Auto-avaliação do Projeto de RCI

Base de dados em anel

2021/2022, 2º semestre, P3

Grupo nº: 104

Nome: Fábio Dias Número: 96201

Nome: Tiago Ferreira Número: 96334

Instruções e notas

1. A aplicação deve ser compilada com o comando make invocado na diretoria onde o ficheiro com o projeto é descomprimido; a compilação com gcc e as opções -Wall não deverá apresentar quaisquer erros ou avisos.
2. Todas as chamadas de sistema que dizem respeito à comunicação devem ser testadas contra situações de erro nas suas execuções.
3. A terminação abrupta do programa resultando em segmentation fault é fortemente penalizada; devem assegurar-se que o programa só acede a posições de memória devidamente alocadas.
4. Não obstante os programas serem testados no LT5, tenham o Wireshark em funcionamento nas vossas máquinas.

Auto-avaliação

Questões	**Sim**	**Não**
A aplicação estabelece um anel com dois nós, comandos **new** e **pentry**?	X	
A aplicação estabelece um anel com mais de dois nós?	X	
A aplicação preserva um anel durante uma sequência arbitrária de entradas e saídas de nós, comandos, **new**, **pentry** e **leave**?	X	
Sem atalhos, a pesquisa de uma chave está correta, comando **find**?	X	
Com atalhos, a pesquisa de uma chave está correta?	X	
A entrada de um nó no anel conhecendo apenas um qualquer outro nó está correta, comando **bentry**?	X	
A leitura de dados nos sockets TCP tem em conta que não há garantia de correspondência biunívoca entre mensagens da aplicação e invocações da chamada de sistema read()?		X
A aplicação usa temporizadores para lidar com a potencial perda de mensagens enviadas por UDP?	X	

Indique sucintamente alguma funcionalidade extra que tenha desenvolvido, para além do especificado no enunciado do projeto; ou algum esclarecimento que queira partilhar.

Para além de reenviar as mensagens caso o Ack se perda, garantimos que, ao fim de x vezes a tentar reenviar por UDP, a mensagem segue pelo seu sucessor (por TCP) de forma a garantir que todas as mensagens são enviadas.
Para evitar perdas de FIND's caso haja um anel muito grande ou com atrasos, implementámos uma lista para guardar em memória as chaves e o respetivo identificador para ser possível confirmar os FIND's mesmo que um chegue primeiro. Isto permite ao utilizador fazer vários FIND's localmente. A gestão de memória, alocação e limpeza da lista é também garantida.
De forma semelhante, implementámos um array para guardar a informação dos nós que querem entrar pelo BENTRY de forma a permitir ao utilizador fazer vários BENTRY's onde o nó de início da procura é o mesmo.
Para além disso, garantimos que os comandos são válidos com a sua versão por extenso (por exemplo, "c" pode ser dado como chord ...).