**Agenda de contatos utilizando RMI**

Fernando L. Klein - 2159562, Marcos V. Rocha -2159678

O programa funciona de forma distribuída, em que há um servidor que disponibiliza os serviços de CRUD e os clientes podem se conectar a ele para realizar essas operações.

As principais operações disponíveis são:

* **Adicionar novo contato:** o cliente envia as informações do novo contato para o servidor, que adiciona o registro em seu banco de dados.
* **Atualizar os dados de um contato:** o cliente envia as informações atualizadas do contato, que o servidor utiliza para atualizar o registro correspondente em seu banco de dados.
* **Buscar um contato pelo ID:** o cliente envia o ID do contato que deseja buscar e o servidor retorna as informações correspondentes do registro.
* **Listar todos os contatos:** o cliente solicita ao servidor a lista de todos os contatos cadastrados em seu banco de dados e o servidor retorna esses registros.
* **Excluir algum contato:** o cliente envia o ID do contato que deseja excluir e o servidor remove o registro correspondente do banco de dados.

A implementação do programa utiliza a biblioteca Pyro4 para realizar a comunicação entre o servidor e os clientes. O servidor expõe uma interface remota que define os métodos disponíveis para CRUD e os clientes se conectam a essa interface para realizar essas operações. O banco de dados utilizado é um simples dicionário Python que armazena os registros de contatos.

**Analise dos pacotes com Wireshark**

Dentro do Wireshark, é possível visualizar a comunicação de rede entre o servidor Pyro4 e os clientes. Ao iniciar a captura no Wireshark, é possível ver os pacotes de rede que são enviados e recebidos pelo servidor Pyro4 durante a comunicação com os clientes.

O Wireshark permite visualizar detalhes sobre os pacotes capturados, como o número de sequência e de reconhecimento, identificadores de porta, flags de controle TCP, bem como o conteúdo do pacote em si.

Quando um servidor Pyro4 é iniciado, ele registra um ou mais objetos Python que deseja expor para os clientes remotos. O registro desses objetos é feito através do Pyro4 Daemon, que é responsável por gerenciar as conexões de rede, receber solicitações dos clientes e encaminhá-las para o objeto Python correspondente.

O registro do objeto é feito através do método register do Pyro4 Daemon. Este método recebe como argumento o objeto Python que deseja ser exposto, um identificador único (URI) para o objeto e um nome de serviço opcional para tornar mais fácil para os clientes encontrarem o objeto.

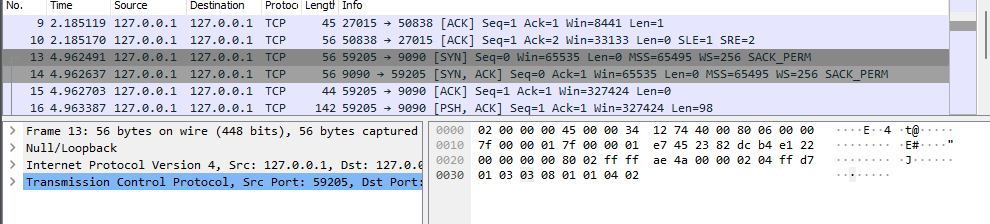
Uma vez que o objeto é registrado, ele se torna disponível para ser acessado pelos clientes remotos através do Pyro4 Proxy. O Proxy é usado pelos clientes para obter uma referência ao objeto remoto e chamar métodos nele como se fossem métodos locais.

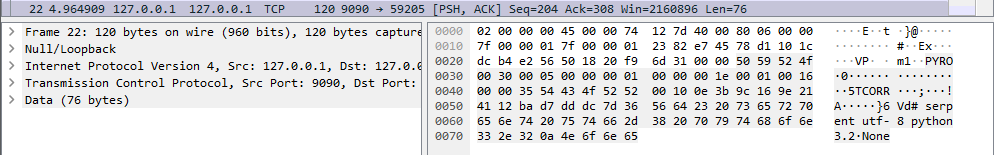
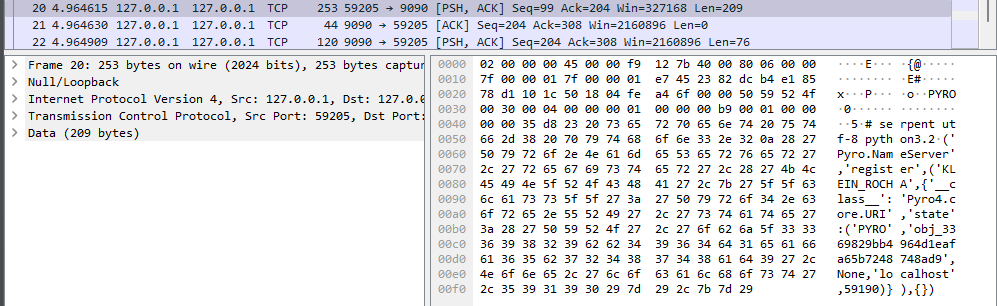
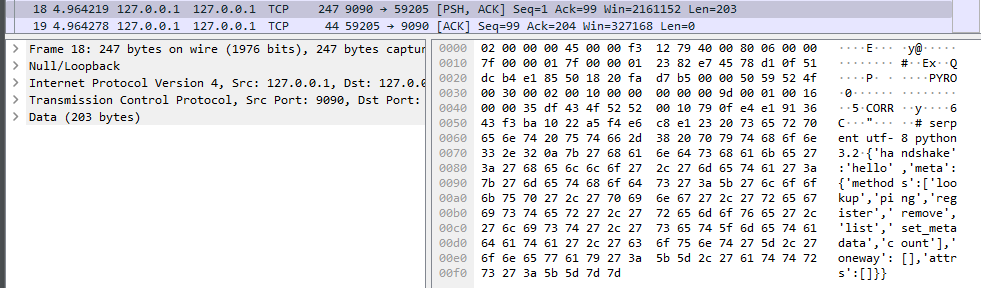
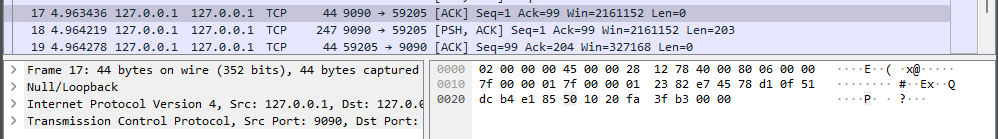
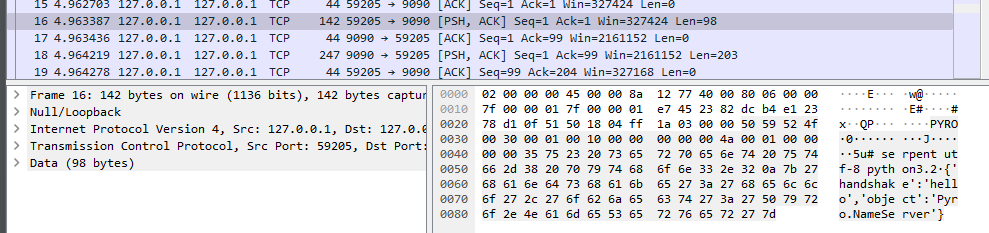
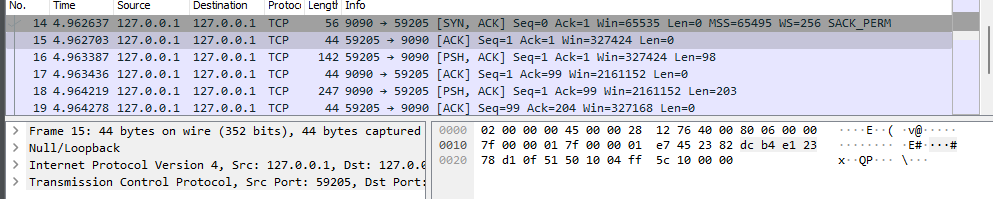
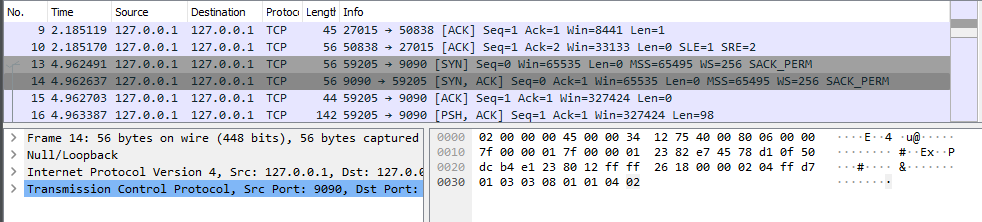
O servidor Pyro4 também pode ser configurado para usar recursos adicionais, como filtros de segurança para controlar quais clientes podem acessar quais objetos remotos, ou namespaces para organizar e agrupar objetos relacionados em uma hierarquia de nomes.

Depois de registrar todos os objetos que deseja expor, o servidor Pyro4 aguarda por conexões de clientes em uma porta de rede específica. Quando um cliente se conecta, o Pyro4 Daemon inicia uma nova thread para lidar com as solicitações do cliente e a comunicação de rede necessária para atendê-las.

Em resumo, quando o servidor é iniciado, ele registra os objetos Python que deseja expor e aguarda conexões de clientes em uma porta de rede específica. Quando um cliente se conecta, o Pyro4 Daemon gerencia a comunicação de rede necessária para permitir que o cliente acesse e interaja com os objetos registrados.

Servidor iniciando





Cliente conectando