15-441 Computer Networks – Project 4
Distributed Document Routing and Forwarding

Hints Addendum

$Revision : 1.5$

December 1, 2002

This document provides hints and additional information about project 4.

1 Routing Module

1. **Q: Are node addresses (:IP-address:fwd-port) only stored locally, or are they exchanged via the routing protocol?**

   Node addresses and document names are exchanged among nodes via the distance vector routing protocol. This is because some packets are forwarded based on node address, other packets are forwarded based on the document name.

2. **Q: What information is included in the routing messages?**

   A Node's routing module exchanges routing messages with the routing modules of neighboring nodes. Each message contains a set of tuples of the form: $\langle destination, distance \rangle$. Here is an example of the contents of a routing message that includes 5 destinations:

$$\langle \text{“}document1'', 10\rangle$$
$$\langle \text{“}document2'', 12\rangle$$
$$\langle \text{“}document3'', 5\rangle$$
$$\langle \text{“} : 128.2.108.10 : 15200'', 9\rangle$$
$$\langle \text{“} : 128.2.108.10 : 15208'', 8\rangle$$

   The first three entries in the message indicate that the originator of the message can reach *document1*, *document2* and *document3* in 10, 12 and 5 hops respectively. The last two entries indicate that the originator can reach nodes :128.2.108.10:15200 and :128.2.108.10:15208 in 9 and 8 hops respectively.

3. **Q: What is the format of the routing messages?**

   Figure 1 shows a routing message format you can use:

   And Figure 2 shows how each entry in the routing packet looks.

4. **Q: What is the distance to documents?**

   If a document D is stored at a node N, then N's forwarding table contains an entry for destination D with distance 1.

2 Forwarding Module

1. **Q: What is the numeric value for the different packet types and error codes?**

   The packet types and error codes are defined in defined in rt_packets.h header file, which can be found in the 'lib' subdirectory of the project directory. Include this file in your C code. Below are the numeric values.

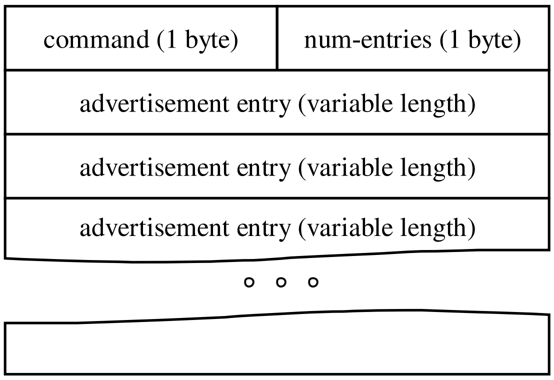

Figure 1: Routing message format.

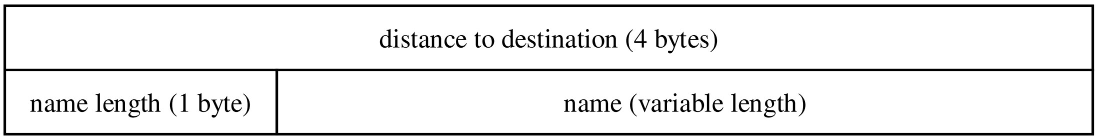

Figure 2: Routing message entry.

```
/**
 * Values for the the packet types
 */
typedef enum _rt_packet_type_t {
  RT_ERROR = 1,
  RT_LOOKUP_REQUEST,     /* 2 */
  RT_LOOKUP_REPLY,       /* 3 */
  RT_QUERY,              /* 4 */
  RT_REPLY,              /* 5 */
  RT_DATA_REQUEST,       /* 6 */
  RT_DATA_REPLY          /* 7 */
} rt_packet_type_t;

/**
 * Codes for error packets.
 */
typedef enum _rt_error_code_t {
  RT_NOERROR,            /* 0 */
  RT_TTL_EXPIRED,        /* 1 */
  RT_DEST_UNREACHABLE,   /* 2 */
  RT_DEST_UNKNOWN,       /* 3 */
  RT_TIMEOUT = 4         /* 4 */
} rt_error_code_t;
```

2. **Q: Is there a way in the packet format to know the length of the data field?**
   It can be derived as UDP packet size headers - forwarding packet header length.

3. **Q: How are packets encoded?**

   A packet is a stream of bytes. Simply make sure that the bytes appear in the order described in the handout, all fields are in network byte order unless specified otherwise. Suppose the packet data is stored in the packet variable below then the following should be the content of the array:

   `unsigned char packet[MAX_PACKET_SIZE]`

packet[0–3]	Source IP in network byte order.
packet[4–7]	Destination IP in network byte order.
packet[8–9]	Source Port in network byte order.
packet[10–11]	Destination Port in network byte order.
packet[12–13]	Sequence number in network byte order.
packet[14]	Type
packet[15]	TTL
packet[16]	Name length
packet[17–16+nam_len]	Document name (char array, no terminating NULL)
packet[17+nam_len–END]	Data

4. **Q: How do node addresses apply to all of the operations that can be done, i.e. LOOKUP, QUERY, etc. It seems that requests can only have IP addresses?**

   Packets handled by the forwarding module have both document names and node addresses. If you look carefully at the packet format described in Section 2.3 you will notice that there is a source node address `(source-IP:source-port)`, a destination node address `(dest-IP:dest-port)` and a document name (`doc-name` field).

   The table below shows which field is used to forward each type of packet:

Packet type	Field used
ERROR	Dest node address (dest-IP:dest-port)
LOOKUP_REQUEST	None, this packet type is not forwarded
LOOKUP_REPLY	Dest node address (dest-IP:dest-port) Generate the LOOKUP_REPLY with the client as the destination, i.e., `dest-IP` = `client-IP`, `dest-port` = `client-port`
QUERY	Document name
REPLY	Dest node address (dest-IP:dest-port)
DATA_REQUEST	Document name
DATA_REPLY	Dest node address (dest-IP:dest-port)

Figure 3: Fields used to forward packets.

5. **Q: How are document lookups done?. Does every node have to know about what documents are located on all the other servers in order to retrieve documents from them?**

   As described in Section 2.4 of the Project handout, the client sends a request to a node, To be more specific, the client sends a DATA_REQUEST packet to a node's forwarding port. The node remembers some information about the request, namely the packet sequence number and the client's address (IP-address:port). Then the node fills in the source-IP, source port fields in the DATA_REQUEST packet. Then the packet is forwarded as usual, the document name `doc-name` is looked up in the FT and the packet is set to the next hop toward the destination, i.e., `doc-name`. As you can see, a node needs to know about the documents located on all the other servers in order to retrieve the documents.

6. **Q: Is it possible to do a broadcast request for files, i.e., "I'm looking for a document of the form project*.*"?**

   No, in order to keep things simple, exact queries are required, no pattern matching searches need to be supported.

7. **Q: Does the client (or the user) have to know that node N has document X and that is the only way you can retrieve it?**

   No, the client does not need to know where the document is located in order to retrieve it. It only needs to know the name (X) of the document. The client can then contact a node to retrieve the document on behalf of the client as described above (5).

3 Handling Loops, Link and Node Failures and Expiring Table Entries

1. **Q: How do you clean out your document lookup table? Are there TTLs on the fields? Can they be thought of as other nodes on the network in the RIP protocol?**

   Entries in the Forwarding Table are cleaned as described in the RIP v2 RFC [1]. There are time stamps and timeouts in the forwarding table. As described in the Project handout, documents are *destination objects*, thus they are *destinations* in the forwarding table and in the routing messages. Yes, documents can be seen as other nodes on the network in the RIP protocol.

2. **Q: When and how are forwarding table entries deleted?**
   An easy strategy is to follow the RIP v2 specification described in RFC 2453 [1]. A forwarding table entry has a timestamp indicating the last time it was updated. Let T_o be the "entry timeout period". Let G_t be the "garbage collection time". If an entry has not been updated for a period of time T_o, then the entry is no longer valid, however it is retained in the forwarding table for a short time G_t, so neighbors can be notified that the route has been dropped. When a route is no longer valid, its distance is set to infinity, and this change is notified to the neighbors. After G_t has elapsed, the entry is deleted from the forwarding table. That is, the entry is deleted after a period of time $T_o + G_t$ has elapsed since the last time the entry was updated.

   T_o corresponds to the `-e` parameter in the command line.
   G_t corresponds to the `-g` parameter.

References

[1] G. Malkin. RIP version 2. RFC 2453, `http://www.rfc-editor.org/rfc/rfc2453.txt`, Nov 1998.

This RFC describes version 2 of the RIP protocol, which is backwards compatible with version 1.