## 1. Introduction

### 1.1 Purpose:

The goal of this design is to route packets in a network based on content identifiers as opposed to using a scheme based on IP or MAC addresses. End hosts make requests and responses using the content identifier.

### 1.2 Definitions:

- Node - A union of all routers and hosts on the content delivery network.
- Host - Peripheral nodes on the network which host content and request content.
- Router - Intermediate nodes on the network which route the requests and responses.
- Routing table - A table maintained at the router which stores information about the best possible path to a certain content.
- Pending request table - A table maintained at the router which keeps track of the requests to be serviced.
- Hop Count - Number of nodes that are traversed before delivering the payload.

### 1.3 Assumptions:

- Number of hosts connected to the network, N has a cardinality of 0...255.
- Number of content files K also have cardinality 0...255.
- Content files are no larger than a single packet of size 1500 Bytes.
- Locations of contents may change over time (insertions and deletions).
- Each end host can only connect to a single router.
- Same content may be available at multiple hosts.
- Metric for selecting best next hop is the hop count.
- Hosts can turn up and go down, but routers cannot.
- Links connecting network elements have probability of loss, p.
- A host does not ask for a new content, unless the last content it has requested for has been serviced.

## 2. Bootstrap and Discovery:

### 2.1 Addressing Scheme:

- Routers - Interfaces on the router are addressed as 0, 1, 2 …
- Contents - Contents are referenced by content IDs [0 - 255]
- Hosts - Hosts are identified by Host IDs [0-255]

## 2.2 Procedure:

**Announcement:**

- Update packets containing the contents each host possesses, are sent periodically by each host to their default routers, and the default routers will forward this information to their neighboring routers. If these routers see a better path, they will increment the hop count and forward the packets. These updates are sent by hosts every 10 seconds and the routers forward this information.
- At boot-up each host sends announcements to its default router indicating the contents it has. These announcements will propagate through the network.
- At the first encounter of a certain content id, the router will make an entry in its routing table.
- If the router receives an update packet for the same content id on another interface it will compare the two hop counts and replace the existing entry with the smaller hop count.
- In case of equal hop counts, it makes no change to the entry.

**Request- Reply:**

- A host interested in content sends a Request packet to its default router with Host Id & Content Id. It creates an entry for this request in the Pending Request Table.
- Each router adds an entry for HostId & ContentId and forwards the request to next router based on the Forwarding Table.
- The content providing host, on receiving the request, creates a Response packet with the content, HostId & ContentId and sends it to the neighboring router.
- Each router, routes the Response back to the requesting host based on the pending request table entry.
- An end-to-end ARQ scheme will be implemented where the requesting host times out and resends the request if no response arrives.

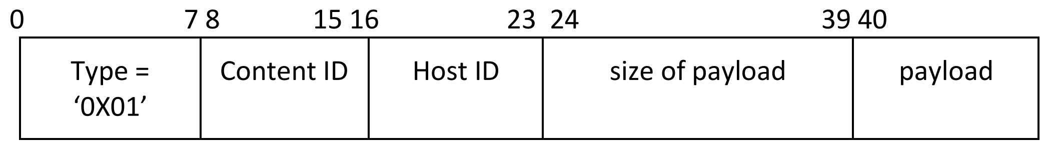
**Packet Format:**

- Query Packet
- Data Packet
- Timed Announcement

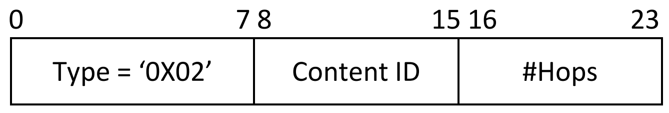
* Request Packet

| 0 7 | 8 15 | 16 23 |
|---|---|---|
| Type = ‘0X00’ | Content ID | Host ID |

* Response Packet



* Timed Update/ Announcement



## 3. Routing:

### 3.1 Content Routing:

There are two tables maintained at each router:

- Routing Table:

  The routing table is created and updated based on the timed announcements received on each interface. It comprises the Content ID, Interface and number of hops fields. It will insert or update entries depending on any new information in the timed announcements. Entries will be deleted if they time out. The timeout value for timers depends on the loss probability of the links and size of the network. For now we just can say it should be more than timeout of announcement timers. Here is an example of a routing table:

| Content ID | Interface | # Hops | Time to Expire |
|---|---|---|---|
| 2 | 7 | 3 | * |
| 8 | 3 | 20 | |
| ... | ... | ... | |

* *it should be set to CURRENT_TIME + EXP_TIME whenever we get an update for that entry.*

- Pending Request Table

  This table keeps track of the active requests and is useful for reverse routing of responses.

| Requested Content ID | Host ID | Incoming Interface | Time to Expire |
|---|---|---|---|
| 2 | 1 | 7 | * |
| 8 |  | 3 |  |
| ... |  | ... |  |

** it should be set to CURRENT_TIME + EXP_TIME whenever we get an update for that entry.*

When a router receives a request it searches the table using the Requested Content ID + Host ID pair to check if it is a repeat request. If not it makes a new entry and forwards this request on the interface by looking up the routing table. If it is a repeat request, it simply forwards the request on the appropriate interface without making a change to the Pending request table.

Once the router receives a response, it matches the Content ID and Host ID combination, forwards the response over the appropriate interface and then deletes the entry.

**3.2 Content Announcement:**

When the router receives a timed announcement it first checks to see if an entry with a better path already exists. If a better path exists, the router drops the packet and does nothing. If the new announcement has the same or lower number of hops, it resets the "Time to expire" field for the corresponding Content ID and updates the interface for that entry and forwards the packet on all interfaces except the incoming interface.
If the announcement packet does not reach the router, the router does not further announce the corresponding Content ID to its neighbors.
If a certain Content ID times out, the entry is deleted and if a new announcement is received a new entry is made.

## 4. Reliability Scheme:

We consider an End-to-End ARQ scheme for the ease of implementation.
If the requested content by a certain host is not serviced, it sends a repeat request for the same content again.

## 5. Timers:

### 5.1 Request Timeout for ARQ @ Host

This value can be less than or equal to the RTT. If it is very less, there will be lots of retransmits.

### 5.2 Periodicity of content updates @ Host

It can be assumed 10 secs.

### 5.3 Time to Live for entries in forwarding table @ Router

This should be greater than 10 secs.

### 5.4 Time to Live for entries in the Pending Request table @ Router

This should be much longer. It has to be at least greater than the RTT. Even if this value is very large, like 120 secs, it doesn't affect anything in the network. This is only to prevent infinite loops due to PRT entries not getting deleted.

## 6. Pseudocode

- **Announcement**

  Host - Announces all its contents periodically to its routers.
  For each content at host, an announcement packet is sent to Router.
  A timer is started. Resend, when timer expires

```
@Host
FOR each contentId
        Create announcement packet;
        Send announcement packet to connected router;
        Reset timer;
END FOR


@Router
FOR each received announcement packet
        Check received packet against Routing table for incoming contentId.
        IF no entry for contentId exists
                Increment incoming_distance;
                Add entry to forwarding table;
                Start timer for new table entry;
                At end of timer, remove table entry;
                Increment distance field in packet by 1;
                Forward packet on all interfaces except the incoming interface;

        ELSE IF incoming_distance < table_distance
                Increment distance field in packet by 1;
                Reset Time to Expire field to CURRENT_TIME + EXP_TIME;
                Forward packet on all interfaces except the incoming interface;

        ELSE IF incoming_distance >=  table_distance
                Drop packet;
        END IF
END FOR
```

- **Query**
  Host - Sends a query to its router. Starts a timer. Resends if it times out.
  Routers - Adds a PRT entry. Checks Routing Table and forwards to next hop router

```
@Host
FOR each request
    Create request packet;
    Send request to default router;
    Start timer;  On timer expiry, Resend the request
    Received_ind = 0;
    //On receipt of DATA packet
    IF packet received
        IF received_ind = 0
            Set received_ind = 1;
            Accept DATA packet;
        ELSE
            Drop DATA packet;
        END IF
    END IF
END FOR
```

```
@Router
Check received packet against Pending Request Table for incoming contentId, hostId
IF no entry for contentId + hostId exist in Pending Request Table
    Add entry to Pending Request Table;
    Forward packet based on Routing Table;
    IF no entry found in Routing Table
        Drop packet.
    END IF
```

```
ELSE IF one or more entries for contentId + hostId exist in Pending Request Table
    Forward packet based on Routing Table;
    Reset PRT timer for the existing entry;
    IF no entry found in Routing Table
        Drop packet;
    END IF
END IF
```

- **Data**

  Host - creates Data packet and sends to router
  Router - forwards based on PRT.

  @Host
  Create DATA packet ; set ContentId, HostID, Content;
  Send Data packet to default Router

  @Router
  Check contentId+HostId from DATA packet against Pending Request Table.

```
FOR each matching entry
    Forward packet over the interface found in the entry;
    Delete table entry;
END FOR
```