Bryan Arnold

CSE 3300

12/9/18

Programming Assignment 4

Exercise 0:

Here is an example output of the exercise 0 of the homework assignment:

```
Please choose a request type. 0 or 1: 0

Please enter the Social Security Number you wish to find the PO Box for: 111111111

Attempting to send message to server (3300, 1031, 2291298003, 111111111, 34120, 0)

Successfully received message from server: (19684, 1031, 2291298003, 111111111, 16502,
1234)

PO Box number: 1234
```

Code:

```
import socket
import random
import struct
import urllib.request  # the lib that handles the url stuff

#############################
# Bryan Arnold              #
# CSE 3300                  #
# 12/9/18                   #
# Programming Assignment 4  #
#############################

# Get the IP address of the machine making
# the client request to the server.
host_ip_address = urllib.request.urlopen('https://ident.me').read().decode('utf8')

# Default IP address and port of the server
# to make requests to. Also initialize UDP socket.
host_IP = socket.gethostbyname("tao.ite.uconn.edu")
host_port = 3300
UDPSock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
timeout = 5.0

# Create the initial states for the packages to be
# sent over the socket to the server
head = 3300
lab_and_version_number = 1031
client_cookie = random.randint(5000, pow(2, 32))
```

```
ssn_request = 111111111
other = 0

# Compute the sum of two 16 bit integers
# that are received/sent to the server
def get_sum(unsigned16_1, unsigned16_2):

    result = unsigned16_1 + unsigned16_2

    if result > 0x0000ffff:

        result = (result & 0x0000ffff) + 1

    return result


# Calculate the checksum for the message being sent
# by looking at each line 1-4 each. Add each line up together,
# then add all up together to generate a checksum.
def get_checksum(unsigned16_1, unsigned16_2, unsigned32_1, unsigned32_2,
unsigned16_3):

    temp_sum = get_sum(unsigned16_1, unsigned16_2)

    temp = unsigned32_1 >> 16
    temp_sum = get_sum(temp_sum, temp)
    temp = unsigned32_1 & 0x0000ffff
    temp_sum = get_sum(temp_sum, temp)

    temp = unsigned32_2 >> 16
    temp_sum = get_sum(temp_sum, temp)
    temp = unsigned32_2 & 0x0000ffff
    temp_sum = get_sum(temp_sum, temp)

    temp_sum = get_sum(temp_sum, unsigned16_3)
    result = ~temp_sum & 0x0000ffff

    return result

# Ask for user input on which type of request they want to make.
# 0 = client to tao.ite.uconn.edu, 1 = server set up by me
while 1:

    request_type = input('Please choose a request type. 0 or 1: ')

    if request_type == '' or request_type == '0':
        break
    elif request_type == '1':
        head = 36068
        break

    # If the user wants to do type 0 request, ask for Social
    # Security number
if head != 36068:

    ssn = -1

    while 1:

        try:

            ssn = int(input('Please enter the Social Security Number you wish to find
the PO Box for: '))
```

```
            if ssn == '':
                break
            elif 0 < int(ssn) < 1000000000:
                ssn_request = int(ssn)
                break

        except ValueError:
            pass

# If the user wishes to make a request to the server created
host_address = 0
if head == 36068:

    # Default initializers. Includes server port, address generation, and
    # ip of sender.
    other = 3200
    server_ip = host_ip_address
    server_ip = server_ip.split('.')
    part_a, part_b, part_c, part_d = server_ip

    # Try to set up server address to send to
    try:
        part_a = int(part_a) << 24
        part_b = int(part_b) << 16
        part_c = int(part_c) << 8
        part_d = int(part_d)
        host_address = part_a | part_b | part_c | part_d
    except ValueError:
        pass

if head == 36068:

    ssn_request = host_address

# Get checksum for checking
checksum = get_checksum(head, lab_and_version_number, client_cookie, ssn_request,
other)

# Variables set up to display and read incoming and outgoing
# messages.
data = None
unpacked_msg = None
packed_msg = struct.pack('!2H2I2H', head, lab_and_version_number, client_cookie,
ssn_request, checksum, other)

print("Attempting to send message to server", struct.unpack('!2H2I2H', packed_msg))

# A lot cap for retransmitting if timeout occurs
max_transmission = 5
retransmission_counter = 1
while max_transmission > 0:

    UDPSock.sendto(packed_msg, (host_IP, host_port))

    # Timeout checker
    try:
        UDPSock.settimeout(timeout)

        data, client_address = UDPSock.recvfrom(20)
        if data:
            print("Successfully received message from server:",
struct.unpack('!2H2I2H', data))
```

```
            unpacked_msg = struct.unpack('!2H2I2H', data)
            if unpacked_msg[4] == get_checksum(unpacked_msg[0], unpacked_msg[1],
unpacked_msg[2], unpacked_msg[3],
                                               unpacked_msg[5]):
                break
            else:
                continue

    # Attempt retransmit
    except socket.timeout:

        print('The request to the server timed out, attempting to retransmit') +
str(retransmission_counter)
        retransmission_counter += 1
        pass

    max_transmission -= 1

# Error checking. If no errors are found, print out the PO box
# corresponding to the requested SSN
if unpacked_msg is not None:

    if unpacked_msg[5] & 0x8000 != 0:

        if unpacked_msg[5] == 32769:
            print('An error occurred while checking the checksum. Please try again.')

        elif unpacked_msg[5] == 32770:
            print('A syntax error occurred.')

        elif unpacked_msg[5] == 32772:
            print('An error occurred. Please wait and try again.')

        elif unpacked_msg[5] == 32776:
            print('An error occurred in the server. Please wait and try again.')

        else:
            print('An error occurred. Please wait and try again.')

    elif unpacked_msg[0] == 19684:
        print('PO Box number: ' + str(unpacked_msg[5]))

else:
    print('No response from Server. Please wait and try again.')
```

<u>Exercise 1:</u>

Here is an example scenario of output from exercise 1 of the assignment:

Server start up:

Server is now listening on port 3200...

Making a request on the client:

```
Please choose a request type. 0 or 1: 1

Attempting to send message to server (36068, 1031, 2927090342, 2304987637, 65052, 3200)

Successfully received message from server: (52452, 1031, 2927090342, 2304987637, 19068,
32800)

Successfully got message from server.
```

Server Status after client request received:

```
Server is now listening on port 3200...

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3917795402, 975387379, 49198, 0)

Checksum match.

Requested Social Security Number was found in the database: 2120

The following message will now be sent:  (19684, 1031, 3917795402, 975387379, 30694, 2120)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 981631251, 402608790, 51433, 0)

Checksum match.

Requested Social Security Number was found in the database: 6290

The following message will now be sent:  (19684, 1031, 981631251, 402608790, 28759, 6290)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3398373136, 325453877, 63961, 0)

Checksum match.

Requested Social Security Number was found in the database: 5113

The following message will now be sent:  (19684, 1031, 3398373136, 325453877, 42464, 5113)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3398373136, 3410624789, 18709, 0)

Checksum match.

Requested Social Security was not found in the database.
```

```
The following message will now be sent:  (19684, 1031, 3398373136, 3410624789, 35088,
32772)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3398373136, 3498008159, 59029, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 3398373136, 3498008159, 9873,
32772)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3398373136, 1211206153, 17210, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 3398373136, 1211206153, 33589,
32772)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 1031, 3398373136, 1211206153, 17210, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 3398373136, 1211206153, 33589,
32772)

Type[0] messages received from ('137.99.3.212', 51992)

Content: (3300, 25607, 3398373136, 1211206153, 58169, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 3398373136, 1211206153, 33591,
32770)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 3180773954, 975387379, 64036, 0)

Checksum match.

Requested Social Security Number was found in the database: 2120
```

```
The following message will now be sent:  (19684, 1031, 3180773954, 975387379, 45532, 2120)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 593345054, 225234552, 13748, 0)

Checksum match.

Requested Social Security Number was found in the database: 4198

The following message will now be sent:  (19684, 1031, 593345054, 225234552, 58701, 4198)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 2828978013, 360332591, 11375, 0)

Checksum match.

Requested Social Security Number was found in the database: 7928

The following message will now be sent:  (19684, 1031, 2828978013, 360332591, 52598, 7928)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 2828978013, 2079249275, 12206, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 2828978013, 2079249275, 28585,
32772)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 2828978013, 1690687825, 16641, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 2828978013, 1690687825, 33020,
32772)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 2828978013, 3321708129, 31417, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 2828978013, 3321708129, 47796,
32772)
```

```
Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 1031, 2828978013, 3321708129, 31417, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 2828978013, 3321708129, 47796,
32772)

Type[0] messages received from ('137.99.3.212', 39052)

Content: (3300, 29447, 2828978013, 3321708129, 3001, 0)

Checksum match.

Requested Social Security was not found in the database.

The following message will now be sent:  (19684, 1031, 2828978013, 3321708129, 47798,
32770)
```

Code:

```python
import socket
import struct
import urllib.request

#############################
# Bryan Arnold              #
# CSE 3300                  #
# 12/9/18                   #
# Programming Assignment 4  #
#############################

# Get local IP address as well as the information within the database of Social
# Security Numbers to PO Boxes
host_ip_address = urllib.request.urlopen('https://ident.me').read().decode('utf8')
dataBase =
urllib.request.urlopen('http://engr.uconn.edu/~song/classes/cn/db').read().decode('utf
8')
dataBase = dataBase.split('\n')
ssn_dataBase = dataBase

# Initialize socket, head, local server port number, etc.
UDPSock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
head = 19684
lab_and_version_number = 1031
info_field = 0
host_port = 3200
UDPSock.bind(('', host_port))

# Check to see if the social security number is in fact
# within the database.
def get_ssn(value):
    result = 32772

    if value > 99999999:

        for line in ssn_dataBase:

            if line.find(str(value)) != -1:  # Find Post Office Box number

                result = int(line[len(line) - 4:])
                print('Requested Social Security Number was found in the database: %s'
% result)
                break

    if result == 32772:

        print('Requested Social Security was not found in the database.')

    return result


# Compute the sum of two 16 bit integers
# that are received/sent to the server
def get_sum(unsigned16_1, unsigned16_2):

    result = unsigned16_1 + unsigned16_2

    if result > 0x0000ffff:

        result = (result & 0x0000ffff) + 1

    return result

# Calculate the checksum for the message being sent
# by looking at each line 1-4 each. Add each line up together,
# then add all up together to generate a checksum.
def get_checksum(unsigned16_1, unsigned16_2, unsigned32_1, unsigned32_2,
unsigned16_3):

    temp_sum = get_sum(unsigned16_1, unsigned16_2)

    temp = unsigned32_1 >> 16
    temp_sum = get_sum(temp_sum, temp)
    temp = unsigned32_1 & 0x0000ffff
    temp_sum = get_sum(temp_sum, temp)

    temp = unsigned32_2 >> 16
    temp_sum = get_sum(temp_sum, temp)
    temp = unsigned32_2 & 0x0000ffff
    temp_sum = get_sum(temp_sum, temp)

    temp_sum = get_sum(temp_sum, unsigned16_3)

    result = ~temp_sum & 0x0000ffff
    return result


# Get the information received in the request message from the client,
# unpack and sort it, then craft another message that will be passed onto
# the server as a new message.
def generate_msg(unsigned16_1, unsigned16_2, unsigned32_1, unsigned32_2,
unsigned16_3):
    temp = get_checksum(unsigned16_1, unsigned16_2, unsigned32_1, unsigned32_2,
unsigned16_3)
    result = struct.pack('!2H2I2H', unsigned16_1, unsigned16_2, unsigned32_1,
unsigned32_2, temp, unsigned16_3)

    return result

# Checksum > Syntax Error > Unknown SSN > Server Error
print('Server is now listening on port %s...' % host_port)

while 1:

    # Wait for incoming messages, and receive if one comes in
    data, client_address = UDPSock.recvfrom(20)
    unpacked_msg = struct.unpack('!2H2I2H', data)
    msg_type = 1

    #If the messages coming in was of type 0, this is S
    if unpacked_msg[0] == 3300:
        msg_type = 0

    print('Type[%s] messages received from %s\nContent: %s' % (msg_type,
client_address, unpacked_msg))

    # Check checksum of message and make sure it matches
    checksum = get_checksum(unpacked_msg[0], unpacked_msg[1], unpacked_msg[2],
unpacked_msg[3], unpacked_msg[5])
    if checksum != unpacked_msg[4]:
        print('Error, the checksum does not match. An error occurred.')
        data = generate_msg(head, lab_and_version_number, unpacked_msg[2],
unpacked_msg[3], 32769)

    #Checksum is good
    else:

        print('Checksum match.')

        #Unpack the message contents and send off a message to the server regarding
the message recevied
        request_ssn = get_ssn(unpacked_msg[3])
        if unpacked_msg[0] != 3300 or unpacked_msg[1] != 1031:
            data = generate_msg(head, lab_and_version_number, unpacked_msg[2],
unpacked_msg[3], 32770)
        elif request_ssn > 0:
            data = generate_msg(head, lab_and_version_number, unpacked_msg[2],
unpacked_msg[3], request_ssn)
        else:
            data = generate_msg(head, lab_and_version_number, unpacked_msg[2],
unpacked_msg[3], 32776)

    #Send off message
    print('The following message will now be sent: ', struct.unpack('!2H2I2H', data))
    UDPSock.sendto(data, client_address)
```