Jason Jaroszewicz

Dr. John Zhao

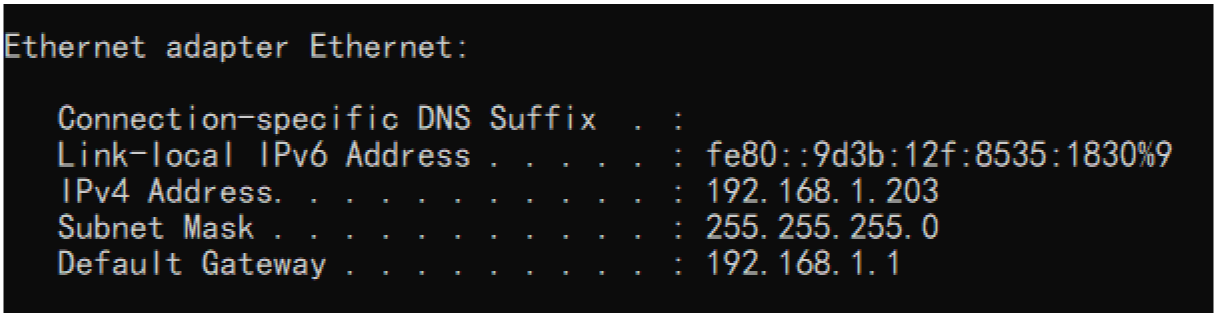
CS-GY 6843: Computer Networking, Section INET

11-22-2023

Socket Prog.3: ICMP Pinger

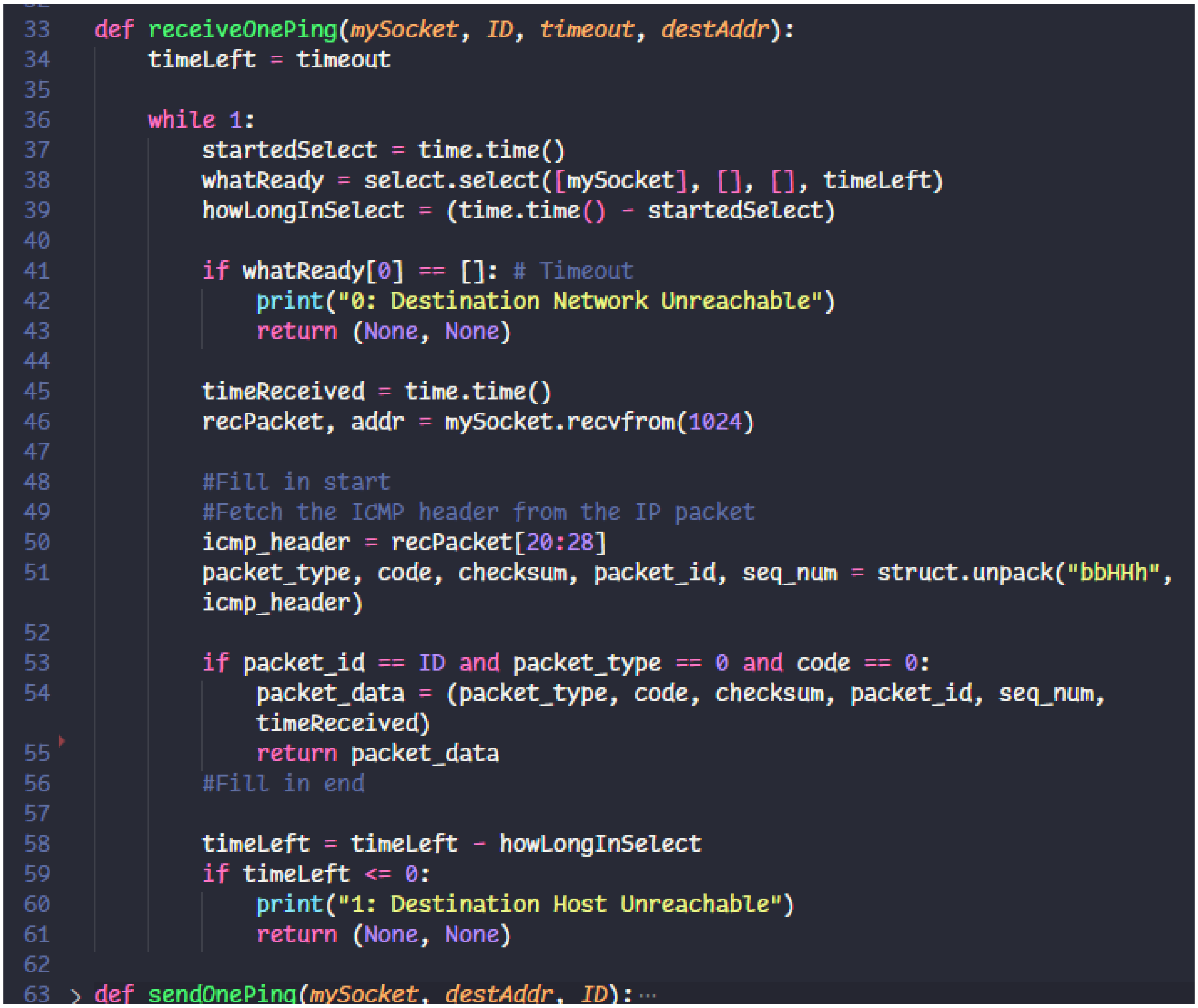
Lab Environment Details

Windows 10 *ipconfig* results:



```
Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::9d3b:12f:8535:1830%9
   IPv4 Address. . . . . . . . . . . : 192.168.1.203
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1
```



```
33   def receiveOnePing(mySocket, ID, timeout, destAddr):
34       timeLeft = timeout
35
36       while 1:
37           startedSelect = time.time()
38           whatReady = select.select([mySocket], [], [], timeLeft)
39           howLongInSelect = (time.time() - startedSelect)
40
41           if whatReady[0] == []: # Timeout
42               print("0: Destination Network Unreachable")
43               return (None, None)
44
45           timeReceived = time.time()
46           recPacket, addr = mySocket.recvfrom(1024)
47
48           #Fill in start
49           #Fetch the ICMP header from the IP packet
50           icmp_header = recPacket[20:28]
51           packet_type, code, checksum, packet_id, seq_num = struct.unpack("bbHHh",
             icmp_header)
52
53           if packet_id == ID and packet_type == 0 and code == 0:
54               packet_data = (packet_type, code, checksum, packet_id, seq_num,
                 timeReceived)
55               return packet_data
56           #Fill in end
57
58           timeLeft = timeLeft - howLongInSelect
59           if timeLeft <= 0:
60               print("1: Destination Host Unreachable")
61               return (None, None)
62
63 > def sendOnePing(mySocket, destAddr, ID):···
```

In the **receiveOnePing()** function, at line 50 through line 61 the received packet is deconstructed to be returned. The packet header is placed into `icmp_header` and then a try block is used to attempt to unpack the header into several variables: `packet_type`, `code`, `checksum`, `packet_id`, and `seq_num`.

`packet_type` is the ICMP type. This will commonly be 8 or 0 depending on if the packet is an echo or echo reply.

`code` is the subtype to the given ICMP type.

`checksum` is error checking data calculated from the header and data.

`packet_id` is the packet ID returned in an echo reply.

`seq_num` is the sequence value returned in an echo reply.

If `packet_id` equals the ID parameter and the packet type and code are as expected, the response tuple will be constructed as a tuple containing the previously mentioned variables and the time at which the packet was received.