
Homework 3

Cedric Bone

Mar 21, 2025

CONTENTS:

1	cedric_bone_hw3	3
1.1	main module	3
1.2	network_simulator module	3
1.3	packet module	3
1.4	receiver module	4
1.5	sender module	4
	Python Module Index	7
	Index	9

Add your content using `reStructuredText` syntax. See the [reStructuredText](#) documentation for details.

CEDRIC_BONE_HW3

1.1 main module

`main.main()`

Main function

1.2 network_simulator module

Network Simulator

Implements a network simulator that can drop and corrupt packets.

class `network_simulator.NetworkSimulator`(*sender_port=11111, receiver_port=22222, loss_rate=0.3, corruption_rate=0.3*)

Bases: `object`

Simulates an unreliable network channel.

loss_rate

Probability of packet loss

Type

float

corruption_rate

Probability of packet corruption

Type

float

run()

Main sim loop.

Receives packets and randomly drops or corrupts them before forwarding.

1.3 packet module

Packet

Implements a packet with sequence number, data, and checksum.

class `packet.Packet`(*seq_num, data, ack_num=None, checksum=None*)

Bases: `object`

A packet with sequence number, data, and checksum.

seq_num
Sequence number
Type
int

data
Data
Type
str

ack_num
Ack number
Type
int

checksum
Checksum
Type
int

calculate_checksum()
Calculate checksum (UDP style)

1.4 receiver module

Receiver

Implements the receiver side of the reliable data transfer protocol.

class receiver.**Receiver**(*port=22222, window_size=4*)

Bases: object

A receiver that receives packets and reassembles

receive_file()

Receive file from sender

send_ack(*ack_num, addr*)

Send an ACK

Parameters

- **ack_num** (*int*) – ACK number
- **addr** (*tuple*) – Address of sender

1.5 sender module

Sender

Implements the sender side of the reliable data transfer protocol.

class sender.**Sender**(*port=11111, window_size=4*)

Bases: object

Implements a sliding window retransmission.

seq_num

Current sequence number

Type

int

window_size

Size of the sliding window

Type

int

window

Buffer

Type

dict

base

Sequence number of the oldest unacknowledged packet

Type

int

send_file(*data*, *receiver_addr*=('localhost', 33333))

Send a file to address

Parameters

- **data** (*str*) – Data to send
- **receiver_addr** (*tuple*) – Destination address (host, port)

send_packet(*packet*, *addr*)

Send a packet to address

Parameters

- **packet** (*Packet*) – Packet to send
- **addr** (*tuple*) – Destination address (host, port)

PYTHON MODULE INDEX

m

main, 3

n

network_simulator, 3

p

packet, 3

r

receiver, 4

s

sender, 4

INDEX

A

`ack_num` (*packet.Packet* attribute), 4

B

`base` (*sender.Sender* attribute), 5

C

`calculate_checksum()` (*packet.Packet* method), 4

`checksum` (*packet.Packet* attribute), 4

`corruption_rate` (*network_simulator.NetworkSimulator* attribute), 3

D

`data` (*packet.Packet* attribute), 4

L

`loss_rate` (*network_simulator.NetworkSimulator* attribute), 3

M

`main`

 module, 3

`main()` (in module *main*), 3

`module`

main, 3

network_simulator, 3

packet, 3

receiver, 4

sender, 4

N

`network_simulator`

 module, 3

NetworkSimulator (class in *network_simulator*), 3

P

`packet`

 module, 3

Packet (class in *packet*), 3

R

`receive_file()` (*receiver.Receiver* method), 4

`receiver`

 module, 4

Receiver (class in *receiver*), 4

`run()` (*network_simulator.NetworkSimulator* method), 3

S

`send_ack()` (*receiver.Receiver* method), 4

`send_file()` (*sender.Sender* method), 5

`send_packet()` (*sender.Sender* method), 5

`sender`

 module, 4

Sender (class in *sender*), 4

`seq_num` (*packet.Packet* attribute), 3

`seq_num` (*sender.Sender* attribute), 4

W

`window` (*sender.Sender* attribute), 5

`window_size` (*sender.Sender* attribute), 5