Universidade de Brasília
Departamento de Ciência da Computação

27 de maio de 2015
(2015/1)

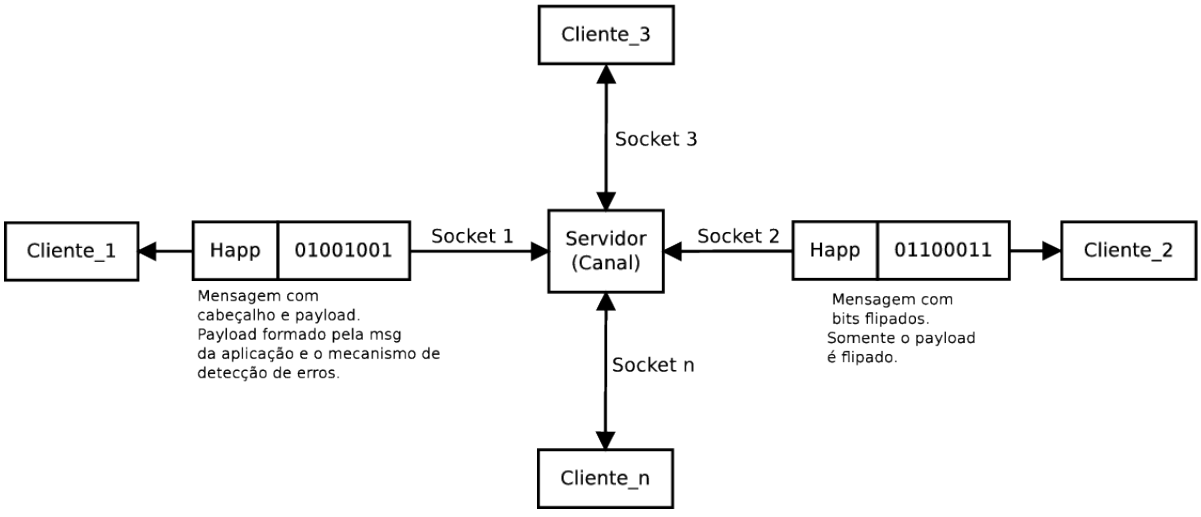
Teleinformática e Redes 1 – Turma A – Trabalho 1
Professor: Marcos F. Caetano <caetano@cic.unb.br>
Monitor: Lucas Saad <lucasaad@gmail.com>
Resumo: Implementação de técnicas de detecção de erros.

1 Descrição do Projeto

Este trabalho tem como objetivo simular a comunicação entre dois ou mais clientes, utilizando um canal de comunicação não confiável como meio compartilhado. Para a implementação desta abstração, deverá ser utilizado a arquitetura *Cliente* × *Servidor*. O canal não confiável será implementado como servidor e a conexão entre os clientes deverá, **obrigatoriamente**, passar pelo Canal de Comunicação. Ou seja, após o estabelecimento da comunicação entre os clientes e o servidor (canal de comunicação), toda a mensagem encaminhada por um cliente origem deverá ser recebida, tratada e encaminhadas aos clientes de destino pelo servidor.

É importante destacar que o canal de comunicação também desempenha o papel de encaminhador de mensagens entre os clientes de origem e os de destinos. Para isso, deverá ser implementado, na camada aplicação, um cabeçalho para a identificação dos endereços de origem e destino. Com base nestes identificadores é que o canal de comunicação terá condições de fazer o encaminhamento entre os *sockets* de comunicação abertos.

A figura abaixo apresenta uma representação da arquitetura descrita.



Todas as configurações do Canal e dos clientes deverão ser fornecidas mediante arquivo de configuração. Desta forma, espera-se ser possível obter uma maior flexibilização nos parâmetros fornecidos ao sistema durante a execução de diversos experimentos.

1.1 Aplicação

A aplicação a ser implementada deverá gerar, de forma sequencial, números inteiros positivos. Este números serão apresentados tanto no *cliente origem* quanto no *cliente destino*. O *payload* das mensagens geradas pela Camada de Aplicação é formado pelos números gerados de forma sequencial.

1.2 Cliente

Os clientes deverão conectar-se ao servidor (canal) mediante ao estabelecimento de conexão *socket* utilizando os protocolos TCP e IP. O *socket* estabelecido é que será utilizado para encaminhar as mensagens ao Servidor. Deverá ser implementado um cabeçalho para as mensagens geradas pela Camada de Aplicação. Este cabeçalho será utilizado para registrar as seguintes informações:

- Endereço da aplicação de origem;
- Endereço da aplicação de destino;
- Algoritmo de correção aplicado ao *payload*.

Com base nos cabeçalhos das mensagens, o Servidor terá condições de realizar o encaminhamento das mensagens recebidas para o *socket* de saída correto. É importante deixar registrado que durante o processo de estabelecimento do *socket*, o cliente deverá fazer o registro do seu endereço junto ao Servidor. Somente com este registro é que o servidor terá condições de fazer o mapeamento: $socket_n \rightarrow cliente_n$;

O *cliente origem* utilizará o *socket* da conexão estabelecida com o servidor para o encaminhamento de suas mensagens. Contudo, antes do encaminhamento, é necessário a aplicação do algoritmo de correção definido no arquivo de configuração. A mensagem recebida pela Camada de Aplicação (número inteiro positivo) terá o seu valor convertido para um binário de 32 *bits* e somente este valor será utilizado como entrada em um dos algoritmos definidos (CRC-8, Hamming, MD5 ou SHA1). A saída resultante será anexada ao cabeçalho da mensagem de aplicação, devidamente preenchido, e encaminhado pelo *socket*.

O *cliente destino* receberá do *socket* estabelecido com o servidor a mensagem encaminhada pelo *cliente origem*. O cabeçalho da mensagem fornecerá a informação de qual algoritmo de correção foi aplicado. O *cliente destino* deverá verificar se a mensagem foi recebida de maneira correta. Caso contrário, deverá tentar recuperar a informação original.

1.3 Servidor

O servidor de comunicação emula o comportamento de um *canal de comunicação* **não confiável**. Sendo assim, desempenha dois papeis: a propagação das mensagens e a simulação de erros de comunicação.

Para realizar a propagação das mensagens, o servidor deverá estar apto a estabelecer conexões TCP/IP e fazer o registro dos clientes conectados. Com base no cabeçalho das mensagens recebidas, o servidor fará o encaminhamento das mensagens para o *socket* que representam a conexão com o *clientes destino* apropriado.

Contudo, antes de fazer o encaminhamento, o servidor deverá simular os erros de comunicação. Os erros possíveis estão definidos no arquivo de configuração de entrada. São eles: *flipar* bits pares, *flipar* bits ímpares, *flipar* de forma aleatória de 1 a todos os bits do *payload* da mensagem, não flipar nenhum bit; **Somente os bits do payload da mensagem é que poderão ser flipados**.

2 Algoritmos de Detecção

Para este trabalho, deverão ser utilizados os seguintes algoritmos: CRC-8, Hamming, MD5 e SHA1. Os algoritmos CRC-8 e Hamming deverão, **obrigatoriamente**, ser implementados. Para os algoritmos MD5 e SHA1 podem ser utilizadas bibliotecas quem possuam a sua implementação.

3 Relatório

Um relatório final do projeto deve ser apresentado. O trabalho deve conter:

- Explicação teórica do funcionamento dos algoritmos de detecção e correção de erros utilizados;
- Comparação entre os resultados obtidos utilizando os algoritmos: CRC-8, Hamming, MD5 e SHA1 sob os cenários: *flipar* bits pares, *flipar* bits ímpares, *flipar* de forma aleatória de 1 a x% os bits do *payload* da mensagem, não flipar nenhum bit;
  - taxa de erro detectados;
- Apresentar conclusão mostrando quais vantagens e desvantagens de cada algoritmo e qual seria o melhor algoritmo em um determinado contexto;

4 Avaliação

A avaliação consiste em 3 etapas:

- Código e funcionamento do projeto.
  - O código e o relatório deverão ser enviados para o email: *lucasaad@gmail.com* até o dia **21/6 as 23:55h**;
  - A apresentação do trabalho deverá ser agendada pelo email: *lucasaad@gmail.com*. As apresentações acontecerão no período de **22/6 a 26/6** e **29/6 a 3/7**. As apresentações serão feitas fora do horário de aula;
  - Não serão aceitos trabalho enviados fora do prazo;
  - Trabalhos não apresentados também não serão considerados.
- Nota do Relatório.
- Uma prova sobre o trabalho.

5 Observações

As seguintes observações deverão ser consideradas pelos alunos que forem fazer o trabalho:

- O trabalho deve rodar **OBRIGATORIAMENTE** na plataforma GNU/Linux;
- O programa pode ser feito em qualquer linguagem;
- No relatório do trabalho deve conter as instruções para compilação do código e a relação de todas as bibliotecas utilizadas;
- Códigos copiados serão considerados "cola"e todos os alunos envolvidos ganharão nota zero;
- Dúvidas sobre o trabalho deverão ser tiradas **utilizando a lista de email da disciplina**. Desta forma, dúvidas comuns e esclarecimentos poderão ser respondidos uma única vez para toda a turma.