Relatório de Teleinformática e Redes 2

Projeto Final de Disciplina – 2018/2

Identificação

Emily Souza Rodrigues – 15/0009101
Ricardo Arôso Mendes Kury – 14/0161082

Teoria

Proxy Server Web

Proxy é um servidor que atua com intermediário entre um cliente que solicita recursos de outros servidores. No caso de um *proxy server web*, os recursos solicitados pelo cliente são conteúdo da *World Wide Web* (WWW), como páginas webs. Cliente nada mais é que um *host* – máquina conectada à rede – que inicia uma conexão de rede com outro *host* – Servidor.

Nesse trabalho, o cliente – browser – ao fazer um pedido de conteúdo – utilizando o HTTP – para um determinado servidor, se conecta ao nosso *proxy* – utilizando o TCP – e tem esse pedido intermediado por ele. Nosso *proxy* permite ao usuário realizar alterações no pedido antes do envio para o servidor, além de alterar algumas informações de cabeçalho do protocolo. Só após o aval do usuário uma conexão TCP não persistente é estabelecida com o servidor – conexão fechada após o recebimento de resposta – e nessa conexão é enviada a requisição do HTTP. O servidor, então, retorna uma resposta que é intermediada pelo nosso *proxy* – usuário pode alterar a resposta do servidor antes de retorná-la ao cliente –, a conexão é fechada e, após o aval do usuário, a resposta é transmitida para o cliente – conexão com cliente é persistente, ou seja, não é fechada após a resposta ser recebida pelo cliente.

Transmission Control Protocol (TCP)

Transmission Control Protocol é um protocolo de transferência confiável de dados da camada de transporte. Ele implementa um conjunto de recursos que garantem, além da transferência confiável de dados, entrega ordenada, controle de fluxo e controle de congestionamento. É um protocolo *full-duplex* – conexão funciona para envio e recebimento – orientado à conexão com *3 way handshake* – mecanismo que garante o estabelecimento da conexão antes do início da transmissão de dados.

Hypertext Transfer Protocol (HTTP)

Hypertext Transfer Protocol é um protocolo da camada de aplicação que funciona sobre o TCP, já que a confiabilidade na entrega, bem como a entrega ordenada aspectos são importantes para esse protocolo.

Esse protocolo permite a comunicação entre a aplicação cliente e o servidor utilizando de caracteres ASCII. Ele pode ser usado para realizar uma requisição ou para responder a uma requisição. Sobre o formato da requisição, o protocolo define oito métodos (GET, PUT, POST, HEAD, DELETE, TRACE, OPTIONS, CONNECT) e informações de cabeçalho como persistência da conexão TCP, idioma

desejado para a resposta e aplicação que gerou a requisição no cliente. Alguns dos métodos necessitam transferir arquivos ou demais informações, para isso existe o campo opcional de corpo da requisição. Por outro lado, a resposta costuma seguir a estrutura: versão do protocolo, mensagem de retorno – com seu devido código –, cabeçalho similar ao da requisição e corpo da mensagem.

Spider – Árvore hipertextual

Essa funcionalidade do nosso *proxy* consiste em, a partir de um URL (*Uniform Resource Locator*) inicial, listar todos os demais URLs – dentro do mesmo domínio – que contam com referência no URL inicial ou dos URLs que o inicial referencia. O caminho para chegar a um URL específico a partir do inicial é preservado na hierarquia da árvore hipertextual.

Cliente Recursivo

De posso da árvore hipertextual, o cliente recursivo copia do servidor todos os arquivos listados na árvore e corrige as referências internas dos arquivos para no lugar de referenciar um lugar na rede, referenciar um diretório da própria máquina onde se encontra uma cópia do arquivo sendo referenciado.

Arquitetura do sistema

BrowserSocket

Módulo responsável por criar, manter ativa e utilizar a conexão TCP com o navegador.

Ao iniciar o módulo, ele fica escutando a porta 8228 – ou outra porta informada pelo usuário – esperando as requisições HTTP do navegador. Quando uma requisição é recebida, um objeto HttpRequest é criado e passado para o programa.

Ao receber um objeto HttpRequest com a resposta da requisição, a resposta HTTP é enviada ao navegador pela conexão TCP mantida.

InternetSocket

Módulo responsável por criar e utilizar a conexão TCP não persistente com o servidor.

Ao receber um objeto HttpRequest, o módulo busca no DNS o IP do hospedeiro informado e tenta criar uma conexão com esse IP. Se bem-sucedida, o objeto é transformado em uma requisição e enviado ao servidor.

Quando a resposta do servidor chega – muitas vezes divididas em pacotes diferentes – o módulo junta corretamente a resposta e gera um objeto HttpRequest com essas informações, objeto esse que será passado para o programa.

Spider

Módulo responsável por gerar a árvore hipertextual de referências.

O módulo necessita de uma URL para funcionar, de posse dela, cria um objeto InternetSocket e HttpRequest. O objeto de requisição é passado para o *socket* que envia para o servidor a requisição. Ao retornar a resposta, o Spider separa o cabeçalho e corpo da resposta e analisa o corpo buscando por todas as referências presentes nele – busca por href – ignorando as

referências para fora do domínio da URL inicial e quando a referência é para dentro do domínio, repetindo as etapas de criação do *socket*, geração e envio de requisição, separação de corpo da reposta e busca por referências recursivamente.

HttpRequest

Módulo responsável por converter a requisição HTTP de texto ASCII para uma estrutura manipulável pelo programa e vice-versa. Ele realiza a análise dos campos do cabeçalho para extração das informações nele contidas – o que permite filtragens de requisições futuras.

Capturas de Tela

Proxy

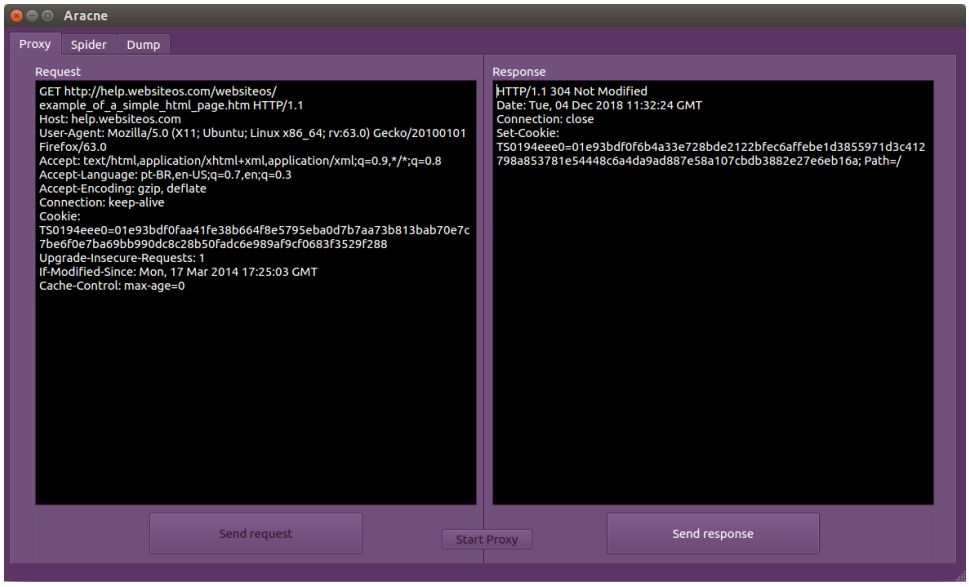

Figura 1: Requisição e resposta sem alteração

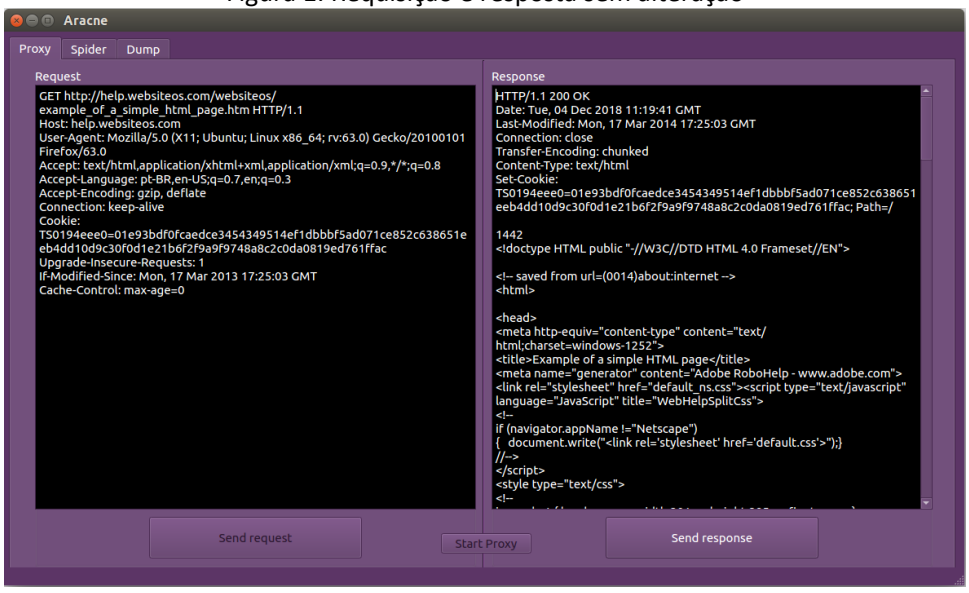

Figura 2: Requisição com “If-Modified-Since" alterado

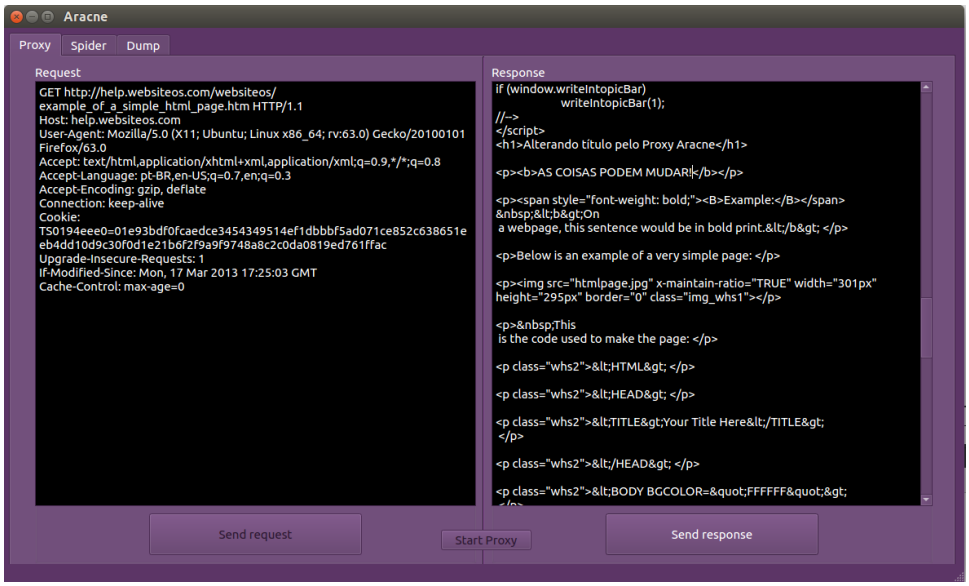

Figura 3: Resposta alterada antes de retornar ao cliente

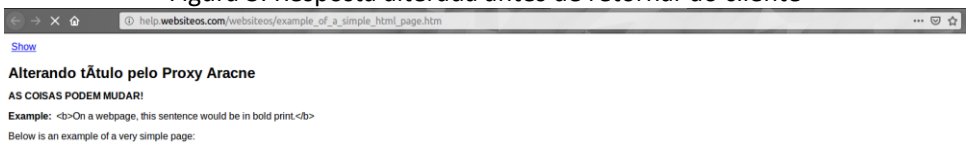

Figura 4: Resultado no cliente da alteração da resposta do servidor

Spider

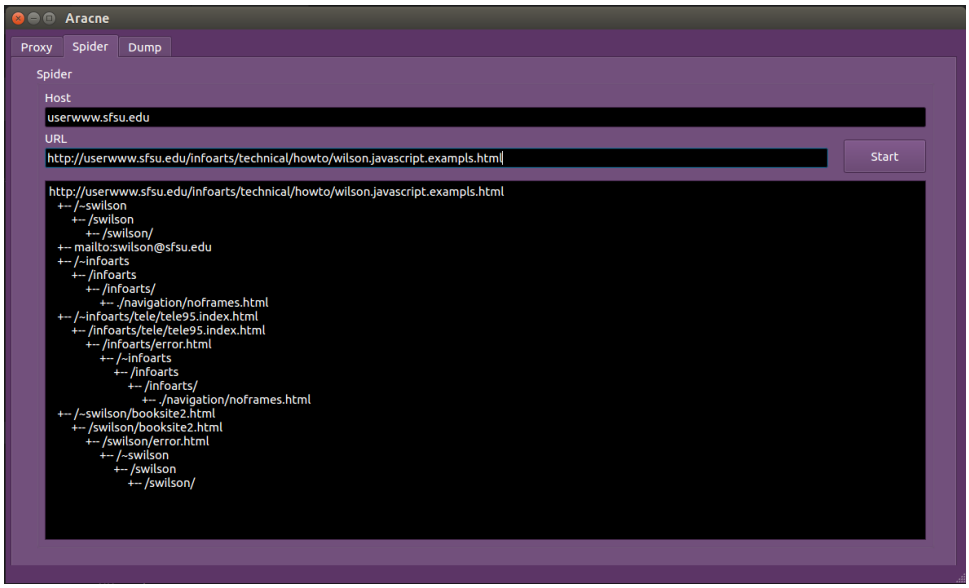

Figura 5: Árvore hipertextual gerada pelo Spider

Dump

(Não implementado)