Um Inspetor HTTP baseado em Proxy Server

Hélio Santana da Silva Júnior - 14/0142959
Jonatas Ribeiro Senna Pires - 14/0090983

Dezembro 2018

1 Apresentação Teórica

1.1 Proxy Server Web

Um servidor proxy web atua como um nó intermediário nas requisições HTTP de clientes. O cliente se conecta ao servidor proxy web fazendo uma requisição, normalmente de um objeto, de um servidor diferente na web. O servidor proxy avalia essa requisição, caso o objeto esteja armazenado em cache, este objeto é repassado ao cliente, caso não esteja armazenado, o servidor repassa essa requisição a um servidor na web e assim retorna a requisição ao cliente. Outras funcionalidades do servidor proxy web são: monitoramento e filtragem, caching e controle de acesso. Neste trabalho o servidor proxy web só irá realizar monitoramento e controle de tráfego editando as requisições de envio e recibo.

1.2 TCP - Transmission Control Protocol

A aplicação, no emissor, envia mensagens atráves de um socket, que por sua vez, no lado do receptor a mensagem é recebida a partir da camada de transporte. Deste modo, o HTTP escolhe o TCP como protocolo de transmissão pela sua caracteristica de entrega confiável de dados e mantimento da ordem das mensagens. O TCP também executa um controle de congestionamento que irá limitar a conexão de a cordo com a banda da rede. O TCP é um protocolo orientado a conexão, pois, antes de cada conexão cliente e servidor trocam mensagens, fazendo um "aperto de mão de 3 vias", para estabelecer a conexão.

1.3 HTTP - HyperText Transfer Protocol

O HTTP define a estrutura das mensagens trocadas na Web. As mensagens, ou páginas web, possuem objetos, que podem ser imagens, arquivos html ou até mesmo arquivos de áudio e vídeo. Cada mensagem é endereçada por uma URL, que é dividida pelo nome do Host do servidor e o nome do caminho do objeto. Os navegadores, como firefox e chrome, por exemplo, são responsáveis por implementar o cliente HTTP. O processo começa com uma conexão TCP com o servidor pela porta 80, em seguida uma requisição é realizada. Existem

alguns métodos podem ser realizados na conexão, como por exemplo: GET, POST, HEAD, PUT e DELETE. A URL e a versão do HTTP estão na primeira linha depois das linhas do cabeçalho da mensagem. Na resposta existe a linha de status que possui a versão do HTTP, o código e a frase de status, seguida pelas linhas do cabeçalho, linha em branco e o corpo da entidade. O servidor não mantém as informações de clientes antigos que realizaram requisições, deste modo, a cada requisição de mesmo cliente, todos os objetos requisitados são enviados novamente.

1.4 Spider e Recursive Client

O spider é um rastreador de páginas que identifica todos os links de uma página, realizando uma árvore hipertextual de todas as URL's dentro do dominio da página raiz. O cliente recursivo tem como objetivo realizar um dump da página acessada, salvando uma cópia de todos os objetos acessados pela árvore gerada, mantendo uma estrutuda de pastas de um servidor remoto.

2 Arquitetura

O sistema está distribuido entre arquivos *.cpp* e *.hpp* de modo a organizar as funções e declarações de variaveis globais. Foram criadas seis classes para a resolução dos problemas propostos, as classes são:

- HTML Parser
- HTTP Request
- HTTP Response
- Proxy Server
- Spider
- String Functions

A função *main* é responsável por receber a porta fornecida pelo usuário, chamar a função de interface gráfica e executar o sistema. As sessões a seguir irão se aprofundar no funcionamento de cada classe.

2.1 HTML Parser

A classe de *parser* é responsável por extrair informações do corpo da mensagem, possuindo quatro funções:

- getUrl
- getSource
- getImport
- getHtml

2.2 HTTP Request

A classe *HTTP Request* é responsável por tratar os *requests* inspecionados a patir do servidor *proxy*. Essa classe possui cinco atributos: método, url, versão, campos e corpo. Esses atributos juntos formam a mensagem de *request*. Os métodos dessa classe são:

- print
- tratarConexao
- avaliaMetodo
- montaRequest

O método *print* é resposável por mostrar na tela os atributos da classe. O método de avaliação testa se o método do *request* é GET ou não, sua resposta é uma variável boleana. O método de tratamento é responsável por colocar no campo *connection* da mensagem, o valor *close* e no campo *accept encoding* o valor *identity*. Por fim, o útlimo método é responsável por montar o request.

2.3 HTTP Response

A classe de *response* é semelhante ao *request* no sentido de montar a mensagem, porém neste caso a mensagem de response. A classe possui quatro atributos: codigoStatus, versao, campos e dados. Seus métodos são:

- print
- montaResponse

A função de *print* semelhante a classe anteriror, mostra os atributos para o usuário. A função de montar o *response* monta uma lista com os atributos para criar a mensagem de *response*.

2.4 Proxy Server

Essa classe é responsável por começar a execução do sistema, seus métodos estão listados:

- init
- get client request
- reply client
- make request

A função de *init* realiza a conexão a partir da porta fornecia, ou da porta 8228 por convenção, cria o *socket* do servidor e realiza o *bind*. A função *get client request* monta um *request* do cliente. O método de *reply client* testa se o envio da resposta do cliente no *socket* deu certo e fecha o *socket* do cliente. Por fim o método *make request* realiza de fato o *request* com o endereço do servidor e retorna um *reply*.

2.5 Spider

A classe de Spider é responsável por gerar a árvore recursiva dos URL subjacentes no mesmo domínio. Suas principais funções são: geraArvore e printArvore. O método de gerar a árvore visita as URL e testa se são válidas e coloca numa lista de URL válidas.

3 Funcionamento

A interface gráfica possui uma série de botões que ajudarão o usuário a usar o sistema. Primeiramente ao clicar no botão de *start* o inspetor HTTP é iniciado pegando as requisições e respostas do browser, porém só é mostrado na tela as mensagens se o usuário clicar nos botões de *request* e *reply*. Abaixo das caixas de texto que mostram as requisições e respostas possui uma caixa de texto que é responsável por receber a URL que o usuário deseja realizar o *spider*, ao clicar no botão *go* ele gera a ávore. Existe também uma caixa de texto onde o usuário deverá colocar uma URL que ele deseja realizar o *dump* de todo o conteúdo da página web. Lembrando que as páginas deverão estar no procotolo HTTP.