Avaliação Progressiva 5

Fernanda Maria de Souza

15 de Julho de 2020

1 Dumps para análise

1.1 TELNET

O primeiro dump analisado foi o TELNET, de modo que o protocolo de transporte utilizado foi o TCP (Transmission Control Protocol).

Primeiramente, a conexão é estabelecida: um pacote SYN é enviado para o servidor informando que o cliente deseja estabelecer uma conexão. Após, o servidor responde o cliente com a flag SYN-ACK, confirmando que recebeu o segmento e está pronto para estabelecer a conexão. Então, o cliente responde o servidor e os dois estabelecem uma conexão.

A partir da conexão estabelecida, a troca de dados é iniciada. O cliente se comunica com o servidor via protocolo TELNET sinalizando que seus dados foram enviados, enquanto o servidor contata o cliente via protocolo TCP confirmando o recebimento dos dados com a flag ACK.

No meio do processo de troca de pacotes há um pacote mal formado, presente na tentativa do servidor de enviar via protocolo TELNET ao cliente um pacote com a flag URG. A bandeira URG ignora o buffering de forma a evitar filas.

Após a troca de pacotes, a comunicação é encerrada por meio de quatro fases via protocolo TCP. Neste caso, o servidor é quem envia um pacote com a flag FIN ativa, obtendo uma resposta ACK do cliente, que por sua vez, irá proceder da mesma forma, enviando um FIN ao qual deverá ser respondido um ACK.

Portanto, as flags utilizadas de modo geral foram:

- SYN (Synchronize Sequence Number): Inicia a conexão.
- ACK (Acknowledgment): Certificação que recebeu o último pacote ou outra resposta.

- PSH (Push): Transmissor notifica ao receptor para que ele passe todos os dados que possui no seu buffer para o processo de aplicação.
- URG (Urgent): É usado para informar que determinados dados dentro de um segmento são urgentes e devem ser priorizados.
- FIN: Termina a conexão.

1.2 SSHv2

Após o dump TELNET, a análise de tráfego no SSH foi feita. O protocolo de transporte utilizado foi o TCP.

SSHv2 é uma facilidade para efetuar login em um computador e emitir comandos remotos como se estivessem sendo executados no computador local. Da mesma forma que o TELNET, por se tratar de uma conexão TCP, a troca de dados somente pode ocorrer após o estabelecimento de uma conexão. É enviado um pacote SYN para o servidor informando que o cliente deseja estabelecer uma conexão. Após, o servidor responde o cliente com a flag SYN-ACK, confirmando que recebeu o segmento. Então, finalmente o cliente responde o servidor com a flag ACK e os dois estabelecem uma conexão.

Neste dump, a versão do SSH do Servidor é a SSH-2.0, enquanto que a do cliente é a SSH-1.99. Após o recebimento das versões, a troca das chaves é iniciada para possibilitar a criptografia da troca de dados pelo método de Diffie-Hellman. O servidor e o cliente recebem uma chave derivada devidamente criptografada e segura.

Após a troca de dados criptografada, o cliente envia o conjunto de flags FIN, PSH e ACK para terminar a conexão, enviando todos os pacotes presentes no buffer. Após, o cliente envia ao servidor a flag RST informando que ocorreu algum erro e a conexão precisa ser resetada. O servidor envia que recebeu a informação enviando a flag ACK, e novamente o cliente envia a flag RST informando que deseja resetar a conexão. O servidor envia três ACK's duplicados, e pela política de congestionamento Reno, resulta no TCP indo para o estado de recuperação rápida. Por fim, a conexão é encerrada com o cliente enviando duas flags RST.

1.3 IPV4

O próximo dump a ser analisado é o IPV4. O mesmo retrata a leitura de e-mail em um cliente da Google. Por meio de conexão via TCP, o cliente realiza o estabelecimento da autenticação com o servidor da Google por meio das mesmas flags informadas no SSHv2 e TELNET (Primeiro o cliente envia a flag SYN, o servidor responde o cliente com a flag SYN-ACK, e o cliente confirma com a flag ACK).

Com a conexão estabelecida, o servidor envia ao cliente via protocolo SMTP uma resposta, com o servidor possuindo ESMTP (extensão do SMTP). O cliente confirma enviando um ACK e após emite a mensagem EHLO, significando que o cliente logou com seu computador e inicou a sessão devido ao servidor suportar ESMTP, com ao final a sessão inicializando corretamente.

Depois, o cliente envia o comando QUIT para o servidor via protocolo SMTP com as flags PSH e ACK. Com isso, a conexão do cliente com o servidor é encerrada. Após, o servidor envia as flag FIN+ACK via TCP para o cliente para fechar a conexão, o cliente envia as mesmas flags e por fim o servidor responde o cliente informando que recebeu as informações, resultando no fechamento da conexão.

1.4 DNS

No dump DNS, o protocolo de transporte utilizado foi o UDP. Como o protocolo não estabelece conexão, o cliente utiliza o DNS padrão principal do Google, que é o 8.8.8.8, para pedir um IP. O IP é retornado com sucesso em seguida.

1.5 HTTP

No dump do HTTP, o protocolo de transporte utilizado foi o TCP.

Primeiramente, por se tratar de uma conexão TCP, o estabelecimento de uma conexão é necessário. É enviado um pacote SYN para o servidor informando que o cliente deseja estabelecer uma conexão. Após, o servidor responde o cliente com a flag SYN-ACK, confirmando que recebeu o segmento. Então, finalmente o cliente responde o servidor com a flag ACK e os dois estabecem uma conexão.

O cliente envia via protocolo HTTP uma requisição GET /download.html HTTP/1.1 solicitando uma representação do recurso especificado, junto com a sua versão do HTTP. O servidor confirma que recebeu a requisição via TCP com a flag ACK e a transmissão de dados é representada para o cliente com a página HTTP. Em dado momento, é feito uma consulta padrão via DNS a um anúncio da página. O anúncio é requisitado para representação do recurso via HTTP com o comando GET. Após os dados do anúncio serem repassados via TCP, o comando HTTP/1.1 200 OK representa que tudo sobre o anúncio foi representado com sucesso.

Depois, o cliente volta para página inicial, recebendo do servidor do anúncio, a execução TCP Spurious Retransmission, significando que o ACK para aquele dado já foi recebido. Após, o cliente envia o recebimento do ACK duplicado via TCP com a flag ACK para o servidor do anúncio e a comunicação é encerrada com aquele servidor. Por fim, o servidor da página principal envia via protocolo HTTP/XML que a solicitação foi bem sucedida.

Após, a conexão é encerrada com as flags FIN+ACK via TCP enviadas pelo servidor ao cliente, o cliente confirma o recebimento via TCP com a flag ACK, e envia a flag FIN+ACK via TCP, finalizando a conexão com o servidor acusando recebimento com a flag ACK via TCP.

1.6 BitTorrent

No dump do BitTorrent, o protocolo de transporte utilizado foi o TCP.

Primeiramente, a conexão é estabelecida entre os peers com o Handshake via protocolo BitTorrent com as flags PSH e ACK, para enviar todas as informações presentes no buffer. Após, é dado Unchoke por um peer com as flags PSH e ACK via protocolo BitTorrent. O outro peer recebe essa informação e após isso a conexão é encerrada. Vários pacotes não são capturados e não é feito o reconhecimento dos mesmos. Após determinado instante que a porta não é reconhecida, o dump é finalizado.

2 Captura de tráfego do Visual Studio Code

Essa análise foi baseada no download do Visual Studio Code (link).

2.1 Fluxo Total

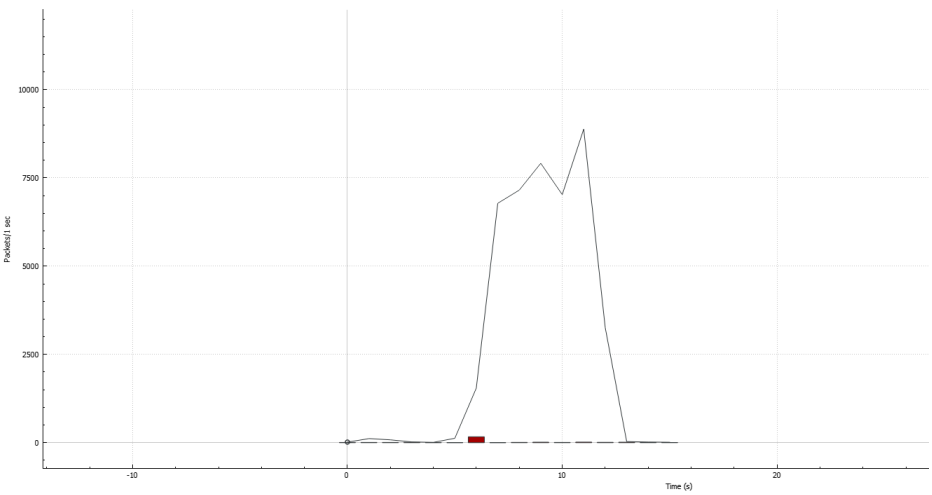

Figura 1: Fluxo Total

2.2 Abertura e fechamento da conexão

O handshaking via protocolo TCP é efetuado nas linhas: 400, 406, 407. Primeiro é enviado a flag SYN do cliente para o servidor, após o servidor responde com SYN+ACK, e por último o cliente responde com ACK.

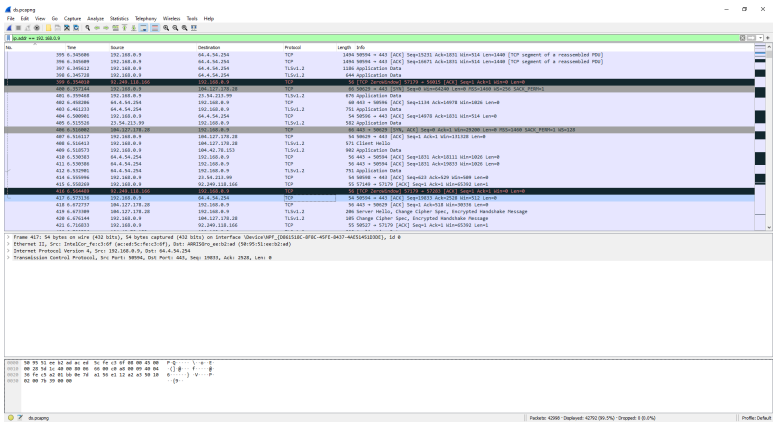

Figura 2: Abertura da conexão.

O fechamento da conexão acontece na linha 42917 com o cliente confirmando por meio da flag ACK via TCP o recebimento da última informação, sendo o último momento que o servidor do VSCode e o cliente se conectam. Após, variadas tentativas por meio de TCP ZeroWindow são observadas, sem sucesso pois o download terminou e a janela tem tamanho 0.

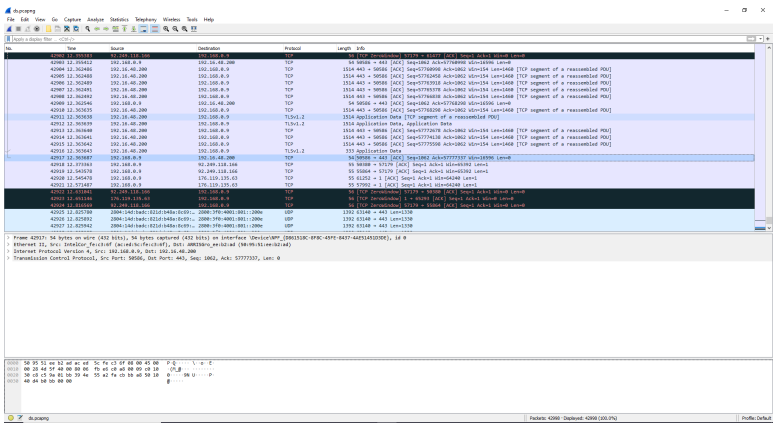

Figura 3: Encerramento da conexão.

2.3 Identificação de eventuais perdas de pacotes, retransmissões e resets (RST)

Nenhum reset foi encontrado.

Um erro de retransmissão encontrado foi devido a determinado pacote ser enviado muito rápido, então, o usuário recebeu uma confirmação de um pacote que não foi recebido. Os 3 próximos pacotes ficaram esperando uma possível confirmação do próximo pacote, porém essa confirmação não tinha sido enviada ainda.

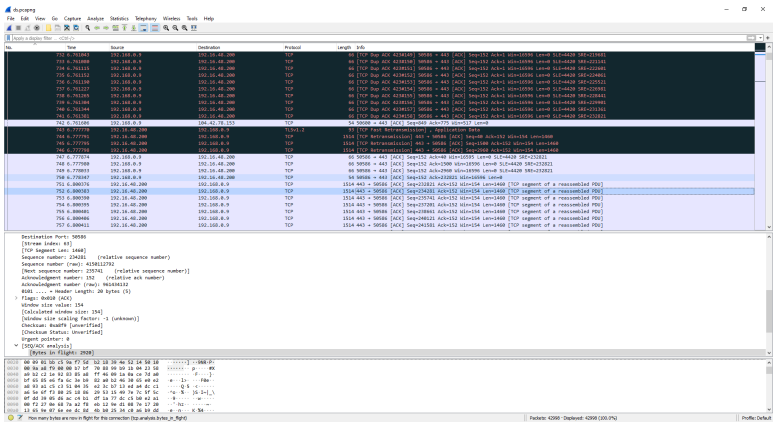

Figura 4: Erro Retransmissão

Na figura 5, podemos ver a ocorrência do erro TCP ZeroWindow. O tamanho da janela repassada é 0, e o mesmo ocorre pela sinalização do servidor ao cliente que o buffer receptor do TCP está cheio e não pode receber mais dados.

Pode acontecer por congestionamentos de processos ou por estar ocupado com alguma outra tarefa. Pode acontecer também por um problema na aplicação, onde o buffer TCP não está sendo recuperado.

Em outro caso como na figura 6, no final do dump se verifica que o mesmo erro acontece várias vezes, ocorrendo pelo fim do download do VSCode, repassando o tamanho da janela como 0, pois o buffer do cliente é inexistente quando não há conexão.

No começo do download foi observado erros relacionados a ACK duplicados. Como na figura 7, isso acontece normalmente por perda de pacotes, processo normal na transferência de dados.

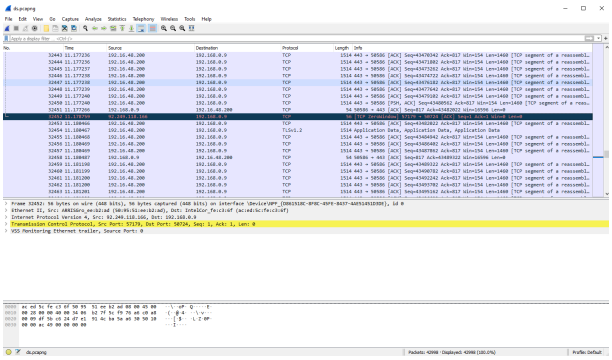

Figura 5: Erro ZeroWindow

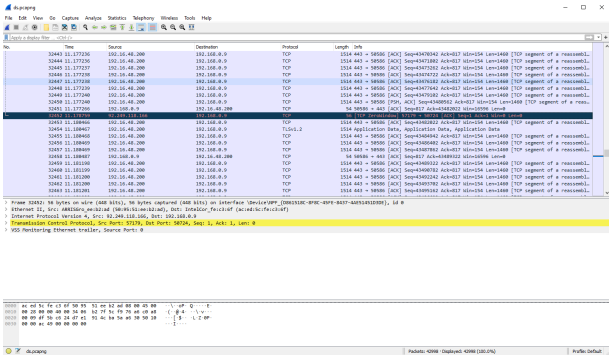

Figura 6: Erro ZeroWindow-2

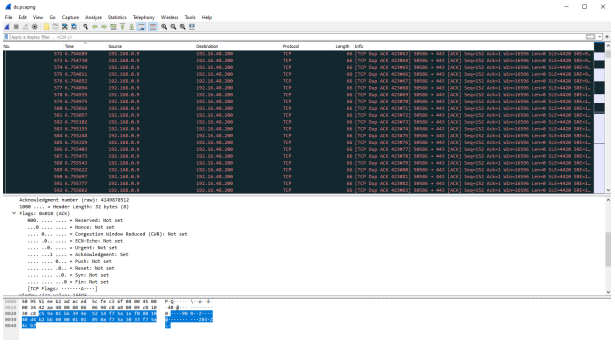

Figura 7: Erro Dup ACK

2.4 Tamanho máximo das janelas (rwnd e cwnd) durante a conexão

O tamanho máximo da janela rwnd é 16596 bytes. O valor é obtido a partir do valor da janela no pico de pacotes enviados.

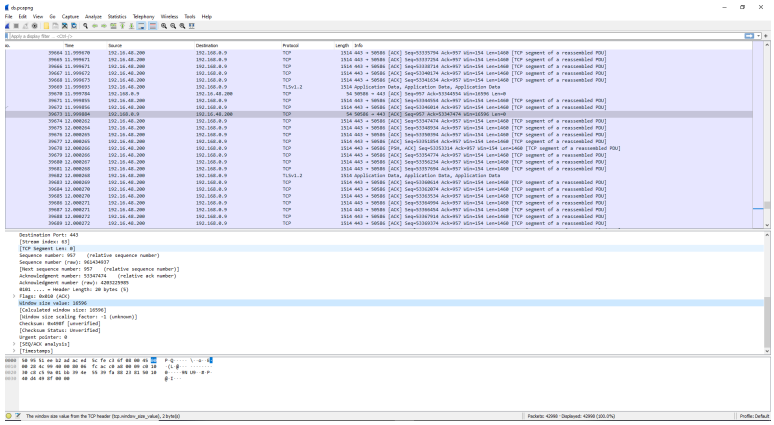

Figura 8: Tamanho máximo da janela rwnd

O tamanho máximo da janela cwnd (congestionamento) atingido na conexão atual, analisando o ponto mais alto do gráfico, é de 8884 bytes.

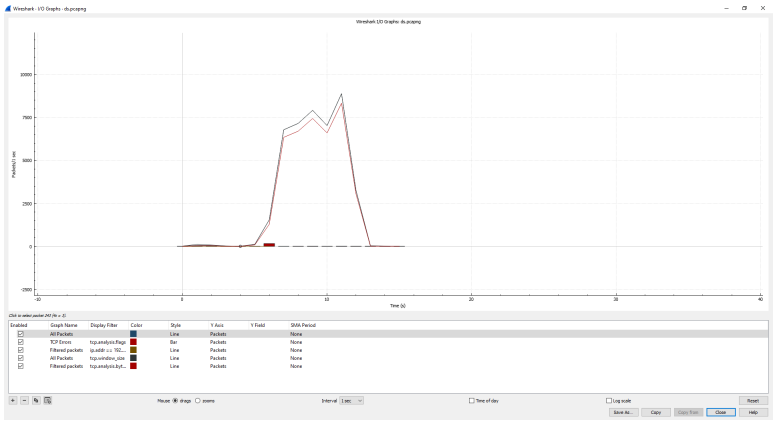

Figura 9: Tamanho máximo da janela cwnd

2.5 Análise da relação entre número de sequência e tempo usando time-sequence graph (stevens e tcptrace)

Pacotes de dados que dão erro não crescem no tempo, pois são descartados. Os pacotes que completam a requisição e transferência na rede com sucesso crescem com o tempo de acordo com seu número de sequência lógica. A contagem da sequência irá iniciar no Handshake. Para o tcptrace, o número da sequência cresce 1 byte a cada dado via TCP enviado.

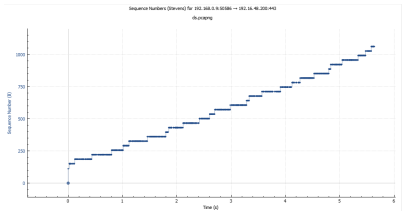

Figura 10: Stevens

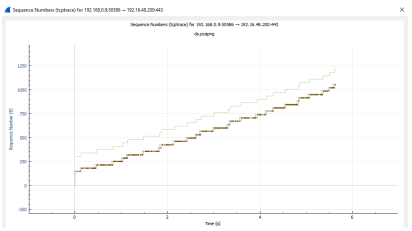

Figura 11: Tcptrace