ENGENHARIA DA COMPUTAÇÃO
REDES DE COMPUTADORES A

PROJETO 2
WhatsAp2p - comunicação P2P

EQUIPE:
Agostinho Sanches de Araújo ---------- RA: 16507915
Evandro Douglas Capovilla Junior ---------- RA: 16023905
Lucas Tenani Felix Martins ---------- RA: 16105744
Pedro Andrade Caccavaro ---------- RA: 16124679

11/06/2019

SUMÁRIO

INTRODUÇÃO

P2P, ou peer-to-peer, é uma rede de computadores que compartilham arquivos pela internet. É uma comunicação onde não há necessidade de um servidor central que armazene os dados mas sim de usuários que disponibilizam downloads para que outros busquem arquivos em sua máquina. Nesse sistema cada computador funciona como servidor e cliente ao mesmo tempo.

OBJETIVO

A proposta desse projeto é criar um “aplicativo” de mensagens e envio de fotos que funcione através de uma rede P2P, onde um cliente tem acesso à um servidor pra quem faz requisições, esse servidor deve retornar o endereço do contato requisitado para que o cliente faça a comunicação diretamente com outro, o servidor deve apenas armazenar temporariamente o status de cada cliente: se está online ou não, endereço IP e porta onde está esperando conexões.

DESCRIÇÃO

1. DESCRIÇÃO CLIENTE

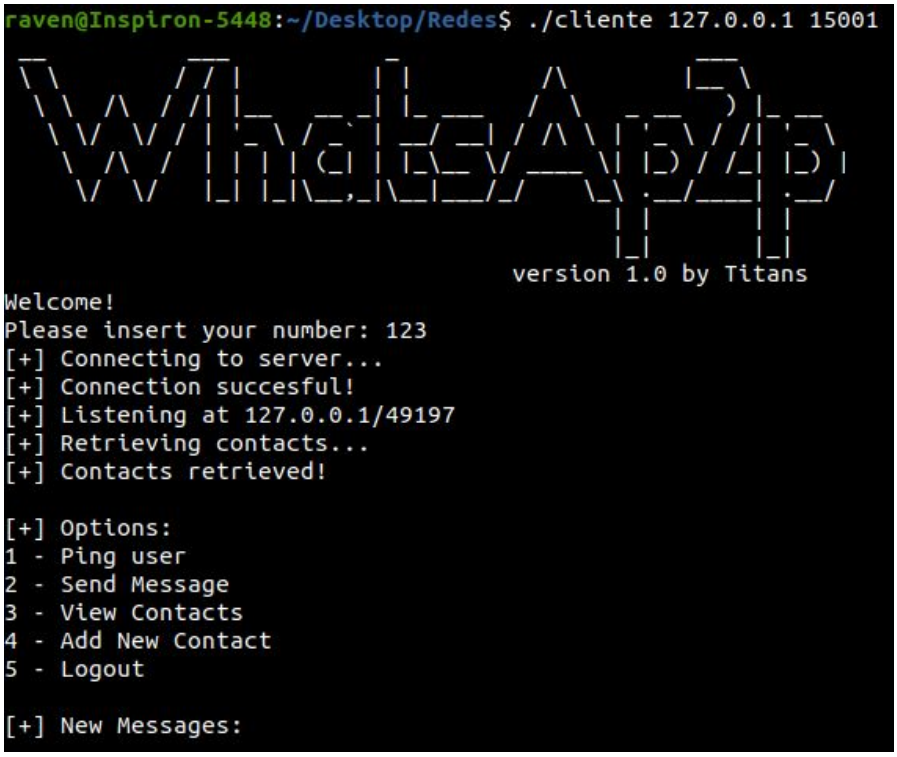

Figura 1: Tela inicial Cliente

O cliente é inicializado passando como argumentos o endereço IP e a porta do servidor central. A tela inicial (fig. 1) mostra a visão inicial do programa, novas mensagens que serão exibidas embaixo do menu e as opções disponíveis:

- *Ping user*: O cliente passa um número que será enviado no servidor e este retornará o endereço e porta do número requisitado, caso esteja *online*, do contrário aparecerá uma mensagem avisando que está *offline*.
- *Send message:* Abre uma tela para envio de mensagem onde é pedido o número de destino e a mensagem à ser enviada, no servidor é checado se o número está online e retorna sua localização.
- *View contacts:* Abre o arquivo com os contatos salvos e os mostra na tela, mostrando quais estão online e sua localização.

- *Add new contact:* Permite adicionar um novo número na lista de contatos.
- *Logout:* Notifica servidor da desconexão e fecha o aplicativo.

2. DESCRIÇÃO SERVIDOR

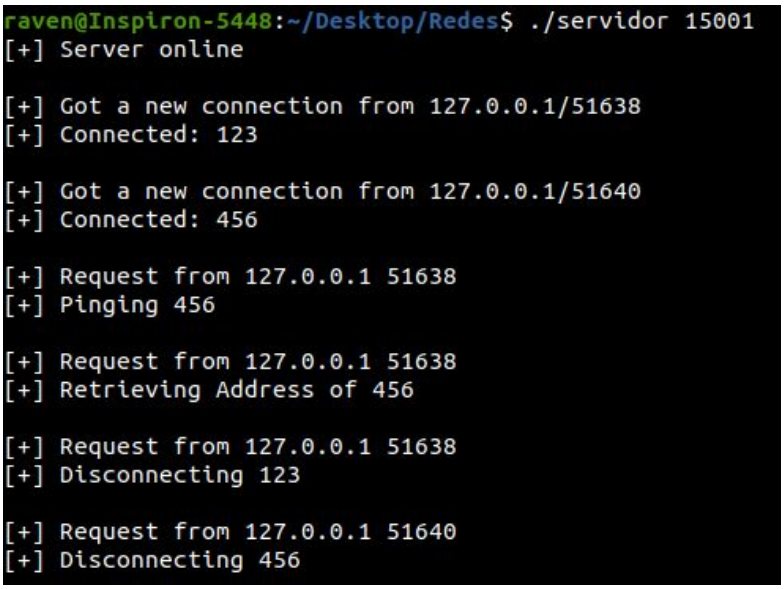

Figura 2: Servidor.

O servidor é iniciado e então fica esperando por conexões dos usuários, ele apenas envia informações para os usuários, mostrado um *feedback* das ações e quem as requisitou.

3. DESCRIÇÃO DO PROJETO

a. Geral

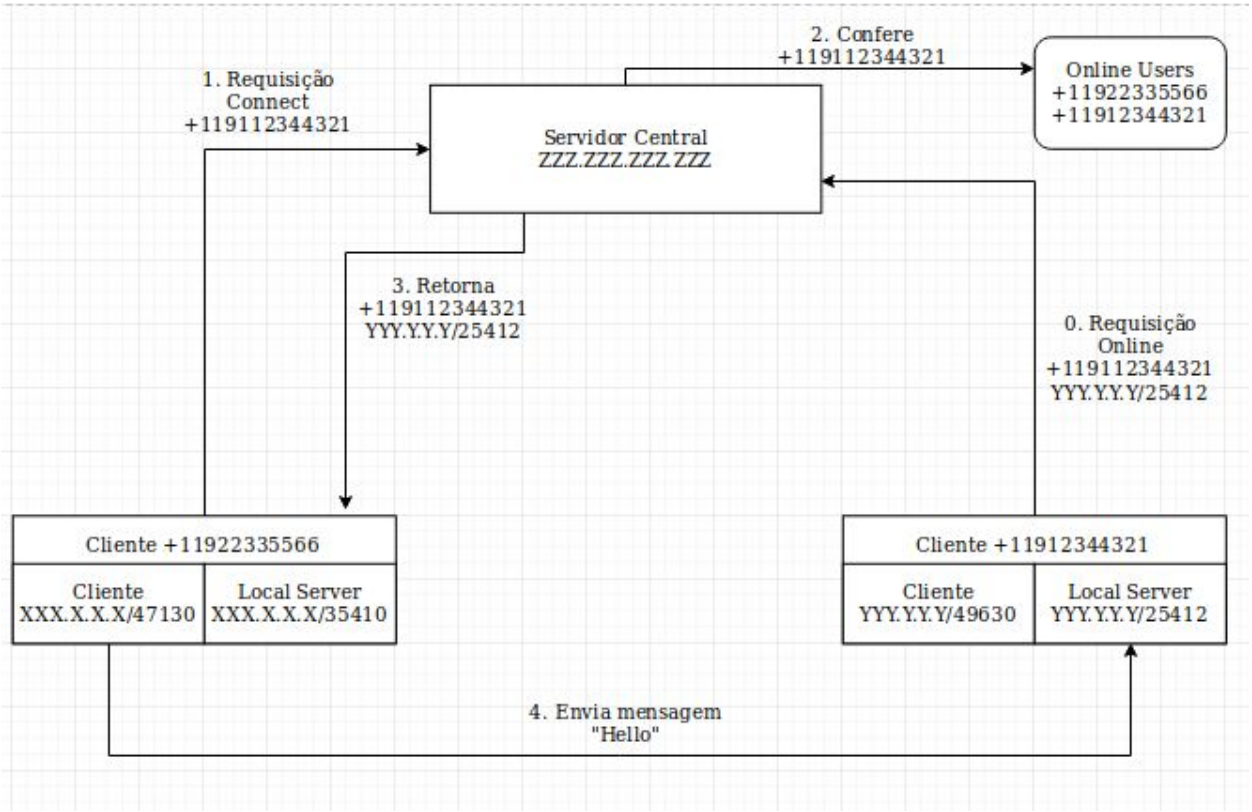

Figura 3: Fluxograma da Interação da conexão.

O usuário ao conectar pede ao usuário que digite o número de telefone, que servirá como identificação no servidor e então é criada a conexão com o servidor central, usando os dados recebidos pela linha de comando.

Depois, a função *createLocalServer* é chamada para criar a parte de servidor do cliente, para esperar por conexões de outros usuários, criando *local_server* que será usada na *thread*. Disparamos a *server_thread* para que o cliente fique esperando mensagens de outros clientes. Agora, com o endereço onde o cliente ficará esperando, podemos enviar as informações ao servidor. O servidor recebe as informações sobre o cliente e o "conecta", salvando suas informações na lista de usuários *online* e então dispara a *thread server_exec* que recebe como parâmetro o *handler* do cliente que se conectou, salvando seu *socket* e mantendo a conexão ativa, esperando requisições do cliente.

O menu é apresentado ao cliente, escolhendo a opção "*Ping user*" é requisitado o número do cliente que ele quer checar, e é enviado ao servidor

o comando *"check"*. O servidor então confere na lista de usuários ativos se o usuário requisitado está *online*, caso esteja, suas informações são enviadas ao cliente.

Na opção *"Send Message"*, é requisitado o número a quem se destina a mensagem e então é enviado um comando *"connect"* ao servidor para conferir se o destinatário está ativo, caso contrário não será enviada mensagem. No caso de estar ativo, a função *goChat* é chamada, ela cria a conexão com o cliente destinatário e salva o *chat_socket* dessa conexão, então o usuário pode digitar mensagem a ser enviada.

As duas últimas opções referentes à contatos consistem apenas na leitura do arquivo local com os contatos. Para lista é feita a checagem de usuário "pingados" pelo cliente e suas informações são mostradas. A inserção de um novo contato é uma escrita de arquivo simples.

O *logout* é feito através do envio de uma requisição para o servidor de desconexão, o servidor então encontra o usuário a ser desligado na lista e reescreve as informações e fecha o *socket* de conexão.

b. Dados

Para armazenamento das informações e envio de dados através dos *sockets* foram criadas estruturas, explicadas abaixo:

- *Message*
  - *Num_sender:* número que enviou a mensagem
  - *Text:* mensagem de texto

- *Command*
  - *Cmd:* especifica a requisição que deve ser feita para o servidor, podendo ser: *Online*, para conectar-se; *Offline*, para desconectar-se; *Check*, para conferir a lista de usuários conectados
  - *Number:* número que realizou a requisição

- *Info (Armazena informações relevantes)*
  - *Number:* número de telefone
  - *IP:* endereço de IP
  - *Port:* porta
  - *Socket:* socket

RESULTADOS

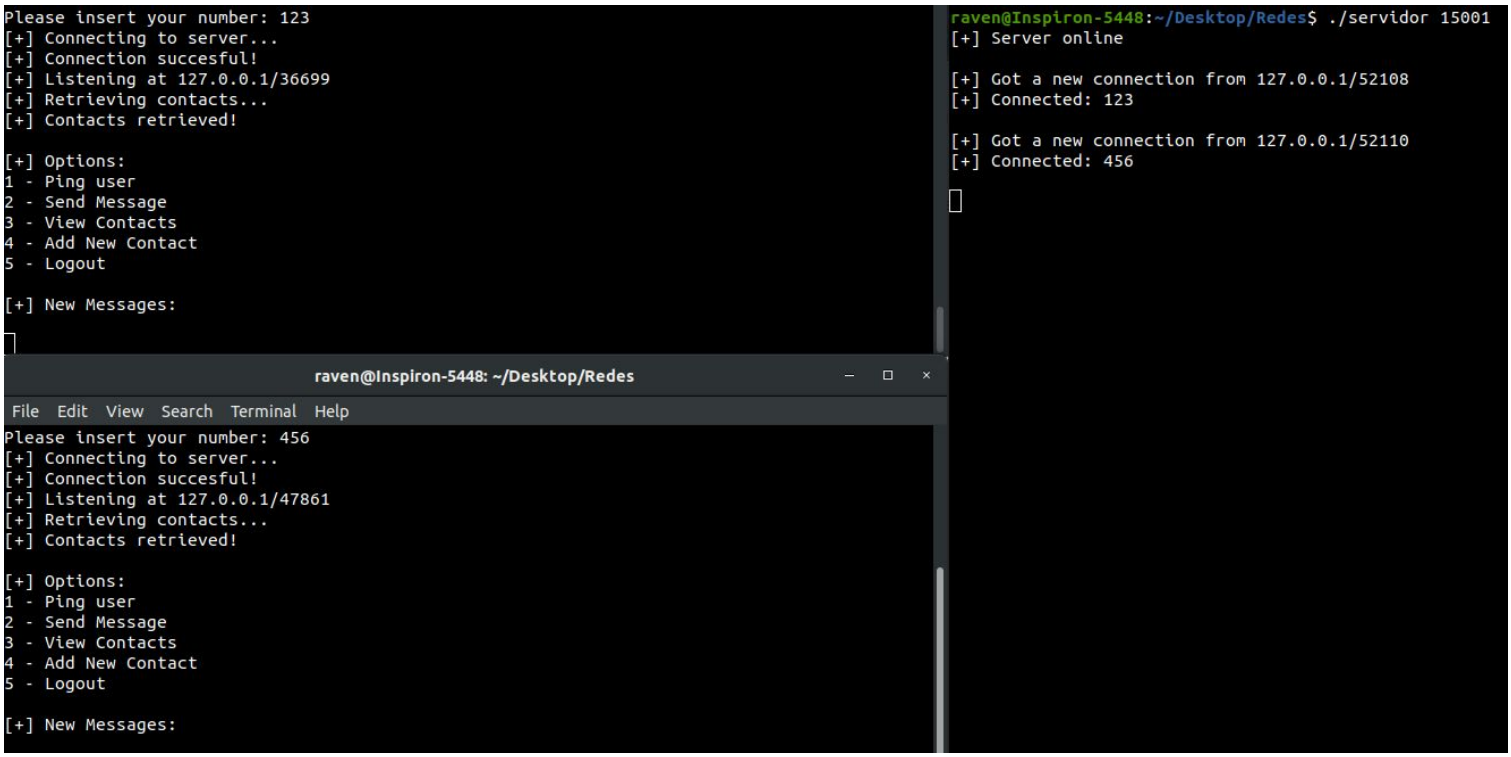

Figura 4: Conexão de dois clientes no servidor.

Figura 5: Demonstração Ping.

Figura 5: Lista de Contatos.

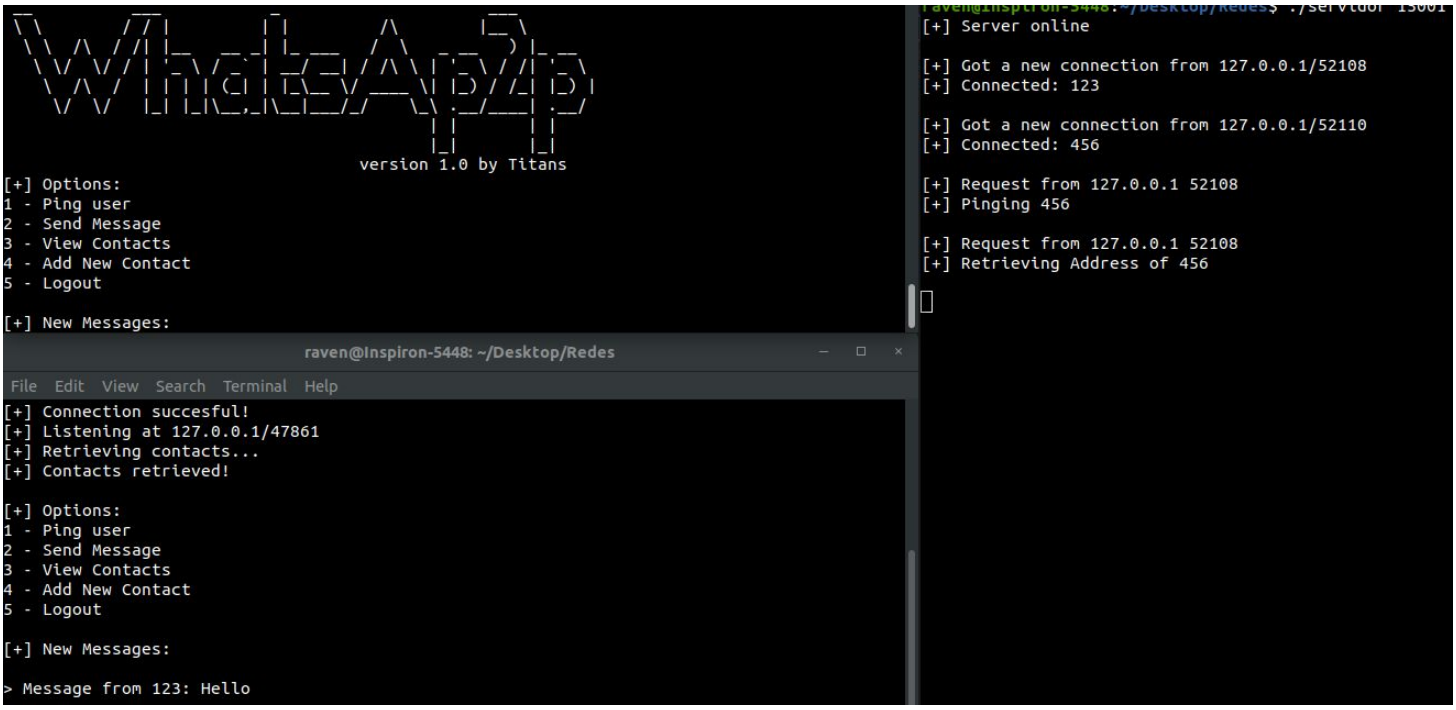

Figura 6: Mensagem Recebida no cliente 456.

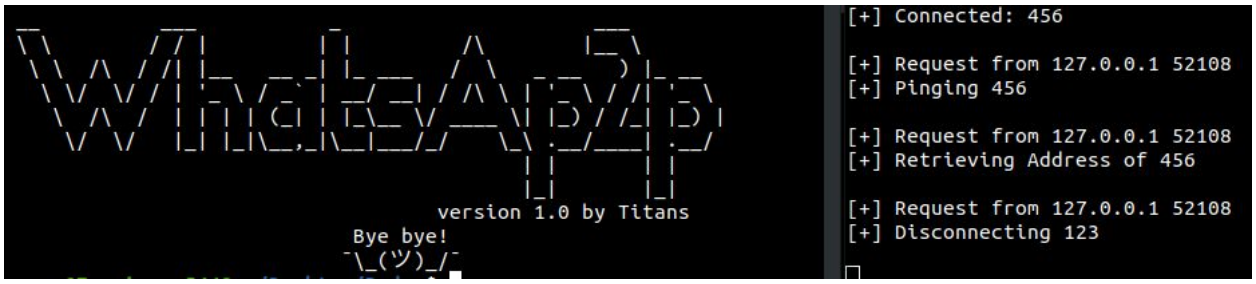

Figura 7: Finalizando Cliente 123.

OBSERVAÇÕES

- Não houve implementação de grupos de contatos
- Não houve implementação da opção de envio de imagens

CONCLUSÃO

Este projeto teve como objetivo a familiarização com o modelo de conexão *peer-to-peer*, através de um aplicativo de envio de mensagens entre clientes. Durante sua realização pudemos aprofundar os conhecimentos das relações entre cliente servidor, percebemos melhor como funcionam as portas de rede, a relação dos *sockets*, além disso aperfeiçoamos nosso conhecimento técnico em relação à linguagem C e o modo que ela cria conexões. Finalizamos o Projeto com resultados satisfatórios, cumprindo com o objetivo inicial e o que fora proposto para o projeto.