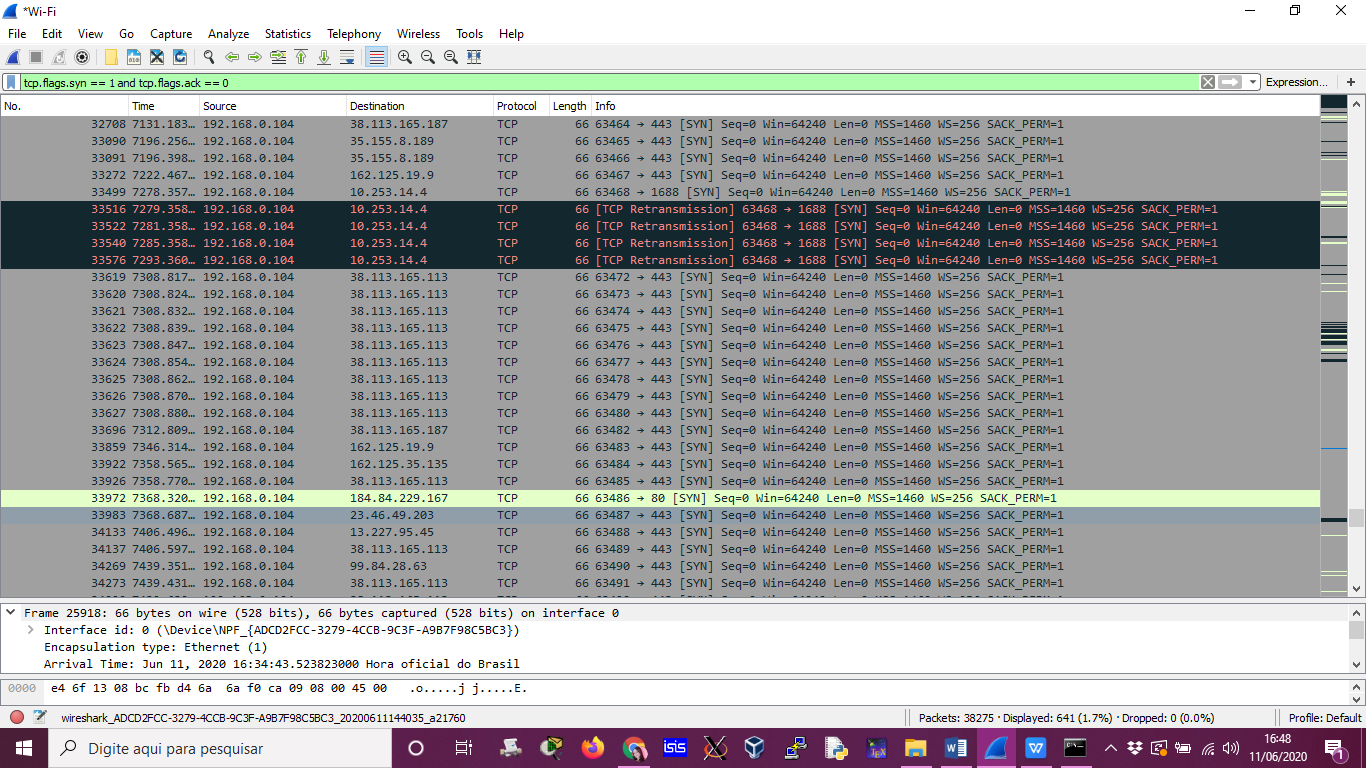
Filtros recentes

Expressões

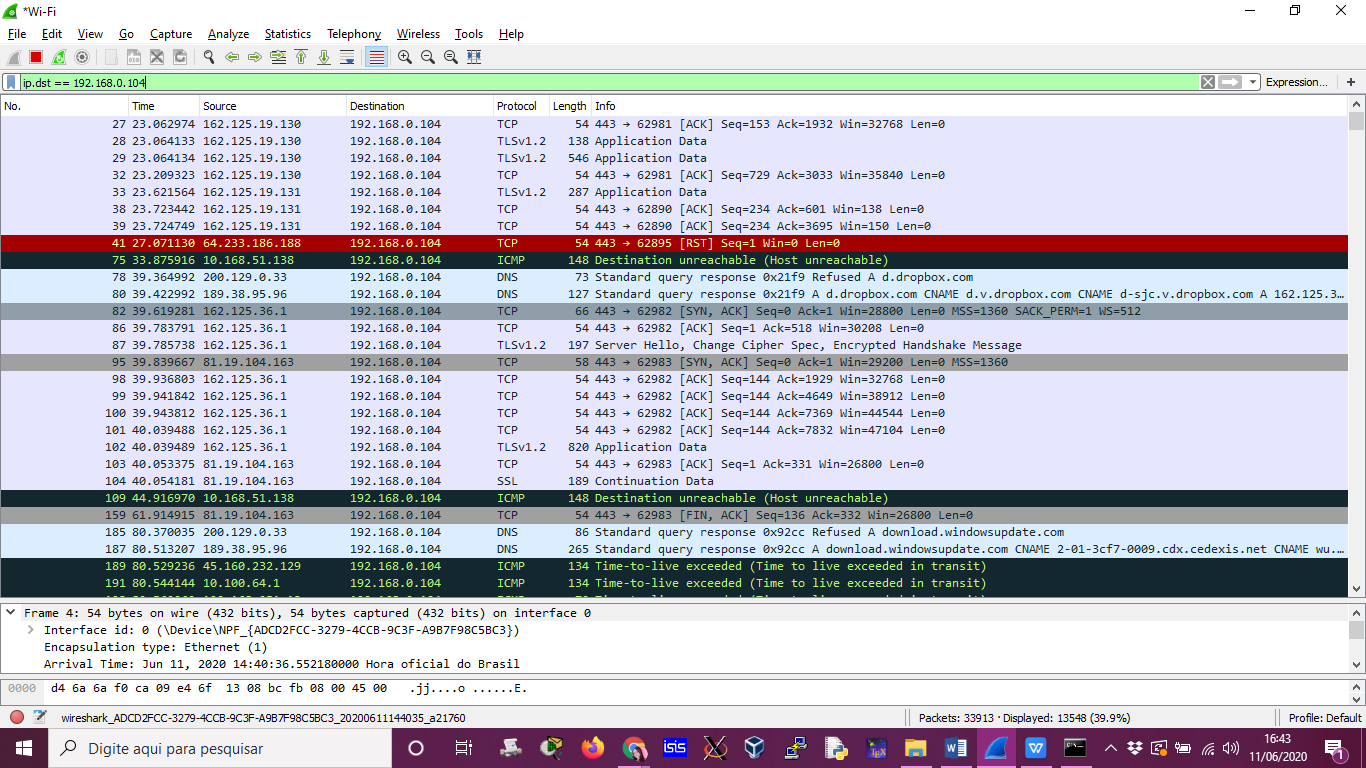
Observe os seguintes exemplos:



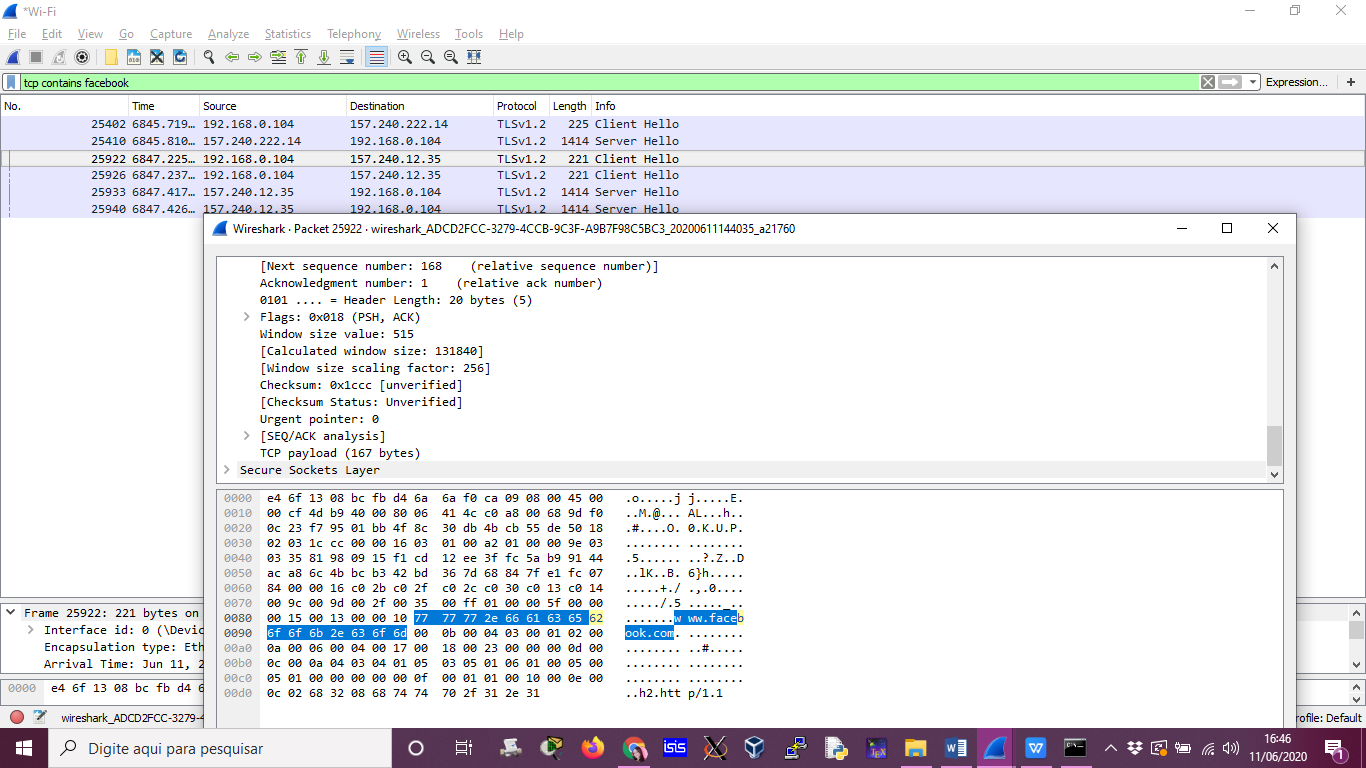
**Figura 6.** Exemplo filtro endereço IP



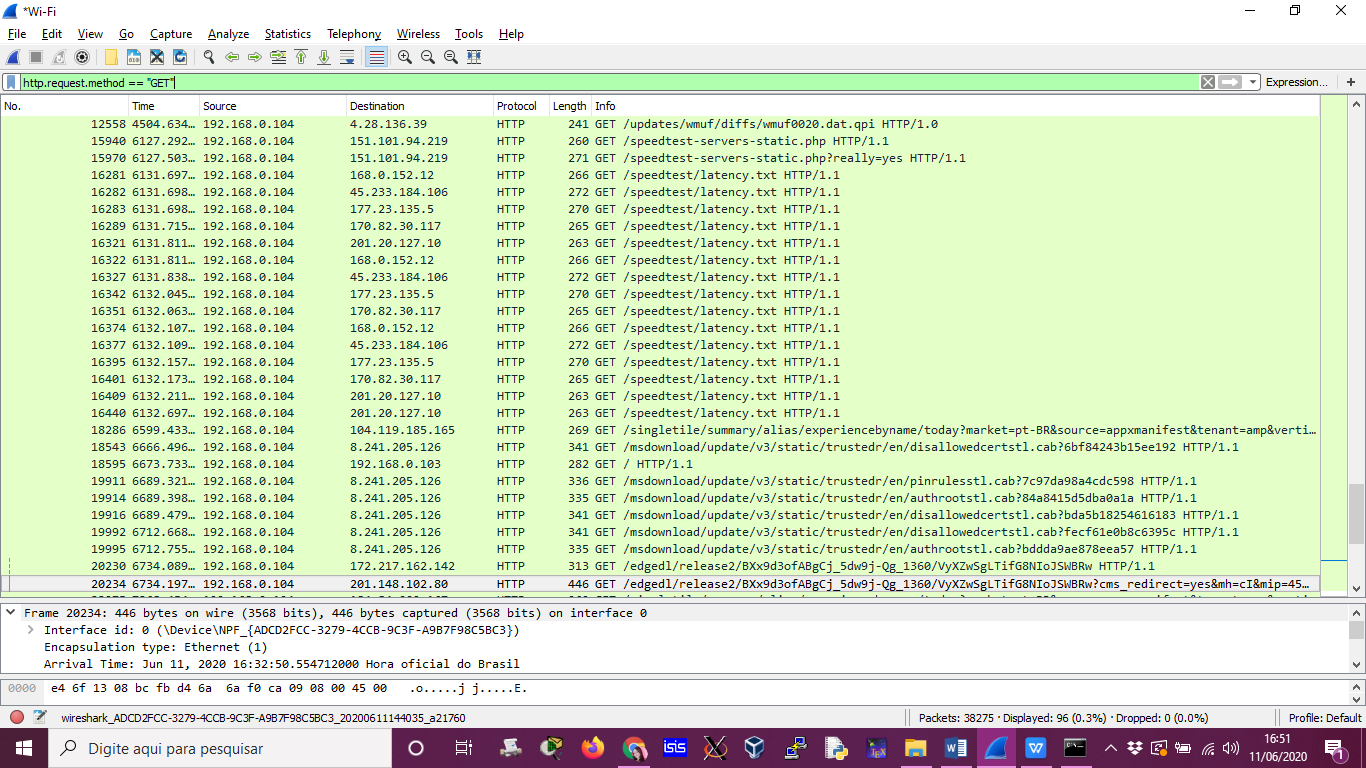
**Figura 7.** Exemplo SYN flood.



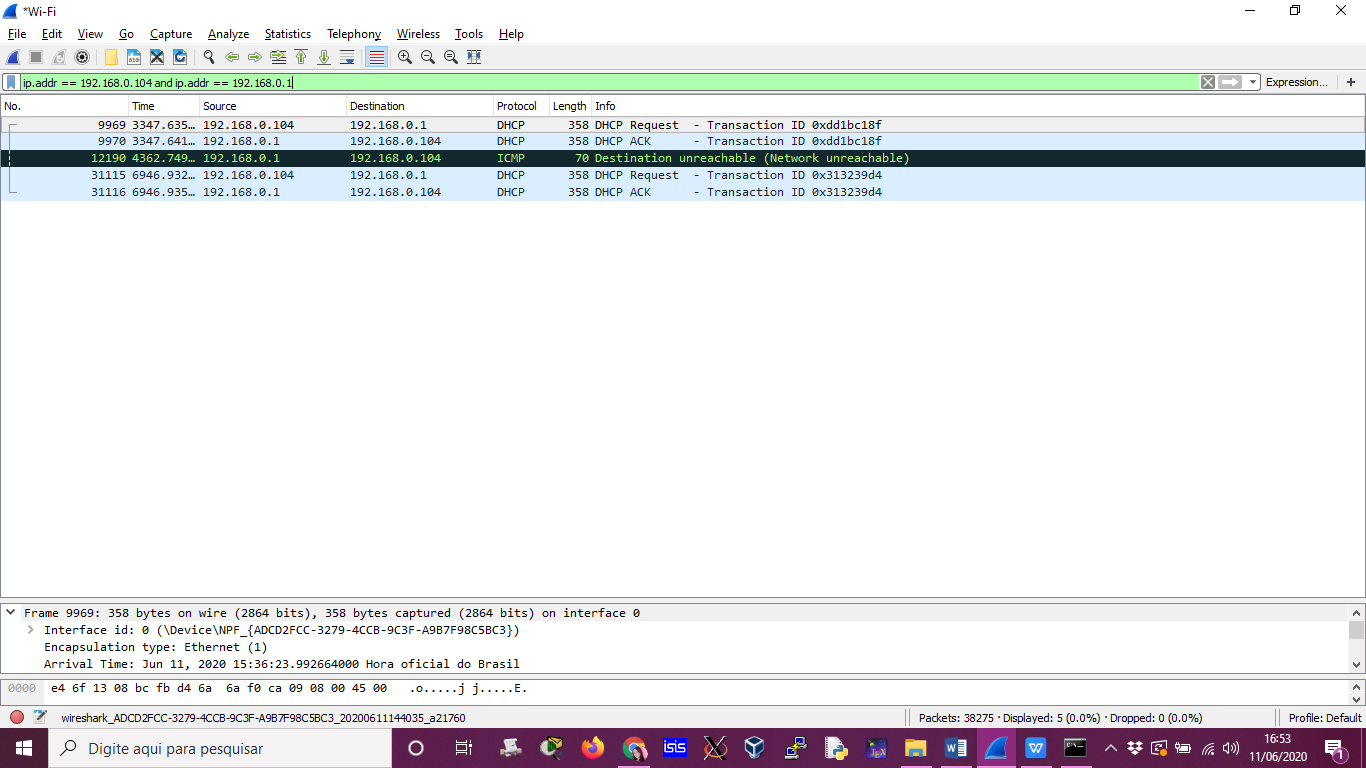
**Figura 8.** Exemplo filtro endereço IP de destino.



**Figura 9.** Exemplo filtro por palavras chaves.



**Figura 10.** Exemplo filtro por requisições GET.



**Figura 11.** Exemplo filtro por tráfego de dados entre IP origem e IP destino.

**Referências**

Bullock, J. e Parker, J. T. **Wireshark Para Profissionais de Segurança: Usando Wireshark e o Metasploit Framework.** 2017. Novatec.

Wireshark User’s Guide. Version 3.3. Disponível em: < <https://www.wireshark.org/download/docs/user-guide.pdf>>. Acesso em 20 mai 2020.

Veiga, M. S. **Wireshark Guia Rápido de Filtros.** Disponível em: < <https://medium.com/canivete-sui%C3%A7o-hacker/wireshark-guia-r%C3%A1pido-de-filtros-49dee2bcf1b>>. Acesso em 20 abr 2020.