**Universidade de Coimbra - Faculdade de Ciências e Tecnologias**

**Licenciatura em Engenharia Informática – 2º ano, 2º semestre**

2020/2021

**Redes de Comunicação**

**Relatóriodoprojeto *Message Xchanger* – Meta 1**

**Autores:**

João Filipe Guiomar Artur, 2019217853

Sancho Amaral Simões, 2019217590

**1. Metodologia utilizada para configuração do cenário de rede**

**1.1. Configuração IP**

**1.1.1.** Definição dos endereços *IP* e espaços de endereçamento de cada sub-rede.

**1.1.2.** Atribuição de endereços *IP* a cada dispositivo das várias sub-redes.

**1.1.3.** Teste de conetividade entre todos os pares possíveis de dispositivos pertencentes à mesma sub-rede, através do comando *ping*.

**1.2. Configuração do *routing***

**1.2.1.** Definição estática de rotas em cada *router* (exceto para a rede *DMZ*).

**1.2.2.** Teste de conetividade entre todos os pares possíveis de dispositivos, exceto aqueles que incluem o *server*, através do comando *ping.*

**1.3. Configuração do *NAT* no *router* R3**

**1.3.1.** Configuração do *SNAT* no *router* R3.

**1.3.2.** Configuração de *DNAT* no *router R3*.

**1.3.3.** Testeda conetividade entre qualquer *PC* e o *server* (bidirecional) através do programa *netcat*.

**2. Configurações/dados relativos ao cenário de rede**

**2.1. Rede *DMZ* (*Server*, R3):**

|  |  |  |  |
| --- | --- | --- | --- |
| Endereço de rede (*CIDR*) | Máscara de rede | Endereço de *broadcast* | Gama de endereçamento |
| 10.90.0.0/24 | 255.255.255.0 | 10.90.0.255 | 10.90.0.1 - 10.90.0.254 |

*Tabela 1* *-* Dados *IPv4* da rede *DMZ*

|  |  |  |
| --- | --- | --- |
|  | Endereço *IPv4* | *Default gateway* |
| Router R3 – *f1/0* | 10.90.0.1 | ----------------- |
| *Server* – *eth0* | 10.90.0.2 | 10.90.0.1 |

*Tabela 2* - Configuração *IPv4* dos dispositivos presentes na rede DMZ

**2.2. Rede interna (R1, R2, R3):**

|  |  |  |  |
| --- | --- | --- | --- |
| Endereço de rede (*CIDR*) | Máscara de rede | Endereço de *broadcast* | Gama de endereçamento |
| 193.136.212.128/29 | 255.255.255.248 | 193.136.212.135 | 193.136.212.129 – 193.212.212.134 |

*Tabela 3* - Dados *IPv4* da rede interna

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Router* R1 – *f1/0* | *Router* R2 – *e0/0* | *Router* R3 – *e0/0* |
| Endereço *IPv4* | 193.136.212.130 | 193.136.212.131 | 193.136.212.129 |

*Tabela 4* - Configuração *IPv4* dos dispositivos presentes na rede interna

**2.3. Rede externa esquerda (PC1, PC2, R1):**

|  |  |  |  |
| --- | --- | --- | --- |
| Endereço de rede (*CIDR*) | Máscara de rede | Endereço de *broadcast* | Gama de endereçamento |
| 193.136.212.136/29 | 255.255.255.248 | 193.136.212.143 | 193.136.212.137 - 193.136.212.142 |

*Tabela 5* - Dados *IPv4* da rede externa esquerda

|  |  |  |
| --- | --- | --- |
|  | Endereço *IPv4* | *Default gateway* |
| Router R1 – *e0/0* | 193.136.212.137 | ---------------------- |
| PC1 – *eth0* | 193.136.212.141 | 193.136.212.137 |
| PC2 – *eth0* | 193.136.212.142 | 193.136.212.137 |

*Tabela 6* - Configuração *IPv4* dos dispositivos presentes na rede externa esquerda

**2.4. Rede externa direita (PC3, PC4, R2):**

|  |  |  |  |
| --- | --- | --- | --- |
| Endereço de rede (*CIDR*) | Máscara de rede | Endereço de *broadcast* | Gama de endereçamento |
| 193.136.212.144/29 | 255.255.255.248 | 193.136.212.151 | 193.136.212.145 - 193.136.212.150 |

*Tabela 7* - Dados IPv4 da rede externa direita

|  |  |  |
| --- | --- | --- |
|  | Endereço *IPv4* | *Default gateway* |
| *Router* R2 – *f1/0* | 193.136.212.145 | ---------------------- |
| PC3 – *eth0* | 193.136.212.149 | 193.136.212.145 |
| PC4 – *eth0* | 193.136.212.150 | 193.136.212.145 |

*Tabela 8* - Configuração *IPv4* dos dispositivos presentes na rede externa direita

**2.5. Tabelas de *routing* (para redes não ligadas de forma direta)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Endereço de rede (*CIDR*) | Máscara de rede | *Default gateway* |
| *Router* R1 | 193.136.212.144/29 | 255.255.255.248 | 193.136.212.131 |
| *Router* R2 | 193.136.212.136/29 | 255.255.255.248 | 193.136.212.130 |
| *Router* R3 | 193.136.212.144/29 | 255.255.255.248 | 193.136.212.131 |
| 193.136.212.136/29 | 255.255.255.248 | 193.136.212.130 |

*Tabela 9*- Configuração do roteamento dos três *routers*

**2.6 Configuração do *NAT* do *router* 3(*DNAT* e *SNAT*)**

**interface FastEthernet1/0**

**ip address 10.90.0.1 255.255.255.0**

**ip nat inside source list 30 interface Ethernet0/0 overload**

**ip nat inside source static tcp 10.90.0.2 9000 193.136.212.129 9000 extendable**

**ip nat inside source static udp 10.90.0.2 9000 193.136.212.129 9000 extendable**

**access-list 30 permit 10.90.0.0 0.0.0.255**

**3. Detalhes de implementação e arquitetura**

**3.1 Armazenamento de dados**

Com a finalidade de armazenar em memória os registos dos utilizadores e as respetivas sessões procedeu-se à implementação de uma árvore binária de pesquisa (*BST*). Ainda foi considerada a utilização de uma árvore *AVL* totalmente genérica, no entanto, por escassez de tempo e para efeitos de maior simplicidade optou-se pela *BST* simples. Quanto ao armazenamento dos registos em disco foi utilizada a seguinte metodologia: no primeiro arranque do servidor, os dados iniciais são lidos de um ficheiro de texto; no final dessa primeira execução os dados contidos em memória são então colocados num ficheiro binário. Nas execuções seguintes, o ficheiro binário referido passa a ser o utilizado para transpor os dados para a memória volátil. Escolheu-se esta abordagem dado que os ficheiros binários são mais eficientes do que os de texto e não necessitam de qualquer *parsing*.

**3.2. Servidor**

Visto que o servidor suporta *TCP* e *UDP* ao mesmo tempo, são criadas, durante a sua inicialização, duas *threads* que tratam de cada um dos referidos protocolos. Esta solução permite assim aumentar a concorrência do programa. Relativamente ao protocolo *UDP* são criadas *subthreads* da respetiva *thread* que tratam cada uma das sessões dos clientes que comunicam com o servidor via *UDP*. Estas *subthreads* são criadas no momento em que o utilizador insere as credenciais corretas na *CLI* do cliente, previamente definidas pelo administrador. Cada novo comando que o utilizador insere é transmitido para a *thread* que trata da sua sessão, através de uma *message queue*. O *type* de cada mensagem enviada para a *message queue* consiste num *ID* gerado artificialmente, após criação de um novo utilizador, através da incrementação do número total de utilizadores. Quanto ao *TCP* não se verifica grande complexidade na sua integração no servidor, visto que bastou criar a respetiva *worker thread* e algumas funções auxiliares para validação dos comandos inseridos pelo administrador.

**3.3.** **Comunicação mediada pelo servidor / pedidos ao servidor**

Para facilitar o *parsing* das informações trocadas entre os vários clientes e o servidor (bem como aumentar a eficiência das comunicações), optou-se por utilizar um esquema de pedido/resposta cujas mensagens possuem uma estrutura fixa e bem definida. Além dos métodos a utilizar pelo cliente/servidor, à semelhança do protocolo *HTTP*, são também fornecidas variáveis que contêm possíveis parâmetros para estes mesmos protocolos.

**3.4. Comunicação *P2P***

O método de comunicação do *P2P* implementado no lado dos clientes é muito semelhante ao implementado no lado do servidor, na medida que é utilizada uma *thread* que fica à escuta de possíveis mensagens de outros utilizadores.

**3.5. Comunicação por *multicast***

Funcionalidade não implementada.