**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.

**Servidor iniciando**

Primeiro é iniciado o servidor *nameservice* na porta 9090 com o comnado:

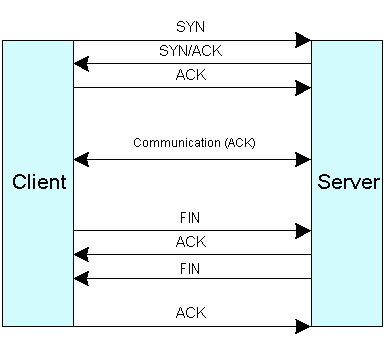
|  |
| --- |
| python -m Pyro4.naming |

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.

Inicialmente foi enviado um SYN da porta 59205 para o *nameservice* porta 9090.

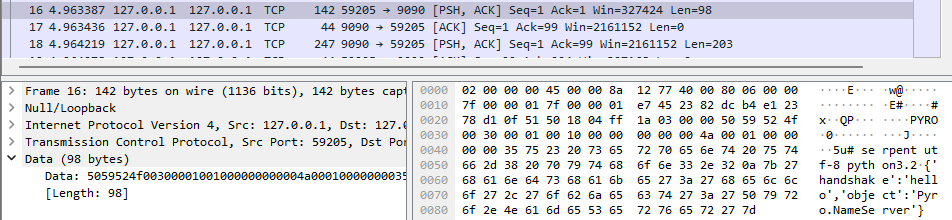


SYN é o primeiro passo na conexão TCP de três vias, que é usada para estabelecer uma conexão entre dois dispositivos na Internet. Quando um dispositivo deseja iniciar uma conexão com outro, ele envia um pacote TCP com o bit SYN definido. Isso informa ao dispositivo de destino que o dispositivo de origem deseja estabelecer uma conexão.



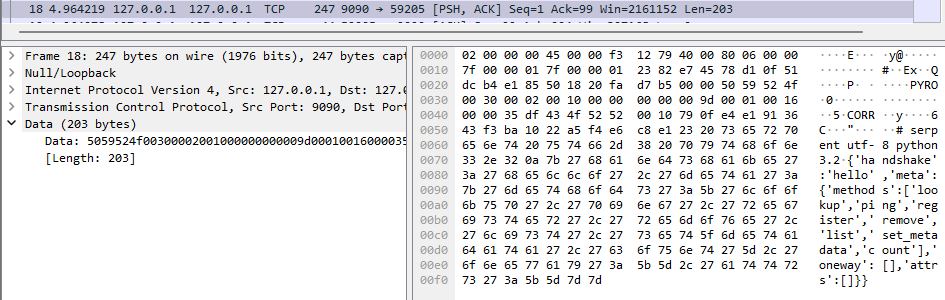
O dispositivo de destino responde com um pacote TCP que tem o bit SYN e o bit ACK (Acknowledgment) definidos. Isso informa ao dispositivo de origem que a conexão foi estabelecida com sucesso e que pode começar a enviar dados.

O servidor na porta 59205 envia uma saudação ("hello") para *nameservice* na 9090 e informa que está se comunicando com o Pyro.NameServer



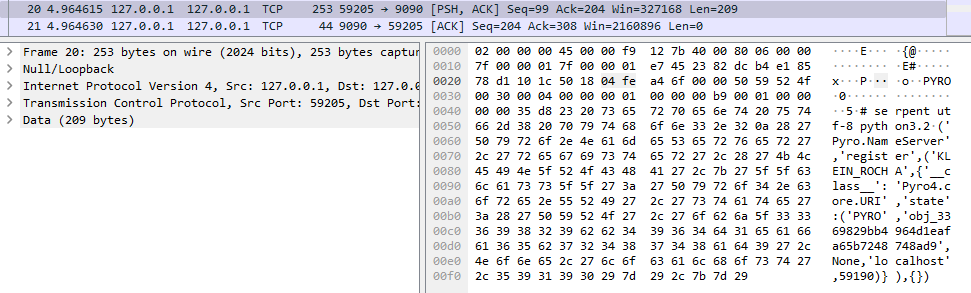
|  |
| --- |
| {'handshake':'hello','object':'Pyro.NameServer'} |

O *nameservice* responde com uma mensagem de saudação ("hello") e envia informações sobre os métodos que estão disponíveis para chamada remota.



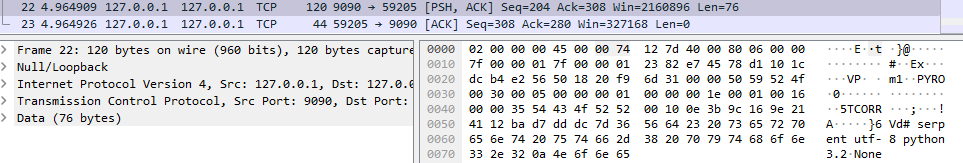
|  |
| --- |
| {'handshake':'hello','meta':{'methods':['lookup','ping','register','remove','list','set\_metadata','count'],'oneway':[],'attrs':[]}} |

O servidor então envia a solicitação de registrar o objeto no *nameservice.*



|  |
| --- |
| ('Pyro.NameServer','register',('KLEIN\_ROCHA',{'\_\_class\_\_':'Pyro4.core.URI','state':('PYRO','obj\_3369829bb4964d1eafa65b7248748ad9',None,'localhost',59190)}),{}) |

O *nameservice*  responde com ‘None’



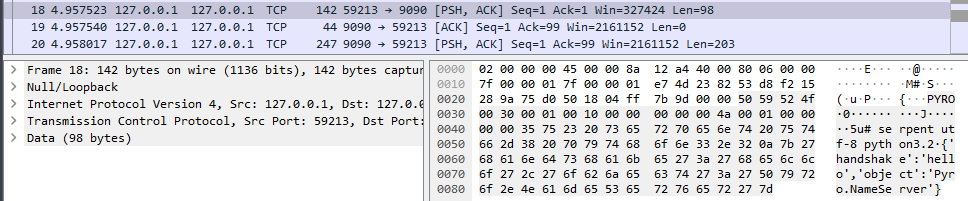
|  |
| --- |
| None |

**Cliente conectando**

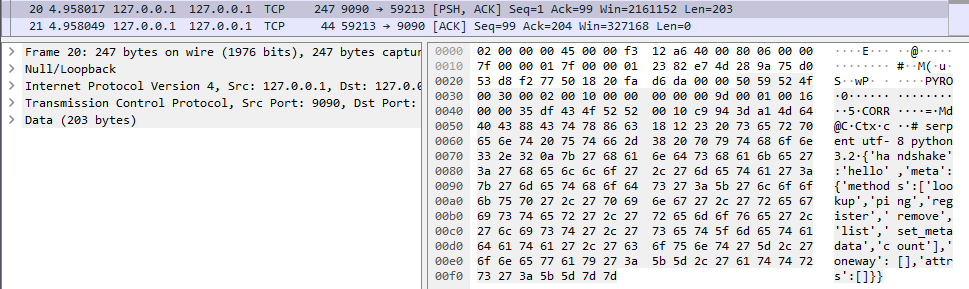
Quando o cliente na porta 59213 tenta se conectar, é enviado um SYN para o *nameservice* na porta 9090.



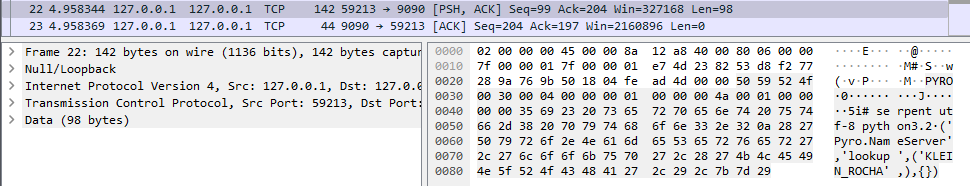
Após estabelecer uma conexão entre os dois dispositivos. O cliente envia uma mensagem de saudação (handshake) para o *nameservice*, indicando que deseja se comunicar com o objeto Pyro.NameServer. A mensagem é enviada da porta 59213 para a porta 9090.



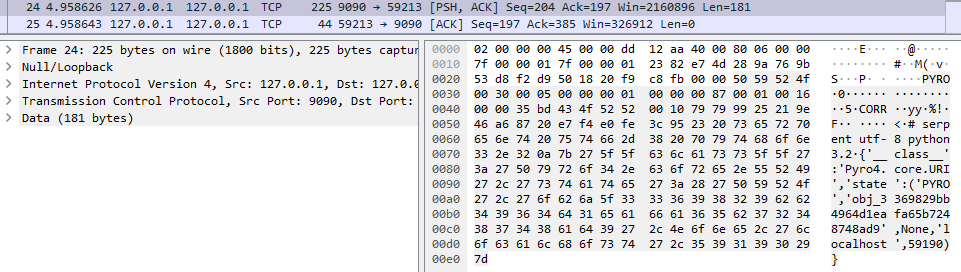
|  |
| --- |
| {'handshake':'hello','object':'Pyro.NameServer'} |

O *nameservice* responde à mensagem de saudação do cliente, enviando uma mensagem de meta-informações (meta) que indica quais métodos podem ser chamados no objeto Pyro.NameServer. 

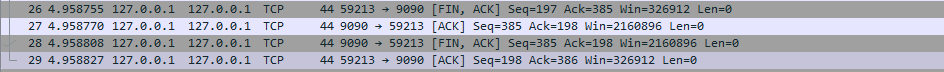
|  |
| --- |
| {'handshake':'hello','meta':{'methods':['lookup','ping','register','remove','list','set\_metadata','count'],'oneway':[],'attrs':[]}} |

O cliente envia uma mensagem de solicitação (lookup) para o objeto Pyro.NameServer, indicando que deseja buscar um objeto remoto chamado "KLEIN\_ROCHA". 

|  |
| --- |
| ('Pyro.NameServer','lookup',('KLEIN\_ROCHA',),{}) |

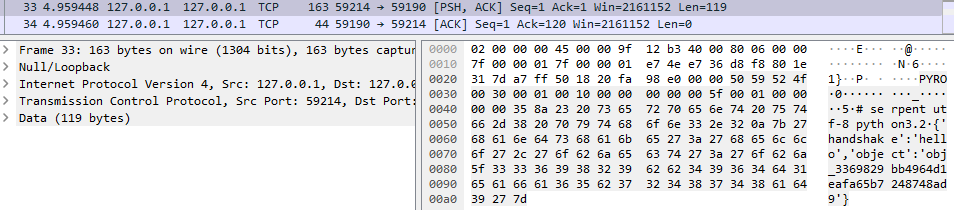
O *nameservice* responde à mensagem de solicitação do cliente, enviando uma mensagem contendo a URI do objeto remoto solicitado. A URI é uma representação serializada do objeto que permite que o cliente se conecte diretamente ao objeto remoto. 

|  |
| --- |
| {'\_\_class\_\_':'Pyro4.core.URI','state':('PYRO','obj\_3369829bb4964d1eafa65b7248748ad9',None,'localhost',59190)} |

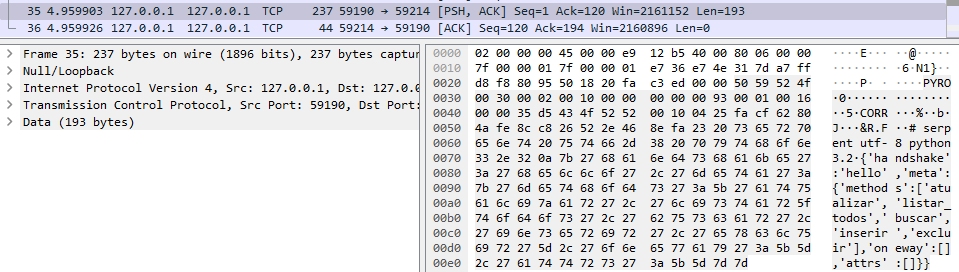
O cliente envia uma mensagem de encerramento (FIN ACK) para o servidor, indicando que a comunicação com o servidor Pyro.NameServer foi concluída. 

O cliente se conecta diretamente ao objeto remoto usando a URI. A mensagem é enviada do endereço na porta 59214 para o servidor na porta 59190. 

O cliente na porta 59214 envia para o servidor na porta 59190:



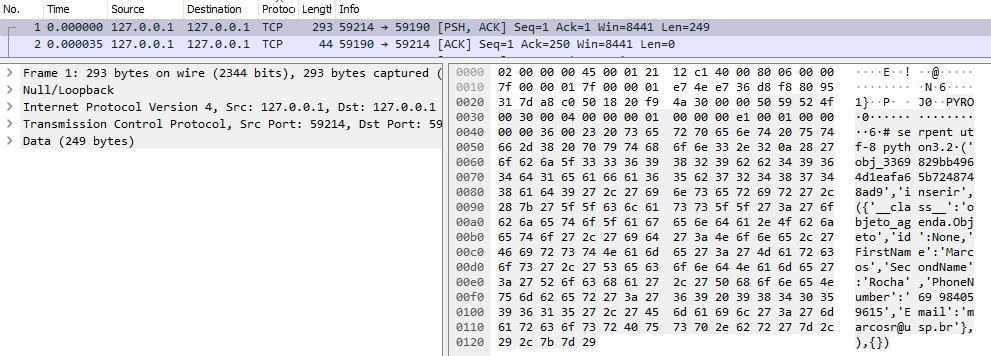
|  |
| --- |
| {'handshake':'hello','object':'obj\_3369829bb4964d1eafa65b7248748ad9'} |

O servidor responde à mensagem de conexão do cliente, enviando uma mensagem de meta-informações (meta) que indica quais métodos podem ser chamados no objeto remoto. A mensagem é enviada do servidor na porta59190 para o cliente na porta 59214. 

|  |
| --- |
| {'handshake':'hello','meta':{'methods':['atualizar','listar\_todos','buscar','inserir','excluir'],'oneway':[],'attrs':[]}} |

**Criando um novo registro**

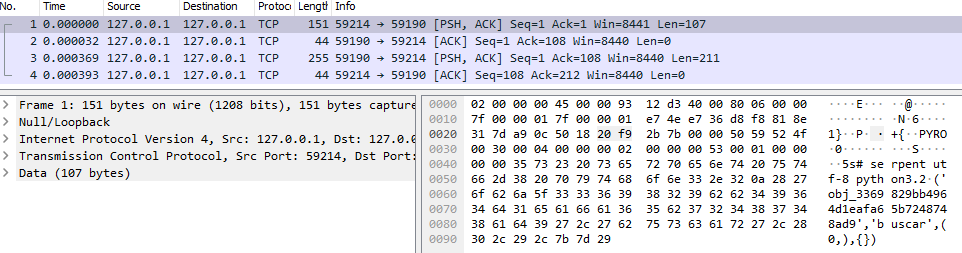
O cliente na porta 59214 envia o PSH utilizando o método ‘inserir’ com os dados que deseja cadastrar para o servidor na porta 59190:



|  |
| --- |
| ('obj\_3369829bb4964d1eafa65b7248748ad9','inserir',({'\_\_class\_\_':'objeto\_agenda.Objeto','id':None,'FirstName':'Marcos','SecondName':'Rocha','PhoneNumber':'69 984059615','Email':'marcosr@usp.br'},),{}) |

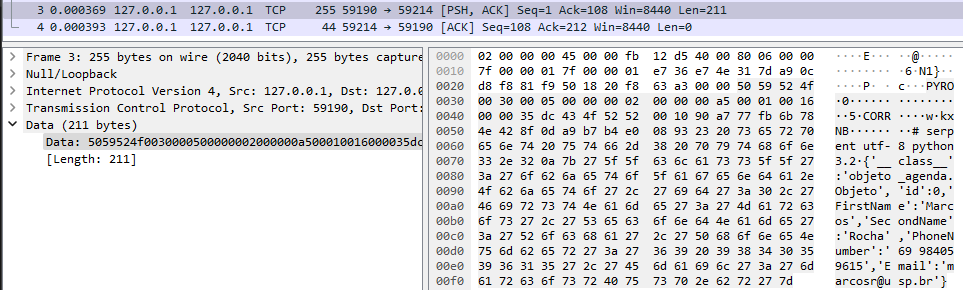
**Listando um usuário**

O cliente na porta 59214 envia para o servidor na porta 59190 a solicitação de buscar o usuário de ID 0 através do método ‘buscar’:



|  |
| --- |
| ('obj\_3369829bb4964d1eafa65b7248748ad9','buscar',(0,),{}) |

O servidor responde com os dados solicitados:



|  |
| --- |
| {'\_\_class\_\_':'objeto\_agenda.Objeto','id':0,'FirstName':'Marcos','SecondName':'Rocha','PhoneNumber':'69 984059615','Email':'marcosr@usp.br'} |