

Socket Programming

Ingrid Santana Lopes
14/0083065

Departamento de Ciência da Computação,
Universidade de Brasília
Brasília, Brasil

Marcos Paulo Cayres Rosa
14/0027131

Departamento de Ciência da Computação,
Universidade de Brasília
Brasília, Brasil

I. METODOLOGIA

Nessa sessão, está explicada a forma como cada requisição foi implementada sobre o protocolo UDP de exemplo fornecido em linguagem Python.

A. Entrega confiável e ordem dos dados

A entrega confiável e a ordem dos dados foi implementada de forma que: dada uma mensagem que o cliente deseja enviar, o checksum da mesma será calculado. Quando a mensagem é enviada ao servidor juntamente com o resultado desse checksum, o checksum é recalculado do lado do servidor apenas para o trecho correspondente à mensagem original para fins de comparação. Caso os dois checksums, o recebido pelo servidor e o calculado novamente com base na mensagem recebida, estiverem iguais, então não houve corrupção de dados. Existe a possibilidade de duas mensagens diferentes terem o mesmo checksum, mas a probabilidade para isso acontecer depois de uma dada mensagem ter sido corrompida é muito pequena.

Quanto a ordem, tanto o cliente quanto o servidor contém um contador que é incrementado para cada mensagem enviada ou recebida respectivamente. O cliente envia a mensagem junto com sua soma de bits e a sequência da mesma. Ao chegar no servidor, ele separa tais informações e checa se o número de sequência enviado pelo cliente confere com o número de sequência que esperaria receber. Caso não seja, ele detecta como estando fora de ordem e informa isso ao cliente para que ele possa então enviar o pacote correto de novo e continuar enviando os pacotes seguintes a partir disso. Tal situação de fora de ordem acontece quando um pacote se perde ao ser enviado do cliente ao servidor.

B. Controle de Fluxo e Pipeline

O Pipeline implementado foi o Go N Back, o cliente envia mensagens contínuas levando em conta sempre o tamanho da janela e a movimentando quando recebe um pacote de confirmação (ACK) do receptor e, no caso do exemplo, a mensagem em caps lock. A ideia de controle de fluxo no Go N Back está intimamente ligada à janela deslizante que representa um valor de pacotes que o servidor consegue suportar dado o buffer do mesmo.

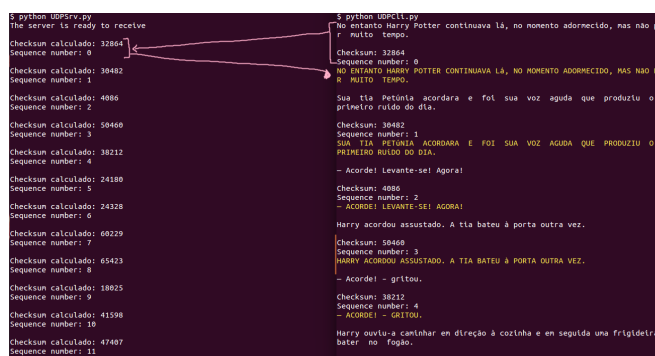
No caso de um erro em que algum pacote se perca em qualquer um dos sentidos, a janela não irá se movimentar e, assim, o pacote que se perdeu será reenviado e todos aqueles depois dele também para que o servidor possa obtê-los na

ordem certa e então mandar a mensagem alterada de volta para o cliente também na ordem certa. Ou seja, se qualquer pacote for perdido ou danificado, então esse pacote e todos os pacotes a seguir na janela (mesmo se eles foram recebidos sem erro) vão ser re-enviados. Dessa forma, o pequeno trecho onde a ordem ficou errada, seria desconsiderado. Como tanto o cliente, como o servidor, notam quando um pacote falta, a mensagem de erro é indicada no terminal.

II. EXEMPLOS DE EXECUÇÃO

Foram feitas execuções fornecendo exatamente 26 mensagens no formato de strings de tamanhos variados retiradas do capítulo 2 de Harry Potter e a Pedra Filosofal. Duas dessas execuções foram registradas para serem mostradas de exemplos neste relatório.

A simulação já funciona usando o pipeline Go n Back. No início, não ocorre primeiramente nenhum erro, como pode ser observado na figura 1. Seguindo as setas presentes na figura rosa, pode-se ver como uma mensagem gerada pelo cliente é recebida pelo servidor que então manda a mensagem alterada de volta que é recebida com sucesso. Como até essa etapa não ocorre nenhum erro ou problema, quando a informação chega no servidor, ela rapidamente é mandada em sua nova forma ao cliente.



```
$ python UDPsrv.py
The server is ready to receive
Checksum calculado: 32864
Sequence number: 0
Checksum calculado: 30482
Sequence number: 1
Checksum calculado: 4086
Sequence number: 2
Checksum calculado: 50408
Sequence number: 3
Checksum calculado: 38212
Sequence number: 4
Checksum calculado: 24380
Sequence number: 5
Checksum calculado: 24328
Sequence number: 6
Checksum calculado: 60229
Sequence number: 7
Checksum calculado: 65423
Sequence number: 8
Checksum calculado: 18625
Sequence number: 9
Checksum calculado: 41598
Sequence number: 10
Checksum calculado: 47487
Sequence number: 11

$ python UDPlit.py
No entanto Harry Potter continuava lá, no momento adormecido, mas não por
nulo tempo.
Checksum: 32864
Sequence number: 0
NO ENTANTO HARRY POTTER CONTINUAVA LÁ, NO MOMENTO ADORMECIDO, MAS NÃO POR
NULO TEMPO.
Sua tia Petúnia acordara e fez sua voz aguda que produziu o
primeiro ruído do dia.
Checksum: 30482
Sequence number: 1
SUA TIA PETÚNIA ACORDARA E FEZ SUA VOZ AGUDA QUE PRODUZIU O
PRIMEIRO RUÍDO DO DIA.
- Acorde! Levante-se! Agora!
Checksum: 4086
Sequence number: 2
- ACORDE! LEVANTE-SE! AGORA!
Harry acordou assustado. A tia bateu a porta outra vez.
Checksum: 50408
Sequence number: 3
HARRY ACORDOU ASSUSTADO. A TIA BATEU A PORTA OUTRA VEZ.
- Acorde! - gritou.
Checksum: 38212
Sequence number: 4
- ACORDE! - GRITOU.
Harry ouviu-a caminhar em direção à cozinha e em seguida uma frigideira
bater no fogão.
```

Figura 1. Início da simulação

Depois de algumas trocas de mensagens, ocorre um erro de corrupção na sequência 17. Ao notar isso, o servidor não manda a mensagem alterada de volta para o cliente pois, para ele, a mensagem está errada. Graças a isso, as mensagens depois dessa serão indicadas pelo servidor como estando fora de

Diagram illustrating a sequence of messages between a user and a chatbot, showing the chatbot's responses and the user's responses.

Chatbot's Responses (Red Boxes):

- Checksum calculado: 47797
Sequence number: 17
- error detected - CORRUPTION**
- Checksum calculado: 64587
Sequence number: 17
- error detected - OUT OF ORDER**
- Checksum calculado: 8610
Sequence number: 17
- error detected - CORRUPTION**
- Checksum calculado: 63437
Sequence number: 17
- error detected - OUT OF ORDER**
- Checksum calculado: 12094
Sequence number: 17
- error detected - OUT OF ORDER**
- Checksum calculado: 9877
Sequence number: 17
- error detected - OUT OF ORDER**
- Checksum calculado: 46178
Sequence number: 17
- error detected - OUT OF ORDER**
- Checksum calculado: 14961
Sequence number: 17

User's Responses (Green Boxes):

- Nada, nada...
- Checksum: 14961
Sequence number: 17
- Simulating bit error**
O aniversário de Duda =
- Checksum: 64587
Sequence number: 18
O ANIVERSÁRIO DE DUDA =
como podia ter esquecido?
- Checksum: 57619
Sequence number: 19
- Simulating bit error**
Harry levantou-se devagar e começou a procurar as metais.
- Checksum: 64337
Sequence number: 20
HARRY LEVANTOU-SE DEVAGAR E COMEÇOU A PROCURAR AS METAIS.
- Encontrou-as debaixo da cama e depois de retirar uma aranha de um pé, ficou as.
- Checksum: 12094
Sequence number: 21
ENCONTROU-AS DEBAIXO DA CAMA E DEPOIS DE RETIRAR UMA ARANHA DE UM PÉ, FICOU AS.
- Harry estava acostumado com aranhas, porque o armário sob a cama vivia cheio delas e era ali que ele dormia.
- Checksum: 9877
Sequence number: 22
HARRY ESTAVA ACOSTUMADO COM ARANHAS, PORQUE O ARMÁRIO SOB

O erro de corrupção da mensagem 17 só vai ser corrigido depois que, após vários envios de mensagens, o cliente notar que tem algo errado pois não irá conseguir movimentar a janela na qual o Go N Back se baseia para seu funcionamento. Dessa forma, como pode ser observado na figura 3, a mensagem de número de sequência 17 será enviada novamente e o servidor poderá retorná-la em letras maiúsculas. Apesar da mensagem de sequência 18 não ter chegado corrompida como pode ser notado na figura 2, ela será enviada novamente pois foi considerada fora de ordem. Todas as mensagens a partir da 17 serão reenviadas para que a sequência possa finalmente ficar em ordem novamente. A mensagem "erro 17" indica o número da sequência na qual o erro foi identificado e então corrigido.

Quando a sequência de 26 mensagens termina, o cliente termina de receber as mensagens que faltavam e então a conexão é encerrada e o servidor mostra quando tempo, em segundos, ele levou para executar tal simulação como pode ser visto na figura 5.

Somente depois, quando o cliente tenta andar com a janela e nota que faltou o ack do pacote 11, que ele reenvia indicando a mensagem de erro com a numeração do pacote e então o

Checksum calculado: 9077
Sequence number: 17
error detected - OUT OF ORDER

Checksum calculado: 40178
Sequence number: 17
error detected - OUT OF ORDER

Checksum calculado: 14961
Sequence number: 17

Checksum calculado: 64587
Sequence number: 18

Checksum calculado: 57619
Sequence number: 19

Checksum calculado: 63437
Sequence number: 20

Checksum calculado: 12094
Sequence number: 21

Checksum calculado: 9077
Sequence number: 22

Checksum calculado: 46178
Sequence number: 23

Checksum calculado: 33012
Sequence number: 24

Checksum calculado: 50762
Sequence number: 25

Checksum calculado: 55335
Sequence number: 26

Checksum calculado: 65535

Checksum: 9077
Sequence number: 22
HARRY ESTAVA ACOSTUMADO COM ARANHAS, PORQUE O ARMARIO SOB A ES
CADA VIZIA CHEIO DEILAS E BRA ALI QUE ELE DORMIA.

Já vestido saiu para o corredor que levava à cozinha.

Checksum: 40178
Sequence number: 23
Já VESTIDO SAIU PARA O CORREDOR QUE LEVAVA À COZINHA.

ERRO: 17
- NADA, NADA...

O ANIVERSARIO DE DUDA -

A mesa quase desaparecerá tantos eram os presentes de aniversá
rio de Duda.

Checksum: 33012
Sequence number: 24
COMO PODIA TER ESQUECIDO?

Pelo que via, Duda ganhara o novo computador que queria, para não fal
ar na segunda televisão e na bicicleta de corrida.

Checksum: 50762
Sequence number: 25
HARRY LEVANTOU-SE DEVAGAR E COMEÇOU A PROCURAR AS MEIAS.

Checksum: 55335
Sequence number: 26
Simulating acknowledgement loss

HARRY ESTAVA ACOSTUMADO COM ARANHAS, PORQUE O ARMARIO SOB A ES

Checksum calculado: 40178 Sequence number: 23	HARRY ESTAVA ACOSTUMADO COM AMANHOS, PORQUE O AMANHÃO SÓM A EU CADA VIEZA CHEGO DELAS E ERA ALÍ QUE ELE DORMIA.
Checksum calculado: 33012 Sequence number: 24	JÁ VESTIDO SATU PARA O CORREDOR QUE LEVAVA A COZINHA.
Checksum calculado: 50762 Sequence number: 25	A RESA QUASE DESAPARECERA TANTAS ERAM OS PRESENTES DE ANIVERSÁRIO DO DODA.
Checksum calculado: 65535 Sequence number: 26	PELO QUE VEM, DODA GANHARA O NOVO COMPUTADOR QUE QUERIA, PARA NBO FALAR NA CORDA TELEVISÃO E NA BICICLETA DE CORRIDA.
FILE TRANSFER SUCCESSFUL THE TAKEN 7-31918515015	Connection closed

```
Checksum calculado: 47407
Sequence number: 11

Simulating packet loss
Checksum: 24328
Sequence number: 6
VIROU-SE DE COSTA E TENTOU SE LEMBRAR DO SONHO EM QUE ESTAVA.
Era um sonho gostoso.

Error detected - OUT OF ORDER
Checksum: 60223
Sequence number: 7

Checksum calculado: 25253
Sequence number: 11

Error detected - OUT OF ORDER
Checksum: 45151
Sequence number: 11

Error detected - OUT OF ORDER
Checksum calculado: 2146
Sequence number: 11

Error detected - OUT OF ORDER
Checksum: 10025
Sequence number: 9
Tinha a estranha sensação que já vira esse sonho antes.

Checksum calculado: 34507
Sequence number: 11

Error detected - OUT OF ORDER
Checksum calculado: 14961
Sequence number: 11

Error detected - OUT OF ORDER
Checksum calculado: 47407
Sequence number: 11

Simulating bit error
VIROU-SE DE COSTA E TENTOU SE LEMBRAR DO SONHO EM QUE ESTAVA.
Checksum: 24328
Sequence number: 6
VIROU-SE DE COSTA E TENTOU SE LEMBRAR DO SONHO EM QUE ESTAVA.
Era um sonho gostoso.

Checksum: 60223
Sequence number: 7

Simulating bit error
Havia uma motocicleta.
Checksum: 65423
Sequence number: 8
Havia uma motocicleta.

Tinha a estranha sensação que já vira esse sonho antes.

Checksum: 10025
Sequence number: 9
Tinha a estranha sensação que já vira esse sonho antes.

A tia voltara a porta.

Checksum: 41598
Sequence number: 10
A TIA VOLTARA A PORTA.

- Você já se levantou? - perguntou.

Checksum: 47407
Sequence number: 11
- Você já se levantou? - PERGUNTOU.
```

pacote 11 e todos os demais posteriores a ele são reenviados para o servidor, o qual envia a mensagem modificada que é recebida então pelo cliente.

```
checksum calculado: 2146
sequence number: 11
error detected - OUT OF ORDER
checksum calculado: 34587
sequence number: 11
error detected - OUT OF ORDER
checksum calculado: 64587
sequence number: 18
checksum calculado: 49461
```

```
OK: 11 - VOCA 11 SE LEVANTOU! - PERGUNTOU.
- BEM, ANDE DEPRESSA, QUERO QUE VOCE TOME CONTA DO BACON. E NAO SE ATREVA A DEIXAR O QUEIMAR.
O aniversário de Duda -
Checksum: 64587
Sequence number: 18
QUERO TUDO PERFEITO NO ARMARIO NO ANIVERSARIO DE DUDA.
```

Em relação a implementação desses erros, foram randomizadas valores que são comparados a uma probabilidade para decidir se irá corromper dados do pacote (no código do cliente antes de enviar o pacote para o servidor) ou perder o pacote (no código do servidor antes de passar os dados para a camada superior). Para detecção de erro, o cliente tenta receber os acks do servidor por um certo período de tempo especificado e, caso haja uma discrepância entre o momento atual e o do

último ack que tenha movido a janela, será visto como um erro e assim reenviada a janela completa. Para o servidor, quando não se recebe mensagem por um certo período e já foi recebida a última mensagem, padronizada como um texto vazio, ele termina seu serviço.

III. INFORMAÇÕES DE DESEMPENHO

Como a simulação de perdas de pacote e corrupção baseia-se na geração de números randômicos que podem ou não estar dentro de uma faixa de probabilidade que determina se isso vai acontecer, a ocorrência dos mesmos se baseia nessa probabilidade. Para a nossa implementação, randomizamos valores entre zero e um e caso fossem menores do que 0.1, aplicávamos a lógica para corromper os dados ou perder pacotes, dependendo do caso.

Para detecção de erro, depende do tempo especificado como a diferença entre o atual e o último ack com número de sequência novo recebido. Assim, quando a diferença é maior que o limiar estabelecido, o cliente reenviará a janela. No caso, colocamos essa diferença como 0.01, testado a cada tempo limite de 0.001. Colocamos esses valores a partir de testes realizados para não reenviar pacotes excessivamente e permitir que o erro fosse detectado pelo cliente em um período que o servidor não ficasse com demasiada captação de erro.