Chapter 9

Internet Control Message Protocol (ICMP)
Outline

- TYPES OF MESSAGES
- MESSAGE FORMAT
- ERROR REPORTING
- QUERY
- CHECKSUM
- ICMP PACKAGE
Introduction

- IP provides unreliable and connectionless datagram delivery
- Drawbacks
  - Lack of error control mechanism
  - Lack of assistance mechanism
- Solution
  - ICMP
Lack of Error Control Mechanism

- No error-reporting or error-correcting mechanism
  - What happens if a router must discard a datagram because
    - Cannot find a router to the destination
    - Time-to-live field has a zero value
  - What happens if the final destination host must discard all fragments of a datagram
    - Because it has not received all fragments within a predefined time limit
Lack of Assistance Mechanism

- Lack of a mechanism for host and management queries
  - How to determine if a router or another host is alive?
  - How to obtain information from another host or router?
Solution

- ICMP: Internet Control Message Protocol
  - A network layer protocol
  - However, its message are not passed directly to the data link layer
  - The message are first encapsulated inside IP datagram before going to the lower layer
Position of ICMP in the Network Layer
Encapsulation of ICMP Packet
9.1 TYPES OF MESSAGES
ICMP Messages

- ICMP messages are divided into
  - Error-reporting message
    - Report problems that a router or a host (destination) may encounter when it processes an IP packet
  - Query message
    - Help a router or a network manager to get specific information from a router or another host
ICMP Messages

- Error-reporting
- Query
## ICMP Messages

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error-reporting</td>
<td>3</td>
<td>Destination unreachable</td>
</tr>
<tr>
<td>message</td>
<td>4</td>
<td>Source quench</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Time exceeded</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Parameter problem</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Redirection</td>
</tr>
<tr>
<td>Query messages</td>
<td>8 or 0</td>
<td>Echo request to reply</td>
</tr>
<tr>
<td></td>
<td>13 or 14</td>
<td>Timestamp request or reply</td>
</tr>
<tr>
<td></td>
<td>17 or 18</td>
<td>Address mask request or reply</td>
</tr>
<tr>
<td></td>
<td>10 or 9</td>
<td>Router solicitation or advertisement</td>
</tr>
</tbody>
</table>
MESSAGE FORMAT

9.2
General Format of ICMP Messages

---

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
<th>Rest of the header</th>
<th>Data section</th>
</tr>
</thead>
</table>

8 bits 8 bits 8 bits 8 bits
Format of ICMP Message

- 8-byte header
  - The first 4-byte are common to all
    - Type(1-byte): define the type of the message
    - Code(1-byte): specify the reason for the particular message type
    - Checksum(2-byte)
  - The rest is specific for each message type

- A variable-size data section
  - For error message
    - Carries information for finding the original packet that had the error
  - For query message
    - Carries extra information based on the type of the query
ERROR REPORTING
Error Reporting

- ICMP only report error
  - Does not correct error
  - Error correction is left to the higher-level protocol

- Error message are always sent to the original source
  - Because the only information available in the datagram is the source and destination IP address
ICMP always reports error messages to the original source.
Error-Reporting Messages

- Destination unreachable
- Source quench
- Time exceeded
- Parameter problems
- Redirection
Important Points about ICMP Error Messages

- No ICMP error message for a datagram carrying an ICMP error message
  - Prevent the problem of infinite loop
- No ICMP error message for a fragmented datagram that is not the first fragment
- No ICMP error message for a datagram having a multicast address.
- No ICMP error message for a datagram with a special address such as 127.0.0.0 or 0.0.0.0.
ICMP Packet Data Section

- The data section in all error message contain includes
  - The IP header of the original datagram
    - Give the original source information about the datagram itself
  - The first 8-byte of data in that datagram
    - Provides information about the port number (UDP and TCP) and sequence number (TCP)
    - Source then can inform the upper layer protocols (TCP or UDP) about the error
Contents of Data Field for Error Messages

Received datagram

IP header  8 bytes  IP data

ICMP header

IP header  8 bytes  ICMP packet

IP header  ICMP header  IP header  8 bytes  Sent IP datagram

Figure 9-6
Destination Unreachable

- When a router cannot route a datagram or a host cannot deliver a datagram
  - Discard the datagram
  - The router or host sends a destination unreachable message back to the source host
## Destination-Unreachable Format

<table>
<thead>
<tr>
<th>Type: 3</th>
<th>Code: 0 to 15</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Unused (All 0s)

Part of the received IP datagram including IP header plus the first 8 bytes of datagram data
Code Field for Destination-Unreachable

- **Code 0**: the network is unreachable
  - Possibly due to hardware failure
  - Can only be generated by a router
- **Code 1**: the host is unreachable
  - Possibly due to hardware failure
  - Can only be generated by a router
- **Code 2**: the protocol is unreachable
  - Delivery to the upper layer protocol (TCP, UDP) is failed
  - Can only be generated by a destination host
Code Field for Destination-Unreachable (Cont.)

- Code 3: the port is unreachable
  - Can only be generated by the destination host

- Code 4: fragmentation is required, but the DF (do not fragment) field has been sent
  - The sender specifies no fragmentation
  - But the router is impossible without fragmentation

- Code 5: source routing cannot be accomplished
  - One or more routers defined in the source routing cannot be visited
Code Field for Destination-Unreachable (Cont.)

- **Code 6**: the destination network is unknown
  - In code 0: the router knows that the destination network exists, but it is unreachable at the moment
  - In code 6: the router has no information about the destination network

- **Code 7**: the destination host is unknown
  - In code 1: the router knows that the destination host exists, but it is unreachable at the moment
  - In code 7: the router is unaware of the existence of the destination host
Code Field for Destination-Unreachable (Cont.)

- Code 8: the source host is isolated
- Code 9: communication with the destination network is administratively prohibited
- Code 10: communication with the destination host is administratively prohibited
- Code 11: the network is unreachable for the specified type of service
- Code 12: the host is unreachable for the specified type of service
Code 13: the host is unreachable because the administrator has put a filer on it

Code 14: the host is unreachable because the host precedence is violated
- The requested precedence is not permitted for the destination

Code 15: the host is unreachable because its precedence was cut off
Note

Destination-unreachable messages with codes 2 or 3 can be created only by the destination host.

Other destination-unreachable messages can be created only by routers.
Error Reporting

- Even if a router does not report a destination-unreachable message
  - Does not mean that the datagram has been delivered
  - For example, in a Ethernet network, there is no way that a router knows a packet has been delivered to the destination or the next router
    - Ethernet does not provide an acknowledge mechanism
A router cannot detect all problems that prevent the delivery of a packet.
Source Quench

- In IP, there is no communication between the source host, the router, and the destination.
- As a result, no flow control in IP
  - The source never knows if it is producing datagram faster than can be
    - Forwarded by router
    - Processed by the destination host
- Problem
  - Congestion in routers or the destination host
Source Quench (Cont.)

- Source-quench message thus adds a kind of flow control to the IP
  - Inform the source that the datagram has been discarded
  - Warm the source that
    - There is congestion somewhere in the path
    - The source should slow down (quench) the sending process
Source-Quench Format

<table>
<thead>
<tr>
<th>Type: 4</th>
<th>Code: 0</th>
<th>Checksum</th>
</tr>
</thead>
</table>

Unused (All 0s)

Part of the received IP datagram including IP header
plus the first 8 bytes of datagram data
Note about the Source Quench

- One source-quench message should be sent for each datagram that is discarded due to congestion.

- There is no mechanism to tell the source that the congestion has been relieved and the source can speed up its sending rate.
  - The source continues to slow down that rate until no more source-quench messages are received.
The congestion can be created either by one-to-one or many-to-one communication

- In one-to-one: source quench is helpful
- In many-to-one: may be useless
  - Each source sends datagram at a different rate
  - The router or the destination has no clue which source is responsible for the congestion
  - Thus, it may drop a datagram from a very slow source
Time Exceeded: Two Situations

- The packet travel in a loop or a cycle
  - Caused by errors in the routing table
  - Finally, time-to-live value is 0
  - The router discards the datagram and sends time-exceeded message

- When all fragment that make up a message do not arrive at the destination within a certain time limit
  - When the first datagram arrives at the destination, it starts a timer
  - When the timer expires and all the fragments are not arrived
  - The destination discards all the fragments and sends a time-exceeded message
Whenever a router receives a datagram with a time-to-live value of zero, it discards the datagram and sends a time-exceeded message to the original source.
When the final destination does not receive all of the fragments in a set time, it discards the received fragments and sends a time-exceeded message to the original source.
Time-Exceeded Message Format

<table>
<thead>
<tr>
<th>Type: 11</th>
<th>Code: 0 or 1</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unused (All 0s)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Part of the received IP datagram including IP header plus the first 8 bytes of datagram data</td>
<td></td>
</tr>
</tbody>
</table>

Code 0: Time to live is zero
Code 1: Fragmentations are not arrived with a set time
In a time-exceeded message, code 0 is used only by routers to show that the value of the time-to-live field is zero. Code 1 is used only by the destination host to show that not all of the fragments have arrived within a set time.
Parameter Problem

- Occurred when a router or a destination discovers an ambiguous or missing value in any field of the datagram
- Code 0: there is an error or ambiguity in one of the header fields
  - Pointer field points to the byte within the problem
- Code 1: the required part of an option is missing
### Parameter-Problem Message Format

<table>
<thead>
<tr>
<th>Type: 12</th>
<th>Code: 0 or 1</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pointer</td>
<td>Unused (All 0s)</td>
<td></td>
</tr>
</tbody>
</table>

Part of the received IP datagram including IP header plus the first 8 bytes of datagram data

**Code 0:** Main header problem
**Code 1:** Problem in the option field
A parameter-problem message can be created by a router or the destination host.
Redirection

- Routing table is updated dynamically
- However, for efficiency, hosts do not take part in the routing update process
  - There are terrible number of hosts

- Host thus use static routing
  - Usually knows only one IP address of the router, the default router

- Thus, the host may send a datagram to the wrong router
Redirection Concept

Figure 9-11

A host usually starts with a small routing table that is gradually augmented and updated. One of the tools to accomplish this is the redirection message.
Redirection Message Format

<table>
<thead>
<tr>
<th>Type: 5</th>
<th>Code: 0 to 3</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IP address of the target router</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Part of the received IP datagram including IP header plus the first 8 bytes of datagram data</td>
</tr>
</tbody>
</table>

Code 0: Redirection for a network-specific route
Code 1: Redirection for a host-specific route
Code 2: Redirection for a network-specific route based on a specified type of service
Code 3: Redirection for a host-specific route based on a specified type of service

A redirection message is sent from a router to a host on the same local network.
Query

- ICMP can also diagnose some network problems
  - Accomplished by the query message
  - A group of four different pair of messages
Query Messages

Query

- Echo request and reply
- Timestamp request and reply
- Address mask request and reply
- Router solicitation and advertisement
Echo Request and Reply

- Determine whether two systems (hosts or routers) can communicate with each other
  - Determine if there is communication at the IP level
    - Because ICMP are encapsulated in IP datagrams
  - Also be used by a host to see if another host is reachable
    - At the user level, this is done by ping command
An echo-request message can be sent by a host or router.

An echo-reply message is sent by the host or router which receives an echo-request message.
Note

*Echo-request and echo-reply messages can be used by network managers to check the operation of the IP protocol.*
Echo-request and echo-reply messages can test the reachability of a host. This is usually done by invoking the ping command.
### Echo-Request and Echo-Reply Message Format

<table>
<thead>
<tr>
<th>Type: 8 or 0</th>
<th>Code: 0</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td></td>
<td>Sequence number</td>
</tr>
</tbody>
</table>

- **8**: Echo request
- **0**: Echo reply

Optional data
- Sent by the request message; repeated by the reply message

- **Identifier and sequence number**
  - Are not formally defined by the protocol
  - Can be used arbitrarily by the sender
Timestamp Request and Reply

- Two goals
  - Determine the round-trip time needed for an IP datagram
  - Synchronize the clocks in two machines
Figure 9-15

Timestamp-Request and Timestamp-Reply Message Format

<table>
<thead>
<tr>
<th>Type: 13 or 14</th>
<th>Code: 0</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td>Original timestamp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Receive timestamp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transmit timestamp</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Timestamp

- **Original timestamp**
  - Filled by the sender at departure time
  - Appear in both request and reply messages
- **Receive timestamp**
  - Filled by the receiver at receiving time
  - Appear only at the reply message and fill zero in request message
- **Transmit timestamp**
  - Filled by the receiver when the reply message departs
  - Appear only at the reply message and fill zero in request message
One-Way and Round-Trip Time

- Sending time = value of receive timestamp - value of original timestamp
- Receiving time = time the packet returned - value of transmit timestamp
- Round-trip time = sending time + receiving time
One-Way and Round-Trip Time (Cont.)

- Note that
  - Sending time and receiving time are accurate only if the clocks in the source and destination are synchronized.
  - The round-trip time is correct even if the two clocks are not synchronized.
  - See the following next slides.
Timestamp-request and timestamp-reply messages can be used to calculate the round-trip time between a source and a destination machine even if their clocks are not synchronized.
Example

- **Given**
  - Value of original timestamp: 46
  - Value of receive timestamp: 59
  - Value of transmit timestamp: 60
  - Time the packet arrived: 67

- **Derive**
  - Sending time = 59 - 46 = 13 milliseconds
  - Receiving time = 67 - 60 = 7 milliseconds
  - Round-trip time = 13 + 7 = 20 milliseconds
Example (Cont.)

- To show that the round-trip time is independent of time difference
  - Following above example, assume that the time difference is 3
    - Receiving node’s clock = 3 + sending node’s clock
    - Sending time = \((56+3) - 46 = 10 + 3\)
    - Receiving time = \(67 - (57+3) = 10 - 3\)
    - RRT = \((10 + 3) + (10-3) = 20\)
Synchronization

- Timestamp request and timestamp reply messages can also be used to synchronize the clocks in two machines.

- Time difference = receive timestamp - (original timestamp field + one-way time duration)
Example (Cont.)

- Time difference = receive timestamp - (original timestamp field + one-way time duration)

- Assume the one-way time duration can be obtained by dividing the round-trip time duration by two
  - Note that, this assumption may be wrong, depends on the network condition

- Thus, time difference = 59 - (46 + 10) = 3
The timestamp-request and timestamp-reply messages can be used to synchronize two clocks in two machines if the exact one-way time duration is known.
Address-Mask Request and Reply

- The IP address of a host contains
  - A network address
  - Subnet address if subneted
  - Host identifier

- A host may know its full IP address, but does not know its network, subnetwork address, and its host identifier
Address-Mask Request and Reply (Cont.)

- Masking is needed for diskless station at start-up time
  - It first asks its IP address using the RARP protocol when it boots
  - Then, it uses the address-mask request and reply to find out its mask
To obtain its mask, a host sends an address-mask-request message to a router.
- If it knows the router’s address, send the request directly to the router.
- If it does not know, it broadcasts the message.
# Mask-Request and Mask-Reply Message Format

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>17: Request</td>
<td>0</td>
<td>Identifier</td>
</tr>
<tr>
<td>18: Reply</td>
<td>0</td>
<td>Sequence number</td>
</tr>
</tbody>
</table>

```
Address mask
```
Router Solicitation and Advertisement

- A host needs to know the address of routers connected to its network
  - Send router-solicitation message by broadcasting or multicasting
  - The router receiving the message can then send the router-advertisement message
- A router may also periodically send router-advertisement message even if no hosts has solicited
Router Solicitation and Advertisement (Cont.)

- Note that, in a router-advertisement message
  - Contain not only its own presence
  - But also the presence of all routers on the network of which it is aware
Router Solicitation Message Format

<table>
<thead>
<tr>
<th>Type: 10</th>
<th>Code: 0</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifier</td>
<td>Sequence number</td>
<td></td>
</tr>
</tbody>
</table>
# Router Advertisement Message Format

<table>
<thead>
<tr>
<th>Type: 9</th>
<th>Code: 0</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of addresses</td>
<td>Address entry size</td>
<td>Lifetime</td>
</tr>
<tr>
<td>Router address 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address preference 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router address 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address preference 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>•</td>
<td></td>
<td></td>
</tr>
<tr>
<td>•</td>
<td></td>
<td></td>
</tr>
<tr>
<td>•</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Router Advertisement Message Format

- **Lifetime**
  - The number of seconds that the entries are considered to be valid

- **Address preference level**
  - The ranking of the router and used to select a router as the default router
  - If zero: the router is considered as the default router
  - If \(80000000_{16}\), the router should never be selected as the default router
CHECKSUM
Checksum

- In ICMP, the checksum is calculated over the entire message

- Checksum calculation
  - The checksum field is set to zero
  - The sum of all the 16-bit words (header and data) is calculated
  - The sum is complemented to get the checksum
  - The checksum is stored in the checksum field
Checksum (Cont.)

- Checksum testing
  - The sum of all words (header and data) is calculated
  - The sum is complemented
  - If the result is 16 0s, the message is accepted
    - Otherwise, it is rejected
### Example of Checksum Calculation

<table>
<thead>
<tr>
<th>8</th>
<th>0</th>
<th></th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>9</td>
</tr>
<tr>
<td><strong>TEST</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8 and 0 → 00001000 00000000  
0 → 00000000 00000000  
1 → 00000000 00000001  
9 → 00000000 00001001  
T & E → 01010100 01000101  
S & T → 01010011 01010100  
Sum → 10101111 10100011  
Checksum → 01010000 01011100
9.6 DEBUGGING TOOLS
Ping

- To find if a host is alive and responding

- Operations
  - Source sends ICMP echo request message (type: 8, code:0)
    - Start the sequence number from 0 and increments each time a new message is sent
  - Destination responds with ICMP echo reply message
  - Can calculate the round trip time
    - Insert the sending time in the data section
    - When the packet arrives, subtract the arrival time from the departure time to get the RTT
Example 2

$ ping fhda.edu
PING fhda.edu (153.18.8.1) 56 (84) bytes of data.
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=0 ttl=62 time=1.91 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=1 ttl=62 time=2.04 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=2 ttl=62 time=1.90 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=3 ttl=62 time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=4 ttl=62 time=1.93 ms

64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=5 ttl=48 time=84.7 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=6 ttl=48 time=84.6 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=7 ttl=48 time=84.7 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=8 ttl=48 time=84.4 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=9 ttl=48 time=84.2 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=10 ttl=48 time=84.9 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=11 ttl=48 time=84.6 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=12 ttl=48 time=84.5 ms

--- mail.adelphia.net ping statistics ---
14 packets transmitted, 13 received, 7% packet loss, time 13129ms
rtt min/avg/max/mdev = 84.207/84.694/85.469
Example 2 (Cont.)

- From above example
  - Sequence number starts from 0
  - TTL (Time to live) = 62
  - At the beginning
    - Data bytes is 56 bytes
    - Total number of bytes is 84 (= 56 + 8 + 20)
      - 8 = ICMP header, 20 = IP header
    - The total number of bytes in the ICMP packet is 64 (= 56+8)
Traceroute

- Trace the *route* of a packet from the source to the destination
- An application program that uses the services of UDP
- Elegantly use two ICMP message
  - Time exceeded
  - Destination unreachable
Traceroute Operation

- In the following figure
  1. the **tracert** program uses the following steps to find the address of R1 and RTT between host A and R1
     a. **Traceroute** at host A sends a UDP packet to B and set IP’s TTL field = 1
     b. R1 receives the packet and decrements the value of TTL to 0.
        - Discard the packet and sends a time-exceeded ICMP message since the TTL value is 0
     c. The **traceroute** at host A receives the ICMP message and find the destination address of the IP packet = IP address of R1
     d. Repeat the a–c steps three times to get a better average RTT value
        - Since the first RTT would by much longer since ARP operation
  2. **Traceroute** program repeats the previous steps with TTL = 2 to find the address of R2 and RTT between host A and R2
The Traceroute Program Operation
3. Repeats the previous steps to find the address of host B and the RTT between host A and B
   - When host B receives the UDP packet
     - Decrement TTL to 0
     - Receive the packet since it is the destination (do not discard)
   - How can an ICMP message be sent back to host A?
     - The destination port of UDP packet is sent to one that is not supported by the UDP protocol
     - Host B cannot delivery the packet to an application
     - Thus, host B discard the UDP packet and sends an ICMP destination-unreachable message to host A
   - Once host A receiving a destination-unreachable message with code value 3
     - The whole route has been found
Example 4

- The un-numbered line shows that
  - The destination is 153.18.8.1.
  - The TTL value is 30 hops.
  - The packet contains 38 bytes: 20 bytes of IP header, 8 bytes of UDP header, and 10 bytes of application data.
  - The application data is used by traceroute to keep track of the packets

- The first line shows the first router visited.
  - The router is named Dcore.fhda.edu with IP address 153.18.31.254.
  - The first round trip time was 0.995 milliseconds, the second was 0.899 milliseconds, and the third was 0.878 milliseconds

```
$ traceroute fhda.edu
traceroute to fhda.edu (153.18.8.1), 30 hops max, 38 byte packets
1 Dcore.fhda.edu (153.18.31.254) 0.995 ms 0.899 ms 0.878 ms
2 Dbackup.fhda.edu (153.18.251.4) 1.039 ms 1.064 ms 1.083 ms
3 tiptoe.fhda.edu (153.18.8.1) 1.797 ms 1.642 ms 1.757 ms
```
Example 6

- An interesting point is that a host can send a `traceroute` packet to itself.
- This can be done by specifying the host as the destination.
- The packet goes to the loopback address as we expect

```bash
$ traceroute voyager.deanza.edu
traceroute to voyager.deanza.edu (127.0.0.1), 30 hops max, 38 byte packets
1 voyager (127.0.0.1) 0.178 ms 0.086 ms 0.055 ms
```
ICMP PACKAGE
ICMP Package

- Input module: handle all received ICMP message
  - Invoked when an ICMP message is received
  - If the received packet is a request or solicitation
    - Create a reply or an advertisement and sends it out
- Output module: create request, solicitation, or error message requested by a higher level (TCP/UDP) or the IP protocol
ICMP Package

Results of error messages sent to processes that requested them

Reply messages sent to processes that requested them

Requests (from applications) to send queries

Requests (from UDP or TCP) to send error messages

Upper layers

Input module

ICMP packet (all types)

ICMP packet (replies and advertisement)

Output module

ICMP packet (requests, solicitation, and errors)

Requests (from IP) to send error messages