Hywre Cesar
Mateus Magalhães

{hcbp, mnbm}@cin.ufpe.br

RPC-CS

IF711 - Programação Concorrente Distribuída

Projeto Final

Objetivos

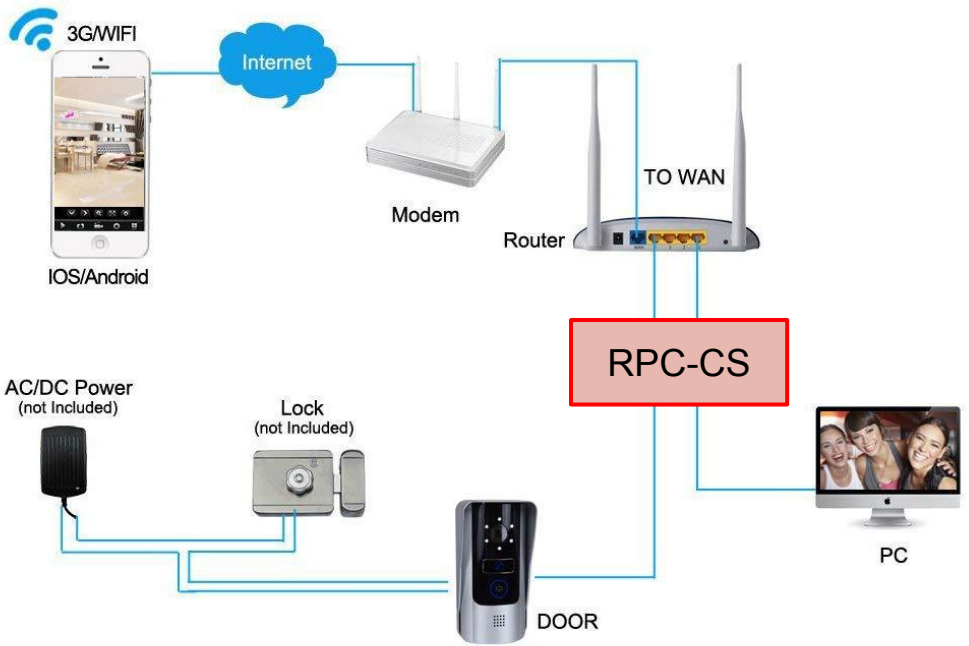
- Construir um middleware em C++ que proveja chamada remota de procedimentos, utilizando os conceitos vistos em sala de aula.

- Demonstrar o uso de tal middleware para o transporte de alguns tipos de dados primitivos e arquivos binários. Provendo funcionalidade de compressão e encriptação.

- Comparar o desempenho do middleware com o XML-RPC.

Caso de Uso

Envio de imagens capturadas por uma fechadura eletrônica conectada a rede Wi-Fi. As imagens devem ser encriptadas e comprimidas antes de serem enviadas.



Requisitos Funcionais

ID	Descrição
RF01	Permitir a chamada remota de procedimentos.
RF02	Fornecer interface de métodos remotos para o cliente.
RF03	Serializar dados (tipos primitivos, vetores, arquivos).
RF04	Criptografar dados.
RF05	Comprimir dados.
RF06	Conexões simultâneas.

Requisitos Funcionais

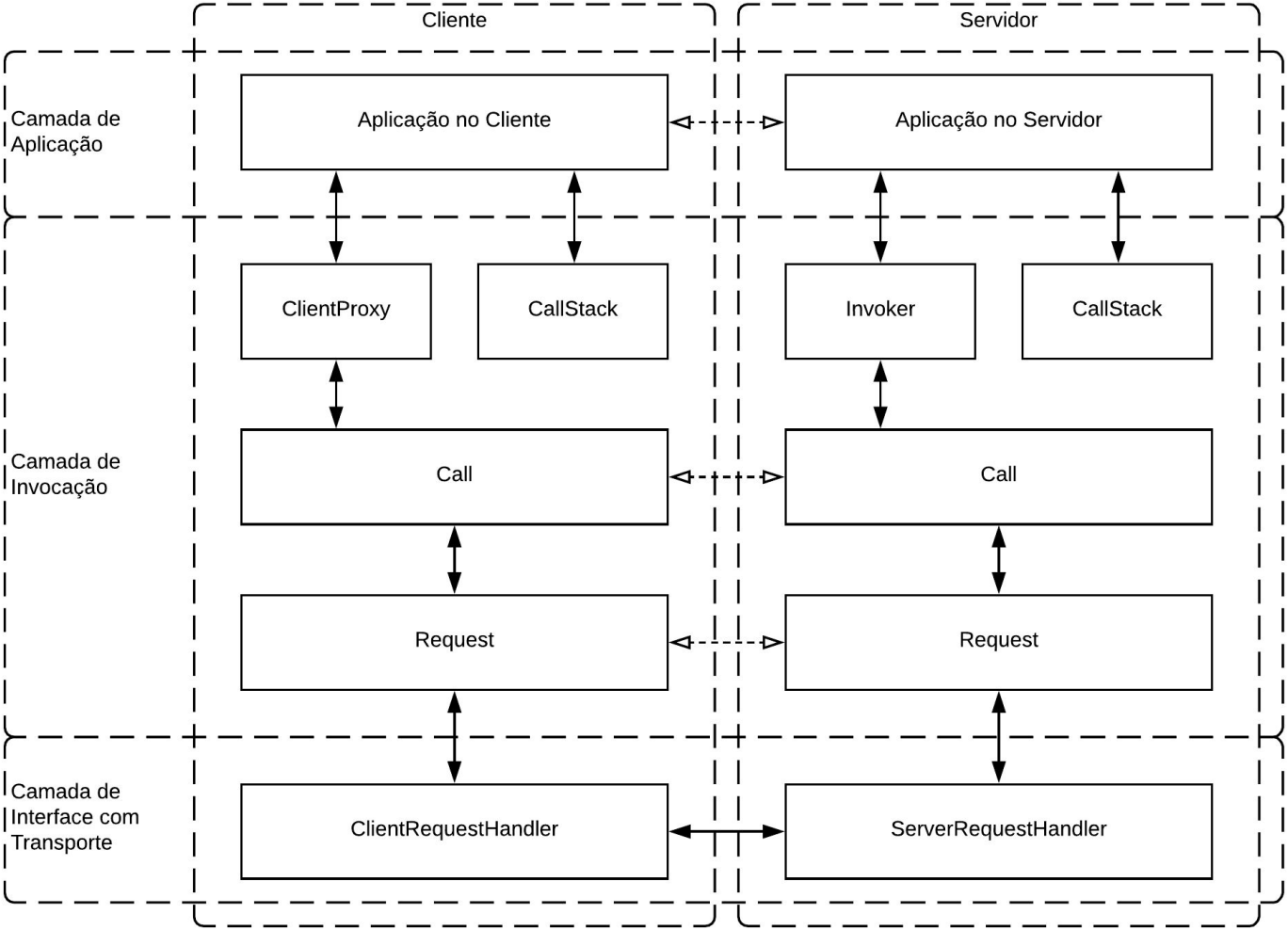
ID	Descrição
~~RF01~~	~~Permitir a chamada remota de procedimentos.~~
RF02	Fornecer interface de métodos remotos para o cliente.
~~RF03~~	~~Serializar dados (tipos primitivos, vetores, arquivos).~~
~~RF04~~	~~Criptografar dados.~~
~~RF05~~	~~Comprimir dados.~~
~~RF06~~	~~Conexões simultâneas.~~

Requisitos Não-Funcionais

ID	Descrição
RNF01	Implementação em C++.
RNF02	Utilizar criptografia de chave simétrica.

Requisitos Não-Funcionais

ID	Descrição
~~RNF01~~	~~Implementação em C++.~~
~~RNF02~~	~~Utilizar criptografia de chave simétrica.~~


Cliente
Servidor
Camada de Aplicação
Aplicação no Cliente
Aplicação no Servidor
ClientProxy
CallStack
Invoker
CallStack
Camada de Invocação
Call
Call
Request
Request
Camada de Interface com Transporte
ClientRequestHandler
ServerRequestHandler

Implementação

- Transporte utilizando TCP.
- Biblioteca pthreads para gerenciar múltiplas conexões.
- Utilizamos o algoritmo Blowfish para criptografar os dados.
- Biblioteca ZLib para compressão de dados.
- Arquivos binários foram serializados utilizando codificação em base64.
- Biblioteca ImageMagick para manipular imagens (prog. de demonstração).

DEMO

Avaliação de Desempenho

- O cliente envia uma imagem para o método remoto e espera que ela seja retornada em tons de cinza.

- Ele afere o tempo do envio até o recebimento da imagem em tons de cinza, precisão de nanosegundos. São realizadas 10.000 invocações.

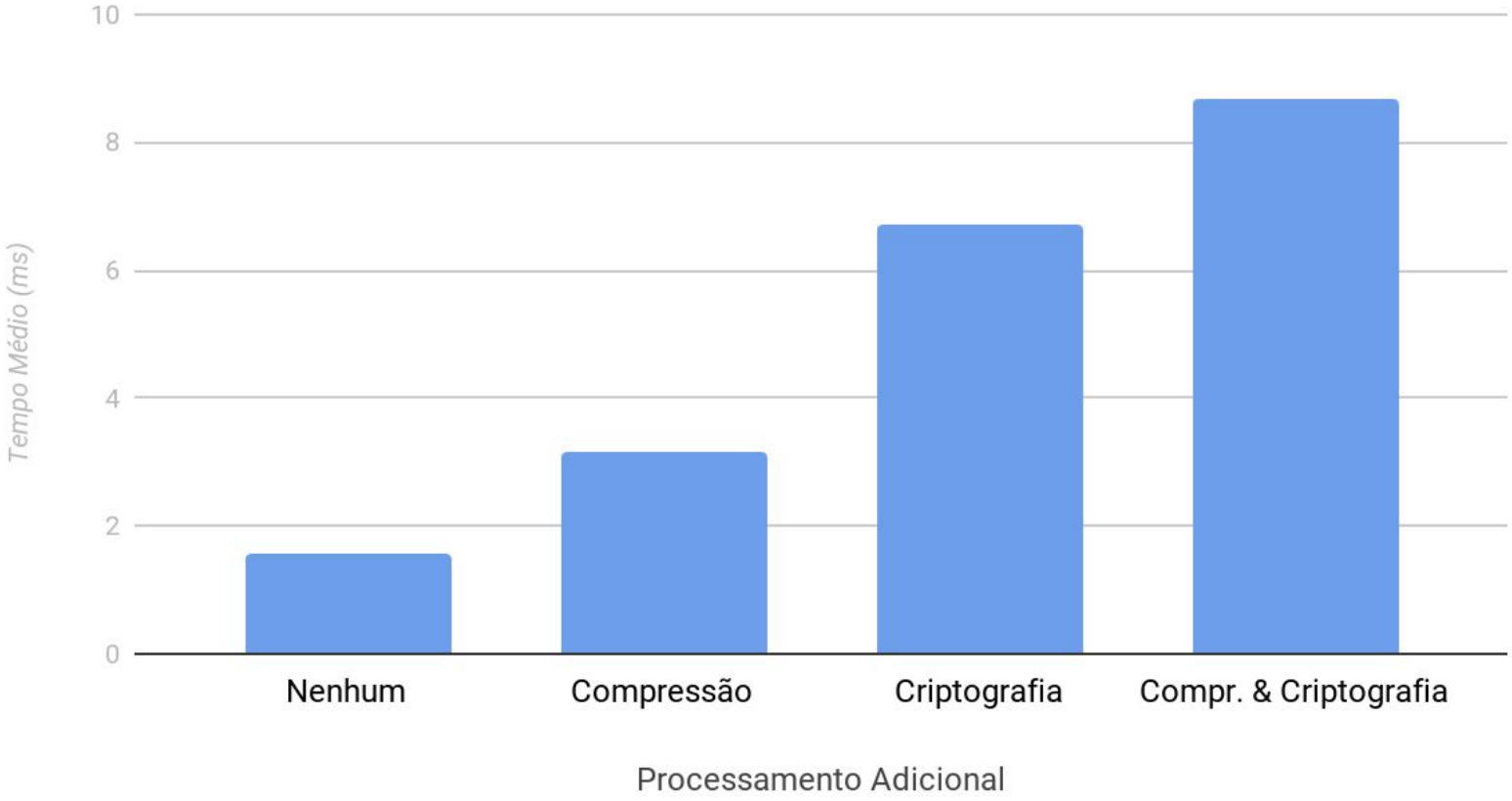
- Avaliamos os tempos para chamadas com dados criptografados e comprimidos, apenas criptografados, apenas comprimidos e os originais.

- Comparação com o middleware XML-RPC.

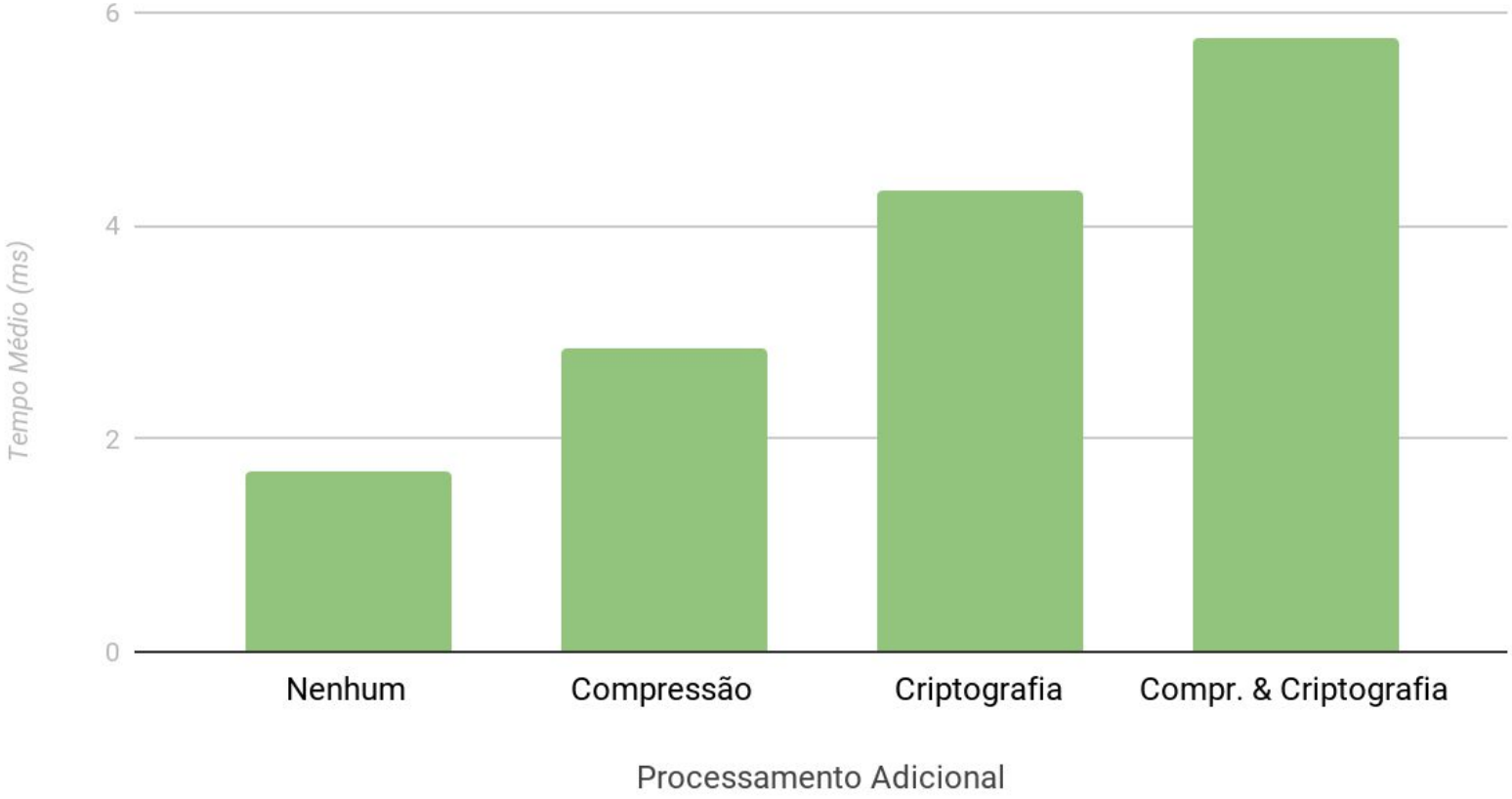
Ambiente de Testes

- Mesmo notebook.
- Conectado ao cabo de alimentação.
- Bateria carregada.
- Wi-Fi desligado.
- Sem programas desnecessários rodando em segundo plano.

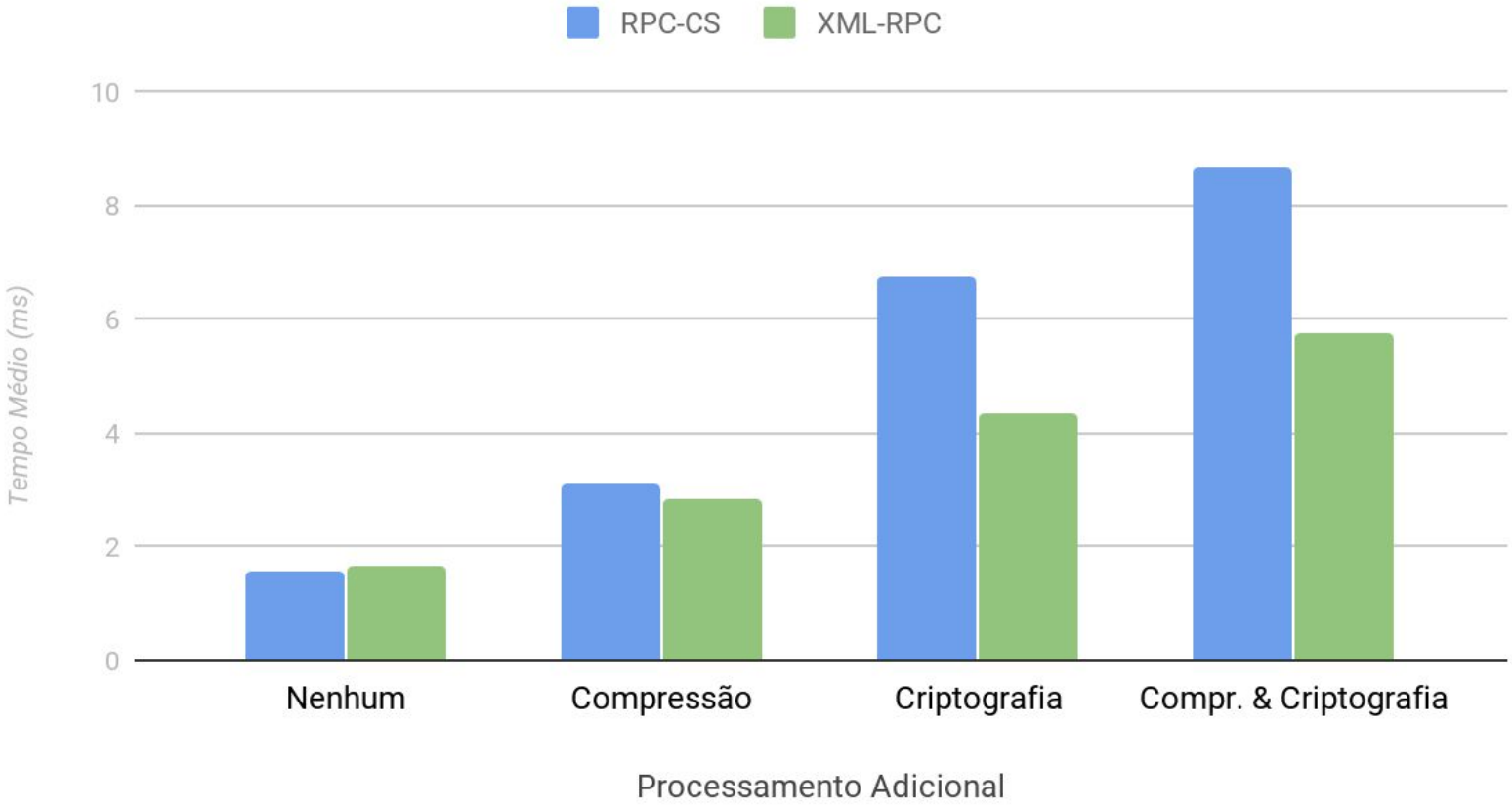
RPC-CS


10
8
6
4
2
0
Tempo Médio (ms)
Nenhum
Compressão
Criptografia
Compr. & Criptografia
Processamento Adicional

XML-RPC


6
4
2
0
Tempo Médio (ms)
Nenhum
Compressão
Criptografia
Compr. & Criptografia
Processamento Adicional

RPC-CS vs. XML-RPC



Middleware / Processamento Adicional	RPC-CS	XML-RPC
Nenhum	1553.26 ± 391.23 µs	1681.48 ± 393.89 µs
Compressão	3143.48 ± 1137.33 µs	2835.70 ± 890.00 µs
Criptografia	6726.14 ± 1140.73 µs	4340.81 ± 769.93 µs
Compressão & Criptografia	8660.32 ± 2810.78 µs	5747.69 ± 1150.89 µs

Conclusão

- Nosso middleware demorou mais tempo que o XML-RPC para realizar as invocações. Provavelmente, devido a grande quantidade de métodos alocando memória dinamicamente.

- Acreditamos que a nossa API ficou mais simples e intuitiva do que a implementada pelo XML-RPC.

- Ter planejado o middleware em termos de camadas poderia ter facilitado no desenvolvimento e na divisão de tarefas.

FIM