UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMATICA E ESTATISTICA
DISCIPLINA: Redes De Computadores I
PROFESSOR: Carlos Becker Westphall
ACADÊMICO: Bruno Aurélio Rôzza de Moura Campos
MATRÍCULA: 14104255

TRABALHO PRÁTICO 03

Florianópolis, maio de 2016

1.Introdução

Proposta do professor:

Neste trabalho prático os alunos deverão utilizar o "wireshark" para identificar a ocorrência de conexões e transferências de dados, envolvendo as camadas de aplicação, transporte e rede. Nestas telas capturadas pelo "wireshark" os alunos deverão identificar os "pacotes" relacionados com criação da conexão, a transferênica de dados e a liberação de conexões. Usar, preferencialmente, FTP, HTTP e/ou SNMP. Fazer o relatório descrevendo as atividades. O arquivo de dados analisado e obtido atráves do "wireshark" também deve ser entregue junto com o relatório.

Esse relatório tem como objetivo analisar o tráfego de uma rede para identificar a criação da conexão, a transferência de dados e a liberação de conexões, com base em um programa snifffer (wireshark). Este tipo de ferramenta registra os pacotes, que trafegam pelas redes, e suas informações. Como são muitos dados, o programa possibilita a filtragem por tipo de protocolos. Para uma maior facilidade foi filtrado os protocolos de interesse HTTP e TCP.

1.1 Dados utilizados

Para a realização deste trabalho, foi escolhido para analisar a URL https://moodle.ufsc.br (150.162.1.112). No arquivo de captura de pacotes foi obtido 545 pacotes sendo o navegador utilizado Mozila Firefox.

2. Criação da conexão

A primeira atitude para se obter a comunicação é denominado processo cliente. O processo de aplicação cliente primeiramente informa à camada de transporte do cliente que ele quer estabelecer uma conexão com um processo servidor. Para isso é realizado a apresentação de 3 vias (**3 way handshake**).

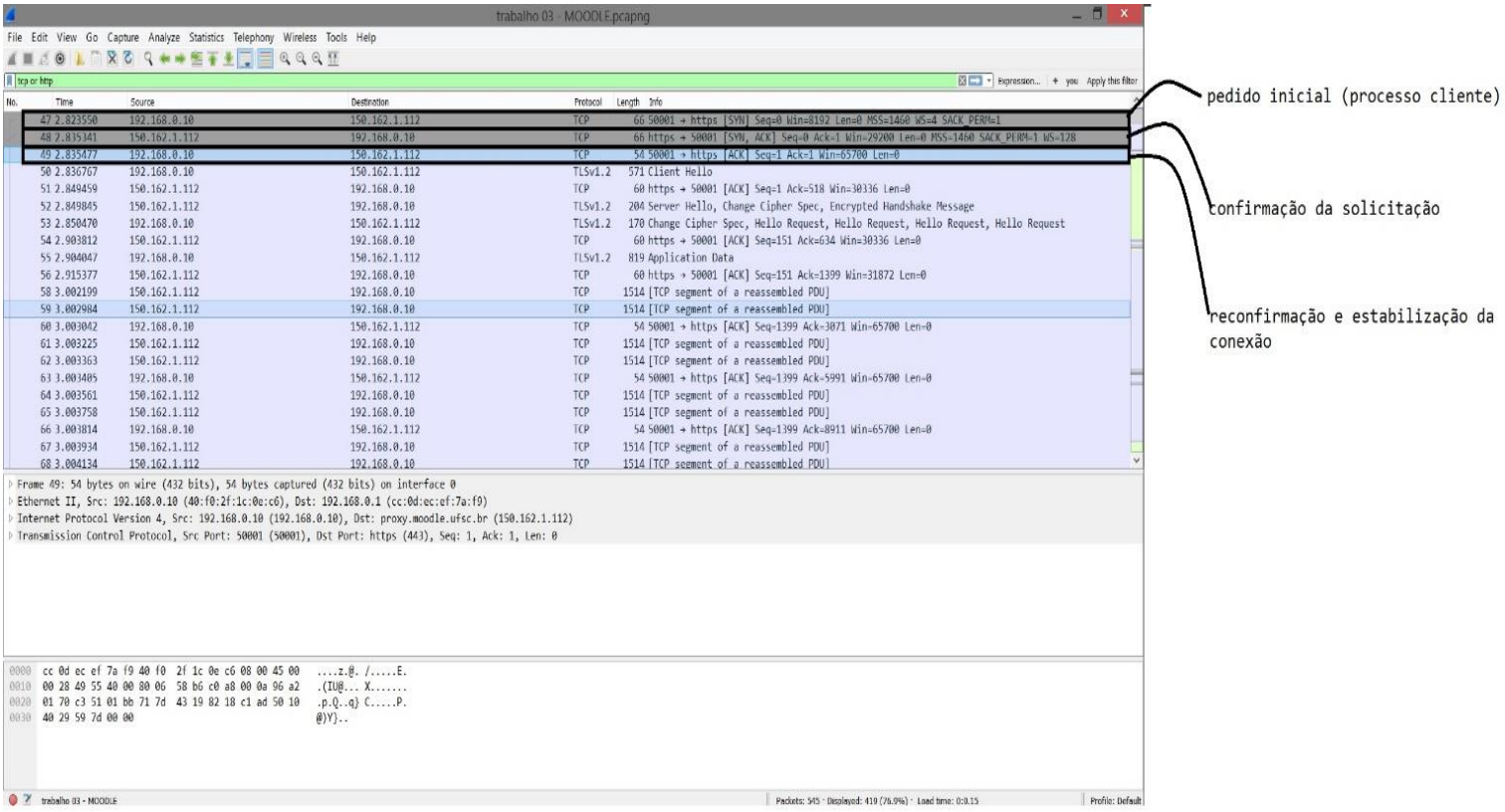

Figura 01. 3 way handshake

Etapa 01:

Primeiro é feito o pedido inicial com a flag SYN, com um valor aleatório.

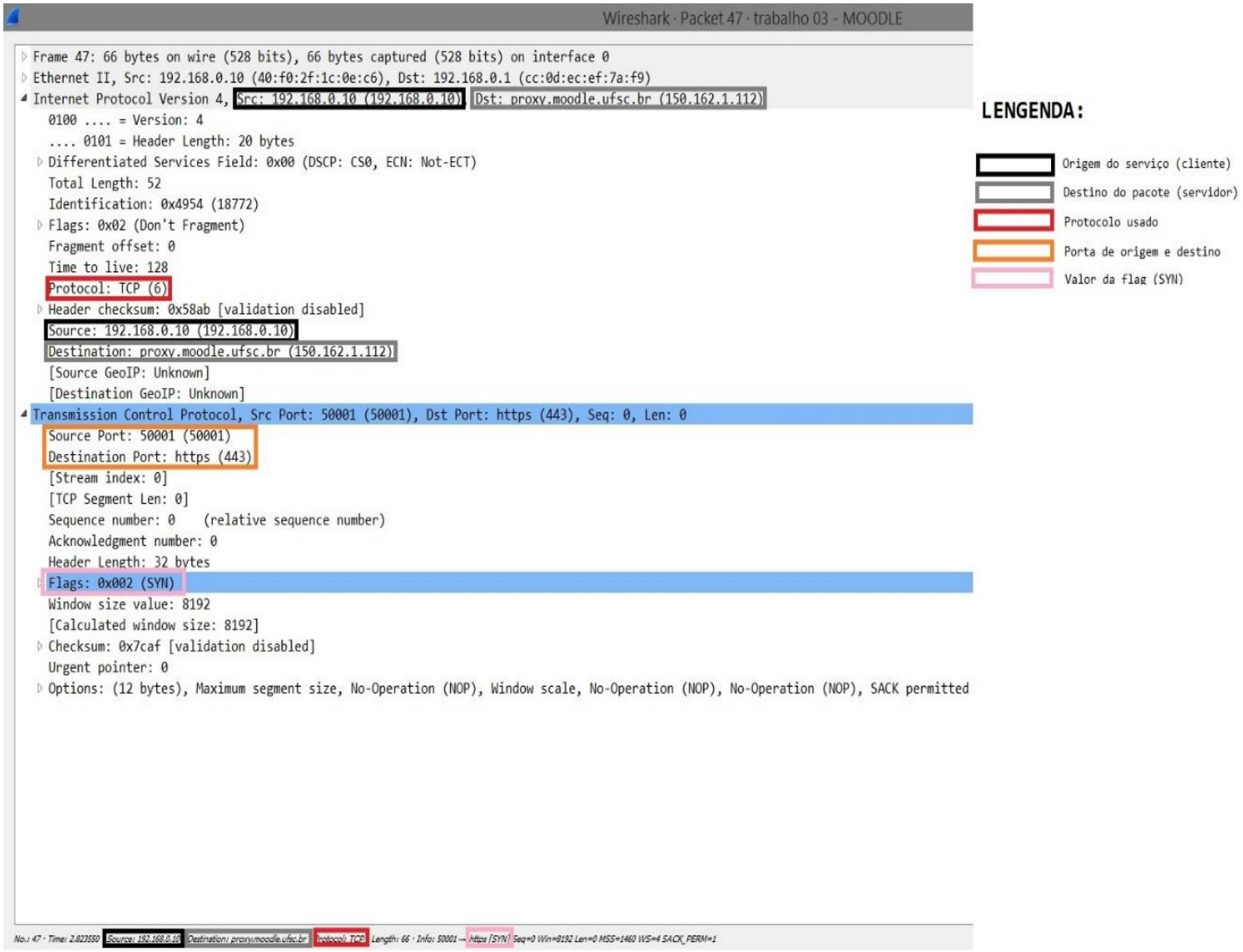

Figura 02. Início do processo de comunicação com a flag SYN

Etapa 02:

Se disponível o servidor responde com uma confirmação do pedido em um novo pacote contendo as flags SYN + ACK.

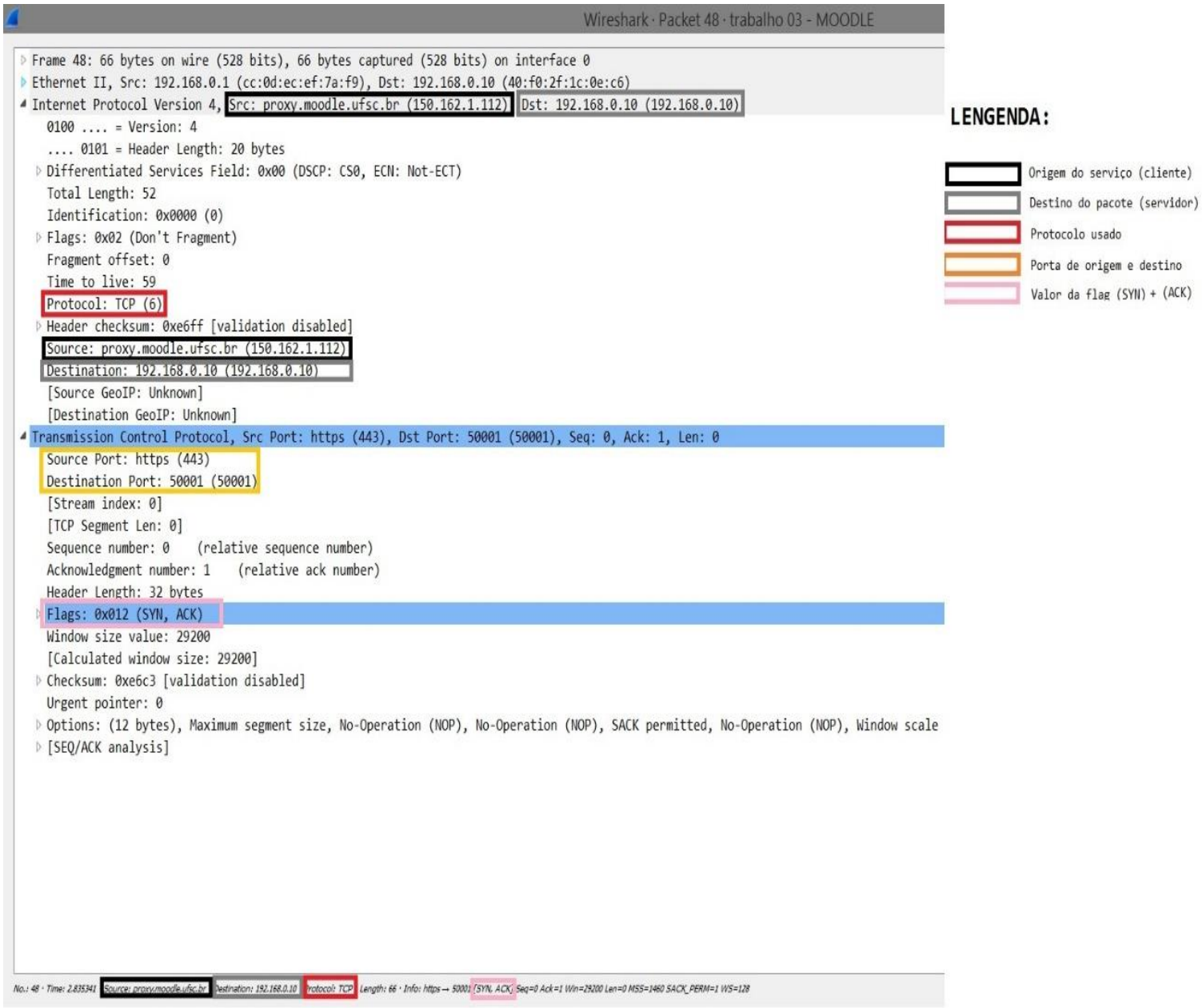

Figura 03. Resposta do servidor com as flags SYN + ACK

Etapa 03:

Para estabelecimento do processo cliente-servidor o cliente envia mais um pacote contendo um ACK do mesmo número de ACK recebido, que veio do servidor.

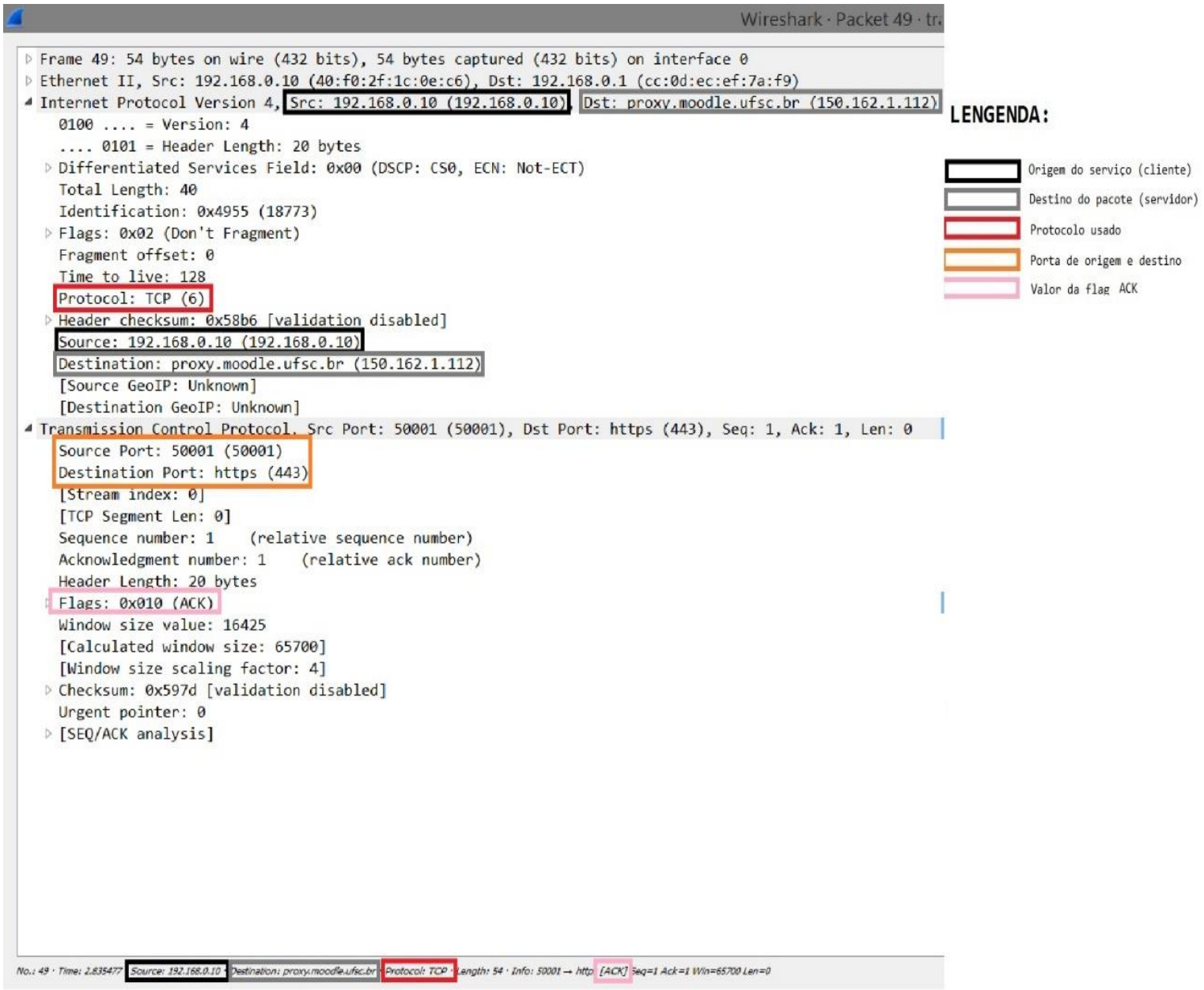

Figura 04. Estabelecimento de comunicação com a flag ACK

3. Transferência de Dados

Após o cliente ter executa uma primitiva CONNECT, especificando o endereço IP e a porta à qual deseja se conectar, A primitiva CONNECT envia um segmento TCP com o bit da flag SYN ativado e um bit da flag ACK desativado, e aguarda uma resposta. Para um maior entendimento sobre as flags, abaixo há um exemplo de como ela é configurada num pacote.

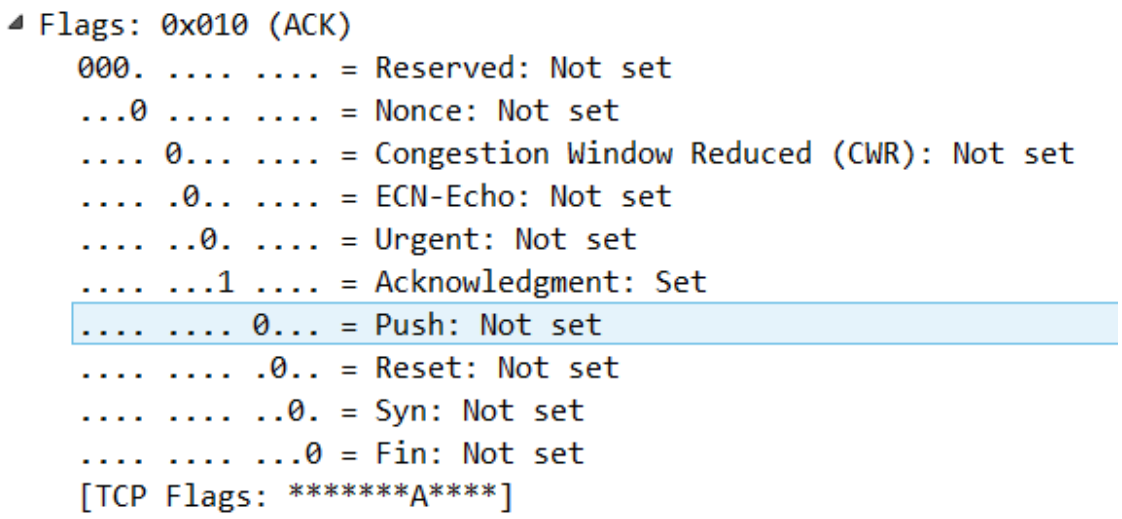

Figura 05. Exemplo de ACK configurado dentro do pacote

Os programas que trabalham com redes estão fortemente ligados na camada TCP/IP, de aplicação. Com a requisição processada, essa camada chama a de Transporte através de uma porta específica, dependendo do protocolo usado, como HTTP, que sempre usa a 80. Assim já indica qual tipo do conteúdo e, no receptor, possibilita entregar os dados para o protocolo de aplicação correto.

A camada de Transporte é responsável no envio por dividir os dados em pacotes; e a receber por ordenar os pacotes. Sendo usado principalmente o TCP. O protocolo nomeado como Transmission Control Protocol encapsula nos pacotes informações de controle, resultando entre 20 a 24 bytes.

Na transferência de dados com o TCP, ao se enviar uma solicitação, ou seja, iniciar um processo cliente, obrigatoriamente haverá uma resposta com os seguintes campos:

Protocolo / versão do protocolo + código de requisição + significado do código + rubricas de resposta + corpo de resposta

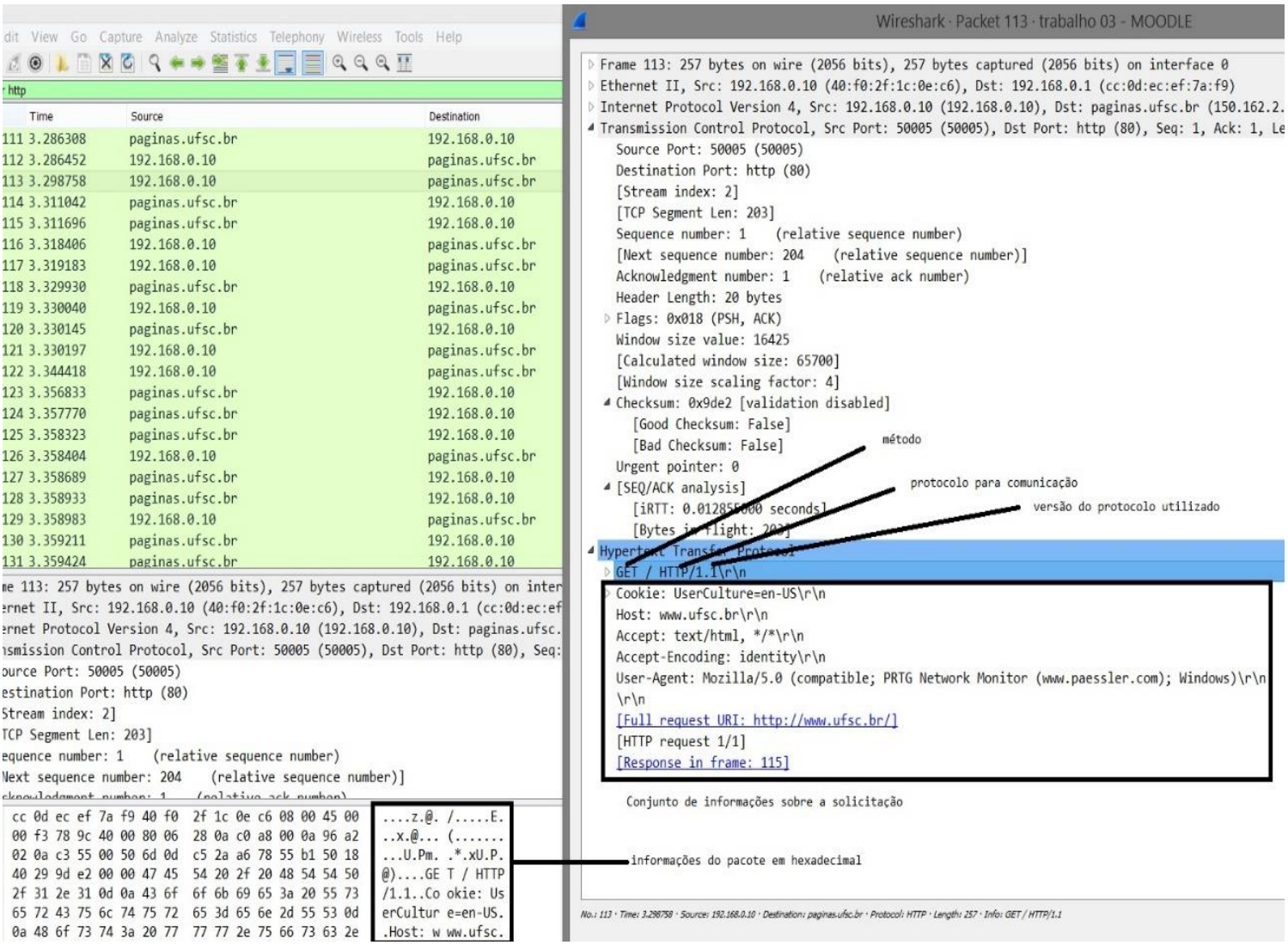

Figura 06. Pacote HTTP com o método GET

No pacote acima há algumas informações que identificam o cliente no caminho mas o mais importante é o método da requisição. O servidor, por sua vez, identifica os cabeçalhos que lhe são convenientes e envia uma resposta. Neste exemplo, recebemos os cabeçalhos de resposta.

Método GET: É o método mais comum para o protocolo HTTP: solicita algum recurso como um arquivo ou um script CGI (qualquer dado que estiver identificado pelo URI) por meio do protocolo HTTP. O método GET é reconhecido

por todos os servidores. Quando uma solicitação obtém sucesso o código do HTTP é 200 caso contrário, se não existir, é 404.

4. Liberação de conexões

Para encerrar uma conexão, qualquer um dos lados pode enviar um segmento com o bit FIN o que significa que não há mais dados a ser transmitidos. Quando FIN é confirmado, esse sentido é desativado para novos dados.

De modo geral, são necessários **quatro segmentos TCP para encerrar uma conexão**, isto é, um FIN e um ACK para cada sentido.

Quando um dos extremos ter a iniciativa de finalizar a conexão, será enviado desse um pacote TCP com a flag FIN. Contando que não ocorra erros, o outro irá confirmar o pedido com o ACK e seguidamente irá enviar um FIN; sendo realmente interrompida a ligação quando o primeiro enviar um ACK.

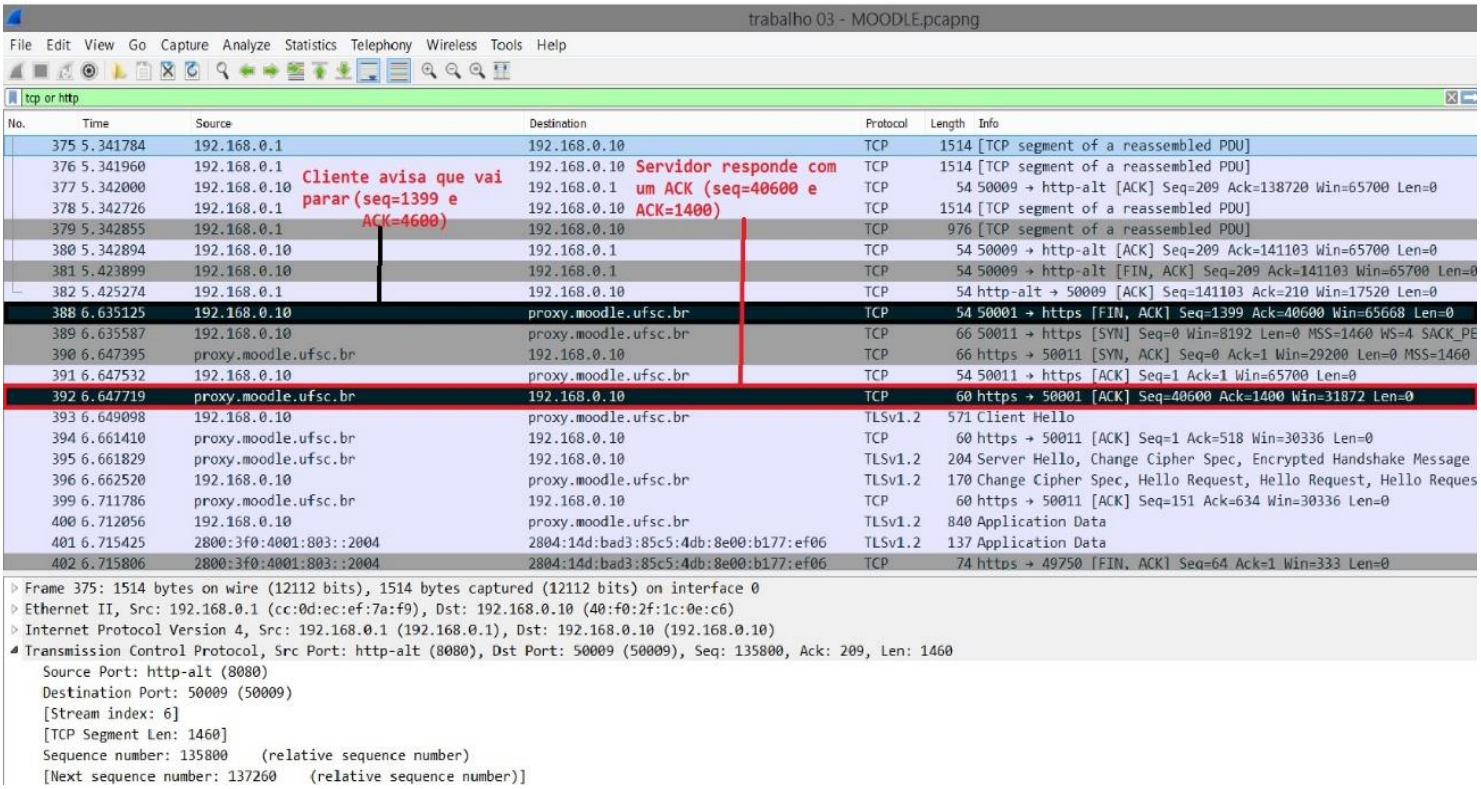

Figura 07. Dois primeiros pacotes para o encerramento da comunicação

Para maiores detalhes do encerramento de comunicação, veremos o pacote que contem a flag FIN.

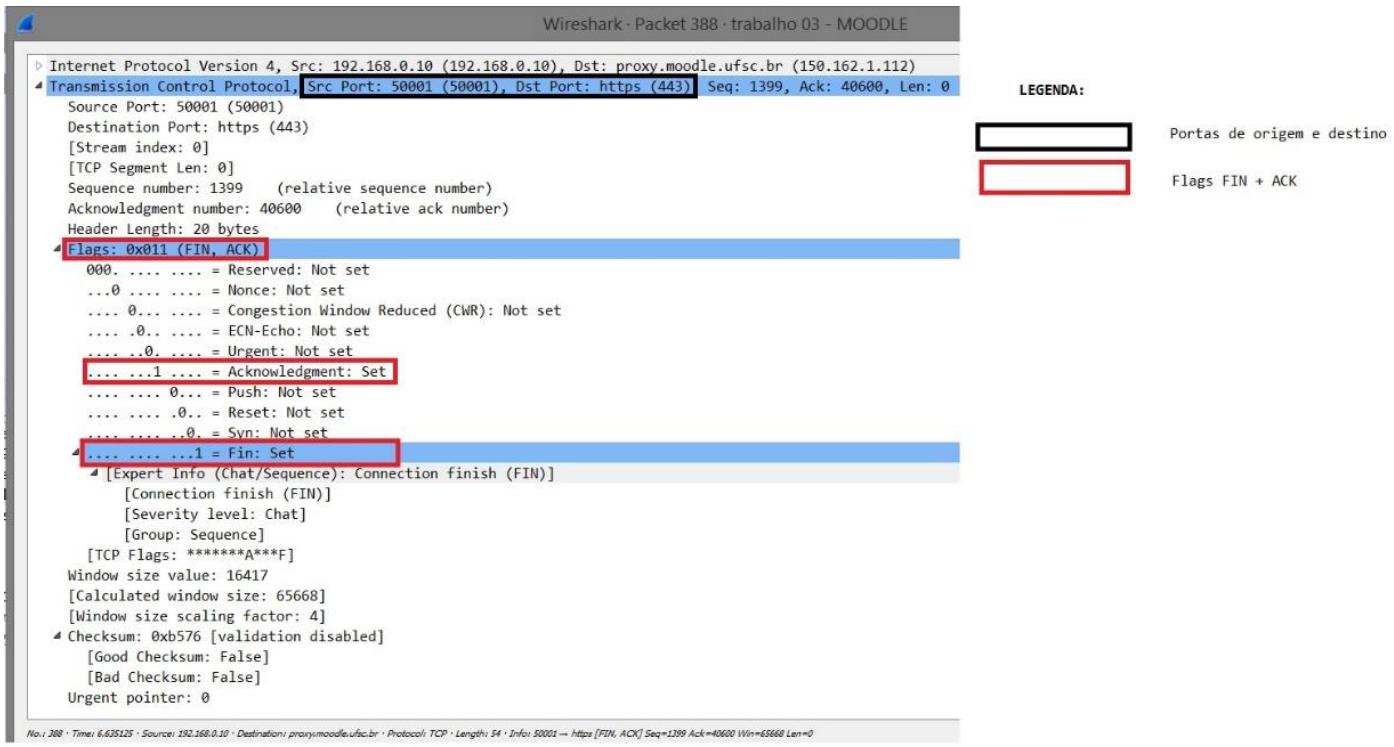

Figura 08. Portas de origem e destino são as mesmas utilizadas no 3 way handshake

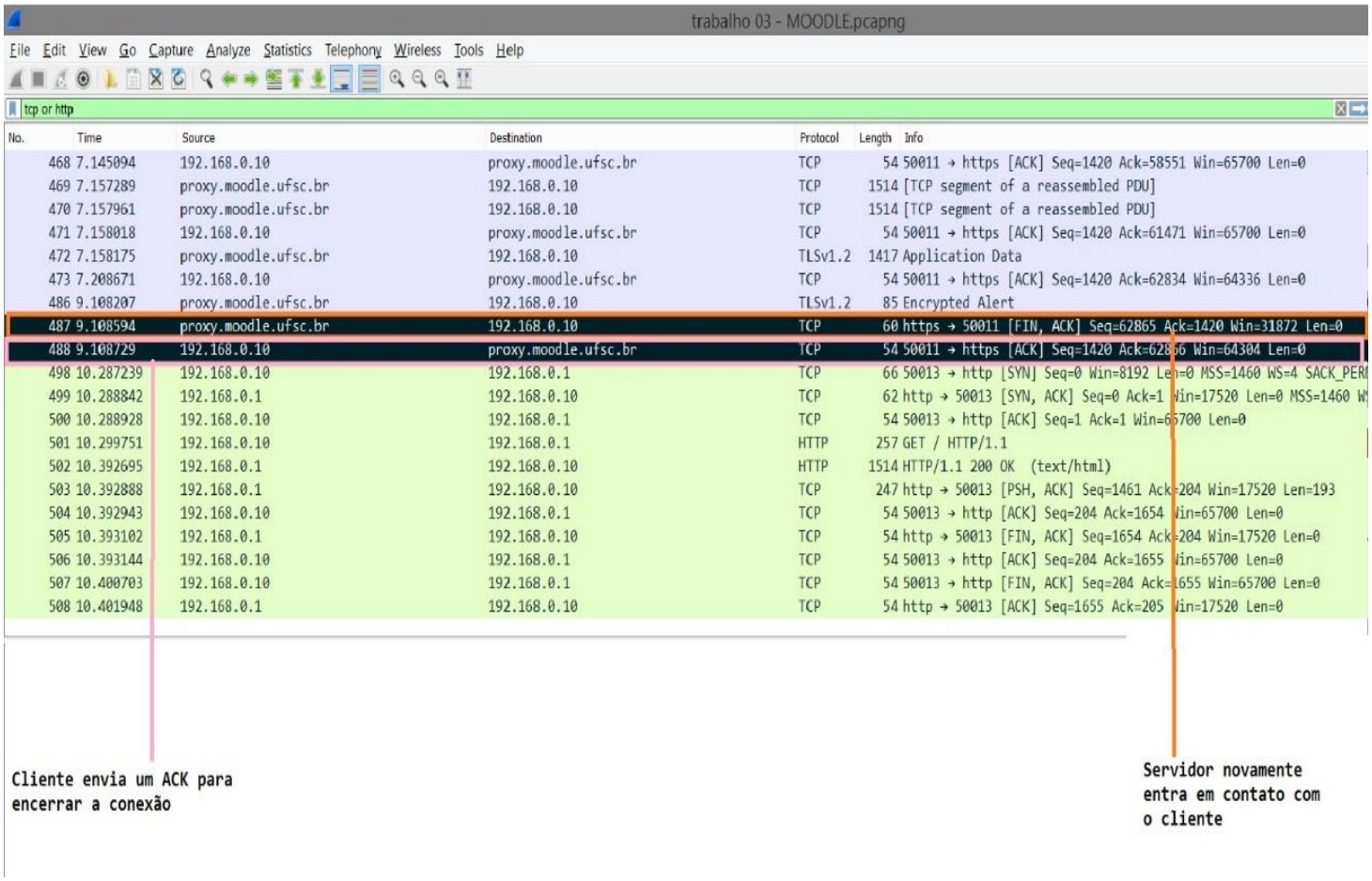

Figura 09. Dois últimos pacotes para o encerramento da comunicação

5. Bibliografia

[1]https://www.oficinadanet.com.br/artigo/459/o_protocolo_http

[2]https://nandovieira.com.br/entendendo-um-pouco-mais-sobre-o-protocolo-http

[3] https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

[4] Kurose, Ross. (2013). *Redes de Computadores e a Internet,* Local: Pearson