Universidade de Brasília

Departamento de Ciência da Computação

Relatório do Inspetor HTTP baseado em Proxy Server

Autores:

Felipe Brandão 12/0044919

Raphael Queiroz 13/0154989

Brasília

10 de Julho de 2018

Conteúdo

1 Apresentação teórica

1.1 Proxy Server Web

Um Proxy Server Web é um servidor (sistema ou aplicação) que atua como um intermediário para requisições HTTP de clientes buscando recursos de servidores Web. O cliente se conecta ao Proxy Server requisitando um objeto de um servidor diferente e esse avalia a requisição, repassando-a para o servidor de origem do objeto. Dessa forma, o Proxy Server se comporta como servidor para o cliente final e como cliente para o servidor de origem.

As utilidades dessa tecnologia são várias, desde monitoramento e filtragem até caching e controle de acesso, porém este trabalho trata apenas de monitorar e controlar o tráfego, podendo editar o que é enviado/recebido.

1.2 TCP

Antes de caracterizar o Transmission Control Protocol (TCP), é importante mencionar sockets, a interface de software entre o processo da aplicação e o protocolo da camada de transporte. A aplicação no lado do emissor envia mensagens através do socket, enquanto do outro lado do socket, o protocolo da camada de transporte tem a responsabilidade entregar as mensagens ao socket do processo receptor. E é pensando na garantia de entrega dessas mensagens que o TCP é escolhido pelo HTTP.

Esse protocolo é orientado à conexão, ou seja, o cliente e o servidor trocam informação de controle antes que as mensagens da aplicação comecem a fluir, e fornece um serviço de entrega confiável dos dados, o que permite que os processos confiem no TCP para entregar sem erro todos os dados enviados e na ordem correta, sem bytes duplicados ou faltando. Além disso, também há um mecanismo para controle de congestionamento que limita cada conexão TCP à sua parte justa da largura de banda da rede.

1.3 HTTP

O HyperText Transfer Protocol (HTTP) está no coração da Web e é responsável por definir a estrutura das mensagens trocadas entre cliente e servidor e como elas são trocadas. Essas mensagens podem ser resumidas a páginas Web, que consistem de objetos (um arquivo HTML, uma imagem, uma folha de estilo CSS ou um arquivo de áudio/vídeo) endereçados por uma Uniform Resource Locator (URL), que por sua vez é dividida em hostname do servidor e nome do caminho do objeto.

Os navegadores Web implementam o cliente HTTP, cujo processo inicia uma conexão TCP com o servidor na porta 80, por padrão, e então envia uma requisição contendo o método (GET, POST, HEAD, PUT e DELETE), a URL e a versão do HTTP na primeira linha seguida pelas linhas do cabeçalho, todas separadas por uma quebra de linha (\cr\lf). A resposta tem uma linha de status como primeira linha, contendo a versão do HTTP, o código e a frase do status, seguida pelas linhas do cabeçalho, uma linha em branco e por fim o corpo da entidade.

Vale lembrar que se trata de um protocolo stateless, ou seja, o servidor não mantém informação sobre clientes antigos, além de que a conexão utilizada por padrão é persistente, fazendo com que o servidor deixe a conexão TCP aberta após enviar o objeto para caso o mesmo cliente tenha mais requisições a fazer.

1.4 Spider e cliente recursivo

Um spider ou Web crawler é um rastreador de páginas muito utilizado para indexação Web que sistematicamente identifica todos os links de uma página e os adiciona a uma lista de URLs a serem visitadas recursivamente. No contexto deste trabalho, por fins de simplicidade, o resultado será uma árvore de páginas HTML contidas no host da página raiz e que podem ser acessadas a partir da mesma.

A parte de dump fica a cargo do cliente recursivo que salvará uma cópia local dos objetos acessíveis pela árvore gerada, se limitando a arquivos HTML e imagens porém mantendo uma estrutura de diretórios semelhante à do servidor remoto.

2 Arquitetura do sistema

A estrutura de diretórios está disposta de forma que a pasta *src* contém os *.cpp*, a pasta *lib* contém os *.h* e a pasta *dump* contém a cópia local da página Web solicitada.

Quanto à codificação dos módulos, a documentação já contém a função de cada um então esta seção fica dedicada à explicação de como os módulos interagem entre si. Na *main*, o construtor do módulo *Server* é chamado para configurar todo o ambiente inicial e então a funcionalidade de inspeção é chamada.

O módulo *Header* é chamado toda vez que uma requisição ou reposta é recebida, interpretando e alterando adequadamente o header e realizando o dump, quando houver um *¡body¿*, ou seja, quando for uma resposta.

3 Documentação

A documentação do código foi feita utilizando Doxygen, por isso se encontra externa a este relatório mas dentro do repositório `https://github.com/Raphaelluissq95/http_insp`.

4 Funcionamento

5 Conclusão

Quanto à funcionalidade de interceptação de requests, as imagens de uma página demoram muito para carregar, mas os demais objetos são carregados normalmente. Também foi detectado que após o término do programa, a conexão TCP ficava aberta por alguns instantes num estado de TIMEWAIT, impossibilitando o bind do socket na mesma porta enquanto esse tempo não terminasse.

Referências

[1] James Kurose e Keith Ross. *Computer Networking. A Top-Down Approach*. 7ª ed. Pearson, 2017. Cap. 2.

[2] *Proxy server*. Wikimedia Foundation, Inc. Jul. de 2018. URL: `https://en.wikipedia.org/wiki/Proxy_server` (acedido em 09/07/2018).

[3] *Web crawler*. Wikimedia Foundation, Inc. Jun. de 2018. URL: `https://en.wikipedia.org/wiki/Web_crawler` (acedido em 09/07/2018).

[4] *Wget. Recursive download*. Wikimedia Foundation, Inc. Jun. de 2018. URL: `https://en.wikipedia.org/wiki/Wget#Recursive_download` (acedido em 09/07/2018).