Definição do Trabalho Final

Implementar as técnicas de controle de fluxo **Stop-and-wait** e **Go-back-n** em uma aplicação utilizando a API de socket(7).

1) Objetivos

a) Compreensão de duas técnicas de controle de fluxo utilizadas na camada de enlace (i.e., *Stop-and-wait* e *Go-Back-n*);
b) Compreensão do algoritmo de CRC;
c) Compreensão da API socket – amplamente utilizada para comunicação em redes; e
d) Compreensão das dificuldades em aplicações que se comunicam em diferentes arquiteturas (i.e., *endianness*, *padding*, *deadlock*).

2) Funcionalidade

Deverá ser implementada uma aplicação que envia um arquivo de uma máquina para outra. O arquivo deverá ser salvo na máquina que o recebe. Além do usuário informar o arquivo a ser transferido, ele também escolherá a técnica de controle de fluxo a ser utilizada e o tamanho da janela para essa transferência.

O arquivo recebido deve ser o mesmo que o enviado. Para confirmação da igualdade deve-se usar um programa externo que verifique a integridade do arquivo (por exemplo, md5sum para Linux/OS X e hashcalc para Windows).

A rede que suportará a simulação é uma rede com taxas ínfimas de erro, de forma que os pacotes gerados pela aplicação não serão espontaneamente recebidos incorretamente. Assim sendo, deve existir um módulo de inserção de falhas que force a falha no código de verificação.

3) Especificações técnicas

a) Arquitetura

A aplicação utiliza um modelo cliente-servidor. As informações de arquivo, técnica de controle de fluxo, janela de transmissão e inserção de falhas são informadas ao cliente.

b) Protocolo de comunicação

O servidor é inicializado e espera o início de uma conexão.

O cliente é inicializado recebendo os parâmetros de execução. Após a inicialização, o cliente envia um pedido de conexão ao servidor. O servidor verifica se os parâmetros estão dentro dos intervalos (definidos na seção de limitações) e, caso afirmativo, aceita a conexão com o envio de um pacote do tipo ACK; caso o pacote não esteja dentro dos parâmetros, o pacote é descartado.

Estabelecida a conexão, o cliente, então, passa a enviar pacotes de dados de acordo com a janela de transmissão disponível e espera pacotes do tipo ACK do servidor. Caso necessário, o cliente poderá reenviar pacotes de dados. O servidor recebe os pacotes e confirma o recebimento daqueles que não possuírem erro de acordo com a técnica de controle de fluxo. Esse processo é mantido até o final da

transmissão. O final da transmissão ocorre assim que a quantidade de dados for igual àquela informada no estabelecimento da conexão.

Visto que os últimos pacotes de dados podem ser recebidos com erro, o cliente somente finaliza a sua execução após receber o ACK para o pacote de dados que seguiria o último; ou seja, o número de sequência do último pacote de dados incrementado em uma unidade. O servidor finaliza sua execução após o envio deste ACK.

c) Pacote base

O formato de pacote que será utilizado nesse trabalho é definido a seguir:

0	4	8	9	10	12	16	254
Máquina Destino	Máquina Origem	Tipo	Sequência	Tamanho	CRC	Dados (veja descrição)	

Máquina Destino (4 bytes): IP da máquina destino em formato inteiro (veja limitações a seguir).

Máquina Origem (4 bytes): IP da máquina origem em formato inteiro.

Tipo (1 byte): Tipo de pacote

→ 00 para pacote de dados.
→ 01 para Acknowledge (ACK).
→ 10 para estabelecimento de conexão.

O tipo do pacote determina o significado do campo "Dados".

(i) Para pacote de dados, é a sequência de bytes correspondente aos dados da aplicação;

(ii) Para pacotes do tipo Acknowledge, essa área não existe; e

(iii) Para pacotes do tipo estabelecimento de conexão, a área de dados conterá um segundo pacote que estabelece os parâmetros da execução. Ele será definido a seguir.

Sequência (1 byte): número de sequência sendo ou enviado ou confirmado.

Tamanho (2 bytes): tamanho da área de dados.

CRC (4 bytes): controle de erro (definido a seguir).

Dados (240 bytes): definição depende do contexto. Veja o parâmetro tipo.

d) Pacote de estabelecimento de conexão

O pacote de estabelecimento de conexão é definido a seguir:

0	4	5	8
Técnica do controle de fluxo	Tamanho da Janela	Tamanho do arquivo	

Técnica de controle de fluxo (4 bytes):

- ➔ 0 para Stop-and-wait
- ➔ 1 para Go-back-n

Tamanho da janela (1 byte): tamanho da janela de transmissão.

Tamanho do arquivo (4 bytes): tamanho do arquivo que será enviado.

e) Inserção de falhas

O usuário pode escolher o uso de inserção de falhas no cliente. As falhas são do tipo *bitflips* apenas no campo "Tamanho" e "Dados" e apenas para pacotes de dados. Não serão necessários erros em outros tipos de pacotes (simplificação do trabalho – não é necessário implementar mecanismos de *timeout*).

f) CRC

Para a verificação de erros o algoritmo de CRC deverá ser implementado, e o polinômio gerador que será utilizado é o seguinte: $x^8 + x^6 + x^3 + x + 1$.

4) Limitações

a) Sugere-se o uso de UDP (i.e., datagrama) como protocolo de transporte. O protocolo UDP possui uma interface mais simples que TCP (i.e., stream).

b) O estabelecimento de conexão deste trabalho é simplificado; normalmente, o estabelecimento requer o uso de três pacotes (por exemplo, TCP *handshake*).

c) Existem campos com bits sem uso; veja, por exemplo, o campo "Técnica de controle de fluxo". Isso é comum em protocolos para permitir futuras extensões. Tais bits sem uso são ditos reservados (*reserved)*.

d) O tamanho da janela não ultrapassará o valor 16; portanto, o tamanho de janela está restrito ao intervalo [1, 16].

e) Deve ser usada a representação em inteiro do IP da máquina (ou seja, ***não utilize*** strings). Sugere-se o uso de inet_pton(3) para conversão de IP do formato string para inteiro. Permite-se o uso do struct in_addr no lugar de inteiro (ambos possuem o mesmo tamanho).

f) É permitido que o servidor fique em deadlock caso (i) nenhum pacote seja enviado ou (ii) o pacote não seja recebido. No caso do cliente, é permitido que ocorra deadlock caso o pacote não seja recebido. Reitera-se que a chance de um pacote ser perdido é mínima, especialmente em uma rede local. Idealmente, deveria ser utilizado uma política de *timeout* para o tratamento dessas situações; porém, não é necessário nesse trabalho visto a complexidade envolvida (por exemplo, poderia ser utilizando alarm(3) ou select(7) para o tratamento dessas situações).

g) É permitido o uso de código existente que implemente CRC desde que seja utilizado o polinômio definido nesse trabalho.

h) Não é necessário lidar com a situação de múltiplos clientes enviando dados ao servidor ao mesmo tempo.

i) O formato do pacote é propositadamente desalinhado; o intuito é simular outros protocolos que foram desenvolvidos assim. A aplicação deve tratar corretamente essa situação; por exemplo, o tamanho do pacote de estabelecimento de conexão é 12 bytes e não 15 bytes.

j) Caso as regras desse trabalho sejam seguidas, é possível diferentes implementações desse trabalho conversarem entre si. Além disso, é possível utilizar esse trabalho para transmissão de dados via internet. Neste último caso, é necessário o uso de *port forwarding*. Nenhum dos dois cenários discutos será cobrado neste trabalho (infelizmente).

5) Regras gerais

Grupos: individualmente ou grupos de até três componentes.

Data da entrega: **21/11/2019**

Entrega final no Moodle:

- Código fonte comentado.
- Se forem realizadas modificações da especificação desse documento, deve-se criar um pequeno texto explicando os motivos da alteração.

Apresentação: todos participantes devem estar presentes.

Visualização de resultados:

- Demonstração deverá acontecer em 2 máquinas;
- A chegada de um pacote ao destino deve ser mostrada;
- Deve existir um *log* para poder-se visualizar o que está acontecendo durante o envio de uma mensagem na rede;
- Deve ser possível identificar a técnica de controle de fluxo que está sendo utilizada durante a transmissão;
- Visualizar se a mensagem chegou corretamente ou não chegou; e
- Verificar o arquivo recebido.

IMPORTANTE: Não serão aceitos trabalhos entregues fora do prazo. Trabalhos que não compilam ou que não executam não serão avaliados. Todos os trabalhos serão analisados e comparados. Caso seja identificada cópia de trabalhos, todos os trabalhos envolvidos receberão nota ZERO.