CHAPTER V
HEURISTIC APPROACH FOR MANAGING MEDIA TRAFFIC (Me-TRAP-Q)

5.1 Introduction

Real streaming media is the most sensitive form of multimedia content for media applications on network. Real-time media traffic requires quality consistency for play-out at receiver end. Without consistent QoS, real-time traffic can experience jitter, delay variations, and information loss. The delivery of continuous media from a central server complex to various distributed clients is a resource intensive task. The evidence of various traffic workload analyses [27, 69] indicates that client media-demand varies highly for differing time intervals. Media applications are characterized by stringent real time constraints, as each stream requires asynchronous parameters for play-out.

Me-TRAP-Q is an adaptive, heuristic based architecture aimed to manage traffic, minimize packet loss and reduce delay for media streaming applications. This scheme is an effective QoS management and adaptive traffic setup with minimized delay media packets resulting in less than 2% of loss in media packets. “Me-TRAP-Q” scheme adopts an intelligent and effective path or link adaptation technique, which selects the optimal buffer path for media transfer based on behavioral-traffic and heuristic “learning” approach. Policy adaptation schemes are equipped for various traffic streams depending on varying bit streams such as bursty traffic, VBR traffic, CBR. Effective utilization of variable parameters with rule based policy generation at run-time provide scalable, inter-operative traffic-scheduling scheme, which avoid congestion and minimizes packet loss.
Me-TRAP-Q scheme aims to minimize media packet loss and proves to work effectively than Me-PLM and Me-TrffSchl. This scheme also reduces play-out delay as well reduces the problem of jitter, which has not been focused in Me-PLM (Chapter-III) and Me-TrffSchl schemes (Chapter-IV). Me-TRAP-Q uses the novel idea of managing media traffic through priority-based channels or tunnels and effective buffer allocation. Me-TRAP-Q is an improvised, modified version of Me-TrffSchl. The priority assignment in scheme Me-PLM is adopted in this scheme also.

5.2 Traffic Differentiation Methods – Related Work

Study of the traffic nature in IP networks for varying intervals of time, varying number of users and applications had been discussed by Patrice[82] and Paul[84]. The discussions show that network traffic is always complex to the core of understanding with varying patterns of packet flow. Internet is a large domain with varying bandwidth from minimum bit rate to high gigabit Ethernet local area networks and mobile access traffic across broadband network. The temporal complexity of traffic pattern are time varying and take place over wide range of time intervals from microseconds to minutes.

5.2.1 Traffic Queuing Models

Miller [27] had studied the varying nature of network traffic with an effective analysis [34] with case studies. The analysis provides an insight into traffic pattern generation, complexity measures and required schemes to solve. The Random Early Detection (RED) algorithm provides an insight into congestion control research in area of Active Queue Management (AQM) [67]. IETF proposed standard adds explicit signaling mechanism ECN (Explicit
Congestion Notification) [117] by extending IP headers to notify the congestion phenomena or packet dropout. With ECN, a router can signal congestion to an end-system by “marking” a packet (setting a bit in header). RED scheme controls traffic congestion but do not result in better output.

RSVP [66] works in conjunction with existing queuing mechanisms, to reserve bandwidth. In IP-Multicast, reliability is the main challenge because there is not a bi-directional path from server to client to support retransmission of lost packets. Among the transport protocols being developed for streaming data, RTP, RTCP and RTSP are used for real time multimedia delivery. RTP do not provide any mechanism to ensure timely delivery or QoS guarantees nor take care of the network parameters to control packet loss. Uninterrupted media requires a reliable transport layer or scheme to handle media packet loss and control traffic to provide a minimized delay play-out at receiver end.

5.2.2 Buffer Management

Allocating the required buffer, when chance of packet loss can take place, is a simple remedy in QoS management. The service policy handles the main role of managing available buffer while providing sufficient QoS for media transfer over network. The scenario in Internet is that the available buffer capacity is less, but link speeds are in order of gigabits per second. The packets that arrive during these transient congestion periods of time are dropped. To implement a reliable, minimized end-to-end delay for streaming media, sufficient buffer capacity should be provided. The buffer capacity should be a variable parameter based on requirement, where optimality should be achieved. Consequently various proposed buffer management schemes and comparison methods have been discussed in literature [1] in
terms of trade-off between performance and complexity. Buffers and traffic in routers can be managed in different degrees of aggregation. In best effort networks, the scheduler maintains an aggregate buffer pool for all connections, which provide the same quality of service for all traffic.

5.2.3 Traffic Policing for Malicious Flow

Traffic Policing Methods are defined in IntServ [83] and DiffServ [90] models. Traffic policing or traffic admission refers to packet admission and control mechanism at gateways. The ETSI Technical report ETR003 [38] reviews various factors influencing the perceived QoS such as the Communication Establishment Delay, Probability of Blocking, and the Effective Bandwidth (EB). The first two delays are specified for circuit switching, while Effective Bandwidth is a common nature in packet switching. Communication Establishment Delay is the time interval between initiating the connection request including the complete messaging process and completion of call session at calling terminal. If the media packets wait at the queue for a long time interval (crosses TTL limit [36]) then packets are branded as old and have to be dropped. Media packets crowded at gateway undergoes collision, resulting in loss of packets.

To support media traffic consistently and reliably, a network must be able to provide packet-forwarding latency that do not exceed the “maximum tolerable level” for a ‘media’ conversation. Packet-forwarding jitter, which is the variation of latency over time, do not exceed the maximum tolerable level to sustain a real time session. Similarly guaranteed network bandwidth and dynamic variable buffer capacity for real time sessions during periods of network congestion should be provided.
5.3 Me-TRAP-Q Design Approach

The Me-TRAP-Q works on adaptive heuristic-based "learning" strategy, which controls the flow of media traffic effectively to increase throughput of media transfer over highly congested broadband networks. This scheme implements an effective media traffic controller, which is intelligent to provide an optimal path for managing media traffic flow. The heuristic approach tries to minimize media packet loss to less than 2% during network "bursty" conditions, while an average of 3.07% and 4.52% is identified for packet loss in schemes DiffServ and IntServ respectively.

Me-TRAP-Q scheme focuses on controlling the flow of traffic of media packets. Allocation of a media packet depends upon the intensity of per-flow traffic between end-to-end hop of a domain. Packets can flow in both directions from set of ingress points as well egress points.

![Traffic Flow Modeler](image)

Fig 5.1 shows Traffic Flow Modeler Module, which manages the congested traffic in Me-TRAP-Q Scheme. This module accepts media packets and generates policy to assign them to an optimal out-going path, which can provide the minimum possible loss of media packets with minimized traffic. Deciding the optimal path for media packet allocation is based on Behavior Traffic Class, Packet Type, Link / Queue Class and Delay Stamp priority.
5.3.1 Traffic Adaptive Queue Control Provider (TRAP)

Me-TRAP-Q approach incorporates effective utilization of buffer states by optimizing their activity periods. Buffer from servers, which are idle and faulty, are removed from activity state during less traffic intensity period. Inclusion of an additional buffer is required from idle state to active state when traffic intensity is high during service times. The dynamic and adaptive buffer management module works on K-Server threshold vector, where ‘K’ being finite number of buffer / links. Any increase in media packets on a link lead to increase in demand for bandwidth, which allocates buffer from ingress points by freeing the idle buffer. Adaptive Queue Control Manager classifies, controls the buffer of both incoming and outgoing queues. Traffic Adaptive Queue Control Provider (TRAP) uses context definitions for each media packet, its corresponding traffic and set of properties.

Context Definition

Context is defined for a stream of packets such as Audio, Video, Images, Text Data or any other types identified in buffer. Context defines the Context-ID, Context-Stream, Stream-Type, and Stream-Traffic. Context is defined for all packets in a gateway or router. Context defines the ID (Identity) for each packet.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context ID</td>
<td>Identity of defining a media context. Represents the corresponding Stream Type and Traffic Nature.</td>
</tr>
<tr>
<td>Stream Context</td>
<td>Type of stream based on application in use.</td>
</tr>
<tr>
<td>Stream Type</td>
<td>Corresponding events to be initiated when a context is called, and managing exceptions.</td>
</tr>
<tr>
<td>Stream Traffic</td>
<td>Managing the audio capability of a receiver device and adjusting its volume, speed of receiving etc is handled by Context handler and Services.</td>
</tr>
</tbody>
</table>
It defines the attributes or properties of the resource components, its corresponding limitations and events on activation. Context provides the dynamics, scope, usage and limitations of each packet, which consists of various attributes as in Table-5.1.

5.3.1.1 Defining Stream Context (Marking Packets)

The incoming packets are analyzed and segregated based on the type of incoming packet and packet early-life-time (TTL). The packet analysis determines the type of packet, (media packet, data packet or web packets). The Media packet's STREAM_CONTEXT field provides the type of media payload context, which can be as in Table-5.2.

<table>
<thead>
<tr>
<th>Priority</th>
<th>Packet Stream Context (PS)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Real Time Streaming Media</td>
<td>Streaming Audio and Video payloads of interactive applications where minimum number of frames created is more 2500 frames per second, requiring high refresh time.</td>
</tr>
<tr>
<td>2</td>
<td>Real Time Media</td>
<td>Media payload for online play-out of media applications, minimum number of frames created are more than 1800 ps.</td>
</tr>
<tr>
<td>3</td>
<td>Offline Media</td>
<td>Offline play-out of video and audio applications, where the required buffer per second is less than 1700ms to 1200 ms one way with faster refresh time.</td>
</tr>
<tr>
<td>4</td>
<td>Web Media</td>
<td>Web traffic applications, based on &quot;http&quot; protocol, where the required buffer per sec is less than 500 per second.</td>
</tr>
<tr>
<td>5</td>
<td>Data Payload</td>
<td>Normal e-mail and text based chat applications, where data transfer rate is less than 120 frame packets per second.</td>
</tr>
</tbody>
</table>

The Stream-Context can be identified based on user's information of the type of media application or the inter-arrival time between packets. Packet Classification (PS) field holds the Stream Context value and early lifetime of packet.
5.3.1.2 Defining Stream Traffic (Marking Traffic Stream)

Traffic Streams are classified based on traffic generated by media application. Traffic Stream Types are Interactive Media Stream or Real Time Media Stream that is bursty - bit rate in nature. Off-Line Media Stream or Web Stream Traffic that is Variable Bit Rate and other type is data traffic. Classification is done based on inter-arrival time of packets, media-type packets (PS Field) and flow behavior of packets (variable or constant bit rate).

The need for classifying and identification of packets is as follows:

a) Media Stream applications: Maintain time-critical events and accurate timing information for interactive media streaming applications, which require high effective bandwidth.

b) High Degree of Schedulability: Refers to degree of resource utilization by which the deadline of each time critical task has been considered.

c) Scalability under transient heavy load: The processing of critical tasks must be ensured. The incoming packet per second is analyzed and marked based on type of packet, flow interval per second, and delay stamp time, which are vital to basic functionality of media system. The traffic stream is classified by following parameters:

<table>
<thead>
<tr>
<th>Traffic Stream Context</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STREAM_TYPE</td>
<td>Identifying Stream Traffic Flow</td>
</tr>
<tr>
<td>STREAM_PRIORITY</td>
<td>Assigning Priority to traffic stream</td>
</tr>
<tr>
<td>CONTEXT_STREAM</td>
<td>Buffer Shaping based on available Traffic Intensity</td>
</tr>
</tbody>
</table>

5.3.1.3 EPT (Early Play-out Time) and EDT (Early Delay Time)

Early Play-out Time (EPT) and Early Delay Time (EDT) have been discussed in Me-TrffSchl scheme. EDT and EPT parameter insists on media packets to reach the destination end within the specified play-out time. Play-
out time depends on application in use. Early Delay Time (EDT) is similar to EPT, where the media packets are provided an extendable delay time (in ms) to reach its destination. This parameter consists of an extendable delay time of required early play-out time (EPT), which a media packet can take to reach its destination. EDT value is assigned to corresponding Me-TRAP-Q header parameter. This parameter is a grace time limit, which is acceptable for the media packet to be received and played out by receiver.

An on-line real time video packet should be played out at receiver end within its specified EPT, but in congested network scenario, a media packet takes 160ms to 250ms to reach its destination. From literature study it is found that, Best-Effort scheme takes 220 [31], due to network latency effects. While DiffServ takes 189ms [15] and IntServ takes 210ms[43]. Me-PLM scheme takes 180ms(Chapter-III) and Me-TrffSchl scheme receives the media packet at 130ms (Chapter-IV), which is comparatively better compared to other schemes.

5.3.2 Behavioral Stream Manager

Behavioral Stream Manager module selects a set of optimal links for transfer of media stream in the network. Multiple links in gateway are selected with required buffer and tolerable traffic flow intensity to allocate the media packet. The selected links should provide feasible solution, where packet loss should be minimal and reduced play-out delay time (round-trip time), if play-out delay is minimized then jitter is also minimized.

The Heuristic based Behavioral Stream Manager is an intelligent functional module, which manages the gateway - network's link status of neighboring gateways in the domain by continuous "probing" and status
“updating”. This function updates the status of link as Busy, Idle, Faulty or Active depending on gateway server’s servicing state. The Behavioral Stream Manager is explained in Fig-5.2 for a media conference application. The figure shows multiple paths available in a gateway with varying port-no and sockets to be established for transfer of media stream. Links are selected during runtime based on incoming packet flow, available flow in link, adaptable traffic intensity and link status obtained from Link Status Table.

“Adaptive Queue Scheduler” works in co-ordination with “Behavioral Stream Manager” to identify the optimal path for media transfer. Behavioral Stream Manager identifies the best traffic and weak stream traffic of a link by continuous “probing” behavior. The method of heuristic learning adheres to continuous update of system resources and transmission process. The continuous learning method is an effective update, which identifies the best course of out-going transfer path for traffic flow. Flow intensity is obtained from “Stream Reporter” Module. The result provides an optimal path with minimal traffic and required buffer for transfer.

![Fig 5.2 Heuristic Link Scheduler](image-url)
The selected link is used for establishing the flow with required ports and sockets between multiple hosts.

5.3.2.1 **Link Status Table**

This table maintains the complete status of all links in the gateway. Status information such as traffic intensity at various time-intervals, variation for different packet inflow, regular fluctuations, intolerable conditions and faulty conditions are updated.

5.3.3 **Tunneling Mechanism for Traffic Flow**

The method of classifying stream traffic into channels or tunnels provides great advantage in controlling and monitoring the flow of packets. The method of defining the tunnel depends on the available buffer capacity at routers. Based on incoming flow of packets at routers, the buffer capacity of a link is segregated or channelized into multiple tunnels. The size of each tunnel is stateless and varies depending upon traffic flow intensity in link. The classification and size of each tunnel depends on:

1. Aggregate incoming packets (bytes per second).
2. Traffic intensity of the selected link at gateways or routers.
3. Managing Early Delay Time (EDT) for media packets, which tries to provide highest priority for transfer of media packets.

Tunnel is a virtual separation between multiple flows of packets in buffer. Fig-5.3 explains the tunnel mechanism. Each flow of packets in buffer is serviced based on priority assigned to the tunnel.

Management of tunnel depends on buffer allocation, and its classification is similar to Me-TrffSchl Scheme. Classification of buffer for allocation of media packets is at run-time, based on number of packets
waiting in queue. Here the allocation of packet into buffer is based on Markovian Birth-Death model. The system follows an exponential packet inter-arrival time, where a packet can be allocated into queue if required buffer is available. On contrary, virtual additional buffer is requested, if required buffer is not provided, then less priority packets are dropped and media packet is assigned into queue. Buffer State is classified as Busy, Idle, Faulty, and Active based on status of server in network. Busy indicates that server is currently handling the service of packet.

<table>
<thead>
<tr>
<th>Buffer State</th>
<th>Priority</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active (1)</td>
<td></td>
<td>Requested buffer is waiting for incoming packets to be scheduled. Ready to be called for service.</td>
</tr>
<tr>
<td>Idle (2)</td>
<td></td>
<td>Requested buffer is Idle, not in active state. Free from fault but has to be called into work.</td>
</tr>
<tr>
<td>Busy (3)</td>
<td></td>
<td>Requested buffer is occupied with servicing packet on allocation. Cannot be extended into service</td>
</tr>
<tr>
<td>Faulty (4)</td>
<td></td>
<td>Requested buffer state is in intolerable state and cannot be called into service.</td>
</tr>
</tbody>
</table>

If the available buffer is 10Kbps, while aggregate incoming packet is 8Kbps, and packets waiting in queue to be serviced are 3Kbps, then tunnels are created. ALP module identifies the type of stream packets, its required buffer capacity and flow of traffic intensity, Fig-5.3 explains tunneling method.

The priority of packets carries importance in allocating packets. Based on buffer state and traffic behavior, required buffer capacity is selected. Buffer is segmented into virtual buffers or called tunnels based on inflow traffic of packets. The available 10Kb buffer is segregated into channels where the maximum buffer capacity is provided for online real time streaming media packets. Online media packets with high EDT carry highest tunnel “A”, which occupies 8Kbps of buffer capacity. This tunnel is triggered off for higher service. Off-Line media packets are placed in tunnel “B”, occupying 1Kb of
buffer. Web traffic packets occupy last priority with remaining 1Kb in tunnel “C”. Tunneling assigns at-least 75% of buffer for media streaming packets, and rest occupied by web packets. The allocation scheme is done on basis of Packet Policy Admission Procedure.

Me-TRAP-Q approach provides effective utilization of buffer states by minimizing their activity periods. Any increase or decrease in media packets on a link leads to addition or removal of buffer from ingress points within the domain.

5.4 Me-TRAP-Q Architecture

Fig 5.4 shows the architecture of Me-TRAP-Q, which consists of multi-tier stack arrangement of inter-dependent modules TRAFFIC ADAPTIVE QUEUE PROVIDER, PACKET CLASSIFIER, BEHAVIORAL STREAM MANAGER and PACKET POLICY ADMISSION MODULE.
Me-TRAP-Q adopts the method of heuristic based behavioral stream manager, which selects the queue or link. The Traffic Adaptive Queue Provider assigns an incoming packet into tunnel-buffer based on packet policy admission procedure.

The system performance considers the following criteria:

a) Allocation of media packets serviced based on servicing capacity

b) Number of media packets in wait

i) **Packet Classifier and Marker Module** checks each incoming packet identifies their types (Table-5.1). The context of each packet, type, and corresponding stream nature are identified and classified packets are sent to next layer.

ii) **Behavioral Stream Manager** identifies all feasible links between incoming and outgoing flows. Available buffers, their traffic intensities, their status are learned through heuristic approach. This layer functions on “learning” and continuous update of Link Status Table.

iii) **Stream Reporter** module informs on available links and corresponding traffic flow. This layer functions as an internal component of Behavioral Stream Manager, to manage and inform link traffic to Policy Manager

iv) **Traffic Adaptive Queue Provider** works on assigning buffer on demand. This module

1. Checks for availability of buffer capacity at out-going links

2. Provides priority on media packets and traffic

3. Checks for traffic intensity at routers / gateways and allocates buffer.

This module manages the flow of media -
- Identifies number of media packets to flow at each interval of time between specific end-points, controlling media traffic.
- Provides priority for media packets with Early Delay Time stamp (EDT) where each real time media packet is regulated in flow by stamping delay-time at gateways/routers. (Stamping provides hierarchy for flow between end-points)
- Maximum available buffer at each gateways/routers is prioritized for media transfer which are assigned for real time media packets.

This module regulates the media traffic flow and minimizes delay. Prioritizes media packet and segregates media traffic flow into independent stateless channels or tunnels such that each channel is controlled and managed effectively by dynamic adaptive manager. Tunnels are stateless in nature, such that flow of media traffic is scheduled depending on priority of traffic to be transferred. The Me-TRAP-Q packet header format of 32 bits is shown in Fig-5.5.

![Fig-5.5 Me-TRAP-Q Packet Header Format](image)

The 32-bit header format consists of:
- 8-bit EPT, specifies Early Play-out Time
- 8-bit EDT, specifies Early Delay Time
- 8-bit Context-ID, specifies the Context of Stream of media application
- 8-bit Stream-Context, specifies type of Stream
- 8-bit Stream-Traffic, specifies traffic intensity of stream
- 8-bit Tunnel Format, specifies the hierarchy of stream

Whenever media traffic exceeds the assigned buffer tolerance limit, enough alternative buffers are extended into activation by which additional media packets can be serviced. This method reduces severe degradation in throughput and latency delay can be avoided. This allocation depends on
availability of space in buffer with already 'i' packets waiting in queue, which forces a non-instantaneous activation of an additional server resulting in buffer increase. Similarly servicing of group of packets than assigned interval of time forces removal of a server from activation.

v) **Policy Manager** is a rule based service module, which controls the activity of QoS traffic parameters, and inter-coordinate together to assign any resource or alter or refuse any resource from activity. Various policy methods are being discussed to invoke the corresponding services. The resource availability, status of resource are identified during run time, the media stream has to undergo variations in accepting a resource or altering the resource.

Three policies are used in this scheme, which are explained as below.

(a) **Policy Negotiator** identifies the resources and selects the parameters, which are reserved for use both by call originator and call receiver. Negotiation process is carried out on the common set of required parameters, which can provide guaranteed QoS for media applications.

```
Policy_Negotiate [ < Media Parameters > ,
                 < Priority = Value > ]
```

Here,
Media Parameters – Set of variable parameter used for negotiation
Priority – Assigning priority type for each parameter

(b) **Policy Generator** is based on required QoS parameters. Policy Generator is being done on basis of Negotiation and Acceptance procedures. The generated policy is in acceptance between various hops in a network domain or in a call process between various endpoints. If a call of higher priority requests the components for call
establishment, then component can be freed or allotted for use on share basis.

**Policy_Generate**

\[
\text{Policy\_Generate} [<\text{CallerID}>, <\text{CalleeID}>, <\text{Policy\_ID}>,
<\text{Policy\_Negotiate\_Parameters}>]
\]

Here,
- CallerID – IP Address of Call Originator
- CalleeID – IP Address of Call Receiver
- PolicyID – Identity of policy generated for each call session

(c) **Policy Adaptation** adapts to any change in end-to-end environment. Change in any resource status provides major “hit back” at run-time. Both end point users should accept any change in parameters. Instances such as change of resource status / availability, resource reservation mechanisms, and user preferences are primary constraints leading to policy adaptability at run-time. Buffers in routers can be managed in different degrees of aggregation. In general networks the scheduler maintains an aggregate buffer pool for all connections, which normally provides the same quality of service for all traffic.

**Policy_Adapt**

\[
\text{Policy\_Adapt} [<\text{Resource\_Available} = \{\text{Resource\_ID}\}>,
<\text{Resource\_Status}=\text{BUSY} | \text{IDLE} | \text{FAULTY} | \text{NO\_RESPONSE}>,
<\text{PortID}>, <\text{SocketID}>, <\text{ChannelID}>,
<\text{Policy\_Negotiate\_Parameters}>]
\]

Here,
- ResourceID – Identity of resource type allotted
- ResourceStatus – Status of resource in use
- PortID – Port Number in Use
- SocketID – Network socket assigned for stream
- ChannelID – Channel assigned for media stream

Fig-5.6 explains the sequence of steps taken for establishing a logical connection for transfer of media data between an end-to-end hop and control of media stream managed by Policy Manager.
Policy Negotiate negotiates between User-A and User-B to identify the required set of QoS parameters, which can provide guarantee. The guarantee is undertaken by all intermediate routers to select on a link with required buffer capacity through out media transfer over domain or egress nodes, where the Policy holds good.

- The Policy Adapt is called to adapt the set of QoS parameters, with both set of users User-A and User-B.

- Policy Generator generates policy with required parameters. The Context ID is valid only inside a Domain or sub-Domain once it comes out of Domain a new Policy is generated.

Fig-5.9 shows sample pseudo-code on controlling media stream using Policy Manager.

```
Policy_Negotiate {
    Context_Stream = REAL,
    Context_Media = MEDIA,
    Bandwidth = 100000 ,
    Context_Status = HIGH )
}

Policy_Negotiate {
    Context_Stream = REAL,
    Context_Media = MEDIA,
    Bandwidth = 100000,
    Bandwidth_Extend=90000,
    Context_Status = HIGH )
}

Policy_Adapt {
    Resource_Allocate = TRUE,
    Port_Id = 0x3840, Socket_Id = 1071,
    IP_Address = 123.231.078.125 )
}

Policy_Generate( User_A, User_B,
    Context_Stream = REAL,
    Stream_Type = MEDIA_STREAM,
    Traffic_Stream = 10000bps ,
    Bandwidth =170000,
    Port_Id = 0x3840, Socket_Id = 1071 )
```

Fig-5.6 Policy Manager

5.5 Working Model of Me-TRAP-Q

Fig-5.7 explains the working model of Me-TRAP-Q. This algorithmic scheme works in a gateway of Internet domain. The model considers a set of
domains and sub-domains, where set of ingress points and egress points are identified. Me-TRAP-Q scheme identifies all out-going links and incoming links and renews its state each time before transmission process. The Links Status Table resides in gateway where Me-TRAP-Q scheme is executed.

Incoming packets of 10kbps are analyzed for their type and payload structure. Streaming media packets of 5kb is identified, while off-line media packets of 3kb are identified. Streaming media packets are stamped with minimum EDT followed by off-stream Media packets. Data packets are not stamped with EDT. The traffic flow intensity and status of out-going links is gathered by Stream Manager and continuously updated in Link Status Table. Behavioral Stream Manager identifies feasible links where an optimal solution is obtained. Traffic Adaptive Queue Provider identifies the best link from set of optimal paths and based on incoming traffic type, tunnels are segregated with varying buffer capacity. Streaming media packets are assigned highest priority, with required buffer into tunnel.

Fig 5.7 Me-TRAP-Q Working Model
Step 1: Inputs segregated media packets and required buffer
2: Behavioral Stream Module identifies Traffic Flow Status from Link Status Table
3: Out-going Link’s Status is gathered from incoming stream of packets.
4: Policy Manager Module negotiates with Traffic Stream Profile Module on identifying the required link with stream-flow.
5: The result is returned to Policy Manager Module
6: Stream Reporter updates the stream flow and corresponding link status to Stream Profile
7: Policy Manager accepts the selected and updated optimal link for assigning packets

Fig-5.7 explains the step-wise approach in selecting the optimal link for assigning packets, where the latency effects are less. Algorithm-1 shows the sample pseudo-code for identifying the media stream and assigning into a channel. Algorithm-2 shows the pseudo-code for creating the channels and controlling stream on channel.

```
For (CONTEXT_ID = 0x01)
   If (CONTEXT_STREAM = Real) AND (STREAM_TYPE = Media) AND
      If (STREAM_EDT > STREAM_EPT) Then STREAM_TRAFF = HIGH
   Begin
      /* Assign maximum buffer at egress points */
      Assign (Channel1)
      /* Decide on Adaptive Behavioral Path */
      /* Each incoming media packet has equal buffer to be serviced in channel. */
      CHANNEL channel1 = CONTEXT_ID
      STREAM_POLICY = NON-DETERMINISTIC
      /* Assign dynamic buffer for buffer allocation */
   End
End If
End For

Algorithm : Identifying media packets and Channels Assignment

/* defining channels to assign packets */
# define B[channel0] 100000 /* Buffer capacity 100kbps */
# define P[channel0] media /* stream types media */
# define T[channel0] high /* stream traffic high */
# define B[channel1] 1000 /* Buffer capacity 10kbps */
# define P[channel1] media /* stream types media */
# define T[channel1] medium /* stream traffic high */
# define B[channel2] 1000 /* Buffer capacity 100kbps */
# define P[channel2] media /* stream types media */
# define T[channel2] low /* stream traffic high */
```
/* Control using Policy Manager */
ClientA = PolicyNegotiate (Context_Stream, Context_Status, channel0)
/* channel0 is the buffer resource requested for allocation */
ClientB = PolicyNegotiate (Context_Stream, Context_Status, channel1)
ClientC = PolicyNegotiate (Context_Stream, Context_Status, channel2)
If (ClientA = TRUE) then
  Begin
    PolicyAdapt(channel0, TRUE, Context_Stream )
    /* assign stream to channel0 */
    PolicyGenerate (ClientA, channel0, ContextStream, ContextType)
  End
End If

Algorithm : Creating Channels and controlling stream using Policy Manager

5.6 Experimental Test Bed

Me-TRAP-Q Scheme was tested using two different experimental setups. A video play-out media setup was used to gather audio data from four different locations. The second experiment setup is done using media probing through different routers.

Experiment 1: Video Play-Out

Fig-5.8 shows the Network Model used in our simulation method. The experimental setup is WAN interconnection where sender is Fighter network server at MAT Systems, Denver, US that transmits media data through network, which is received by Birds network server at MAT Systems, Coimbatore, India (Receiver). A digital video play-out setup is established between client nodes in both networks (Fig-5.8) where users conducted different request and call procedures among them. Video play-out software was used to capture the streaming media and play-out. The software enables maximum of six different sessions to connect online for transfer on minimum required bandwidth of 50Mbps at receiving end for media packet play-out. The experiment was carried out over two different Ethernet networks connected to
lease line Internet through multiple routers and gateways. A monitoring
module is maintained at the routers or gateways, to gather the media packet
information and forward, it to the server of corresponding networks. Various
network packet transfer parameters and networking latency factors are
determined shown in Table-5.5.

![Network Setup](image)

*Fig-5.8 Network Setup used for Simulation*

Four movies were selected for transmission by sender and played out
at receiver end. Table-5.5 shows the list of movies, date of experiment,
varying time intervals, network (A) from MAT Systems, Denver, US and
network setup (B) at MAT Systems Inc, Coimbatore, India. The set of movie is
being transferred by sender (A), which is received and played out by receiver
(B). Similarly another movie is transferred at sender (B) and received at
receiver (A). The experiment is repeated for three different days for differing
time-intervals. Different dates and time intervals for test is considered to study
the scheme under varying test environments. SNMP MIBs help in gathering
the traffic intensity, packet loss at each network and response delay time. The
results are submitted in Table-5.5.
Fig-5.10 shows packet loss obtained for DiffServ model and Me-TRAP-Q scheme. The percentage of average packet loss in DiffServ model is found to be 3.25%, while Me-TRAP-Q scheme exhibits an average packet loss of 1.15%. It is found that DiffServ is not found to be effective for discontinuous bursty traffic intensity, while Me-TRAP-Q manages the bursty traffic by assigning them in channels with higher priority. Fig-5.11 shows end-to-end delay obtained between DiffServ model and Me-TRAP-Q scheme. DiffServ model shows an invariably higher delay as compared to Me-TRAP-Q scheme.

**Experiment 2: Traffic at Routers**

This experiment aims to prove the feasibility of Me-TRAP-Q scheme in existing router setup. As well the setup is simple and adaptable to different network setups working under different browsers and operating systems. Dynamic traffic propagation which drives the problem is based on recent large-scale analysis of media traffic [27]. This model carries an application-level description of critical elements and required components, which decide on the effectiveness of providing QoS. Me-TRAP-Q scheme uses critical elements that characterize web centric protocols HTTP/1.0 and HTTP/1.1 protocols. An important property of this model is that it reflects the use of persistent HTTP connections as implemented by many contemporary browsers and servers. Other protocol components used to identify media traffic are RTP, RTCP for real time media packets.

Real time streaming data is identified by streaming protocol element, streaming protocol [55] such as RTSP or ST2, which is used to stream voice, video and other real time transmission data is used to stream data. The offered media load is defined as network traffic resulting from emulating
browsing behavior of a fixed-size population of web users or type of media applications activated by different users connected to routers. The test bed shown in Fig-5.9 consists of a C-coded program running at agent system to generate media requests and gather the traffic generated by probing or replies.

![Fig-5.9 Gather Traffic at Routers - Experimental Approach](image)

5.7 Simulation Test

Table-5.6 displays the response time and packet loss gathered at various routers belonging to North American domain. The results were obtained by continuous probing for different intervals of time. The results are provided under a simulated run setup for Me-TRAP-Q scheme. The agent or probe system resides in MAT Systems Inc, Coimbatore, India. Simulated results obtained on basis of variable set of incoming media packets provide optimal results.

**Result: Average Delay (Response Time)**

Fig-5.12 shows the number of media packets received within the specified delay time for a set of sources (media packets) transferred over source links of 100Mbps of Ethernet-LAN in MAT Systems Inc, Coimbatore.
Response time for Best-Effort system is found to be aggressive than MeTRAP-Q scheme. The result shows the performance of the algorithm by which the source of media packets increases per interval of time the receiving time (ms) increases, but it maintains a consistent time of average varying between 110ms–124ms for any increase in media packets reducing the delay time taken to play-out. The queue buffer assigned is maintained between 62kbps – 75Kbps, which is desired criteria for avoiding congestion. Fig.5.13 shows the percentage of media packets lost for DiffServ scheme with Best-Effort and MeTRAP-Q. The figure shows that an average of 0.98% loss is found in MeTRAP-Q scheme.

5.8 Summary

The algorithm produces the best effort result as the time taken to receive the media packets over network whose bandwidth is variable depending on number of incoming media packets at an interval of time. MeTRAP-Q scheme is found to be effective for controlling media packet loss and minimizing delay. Considerable results are achieved when queue size is variable, which can allocate buffer dynamically. Minimizes media packet loss negligibly, to an average minimum of 1.13%, which is above the latency value.

Outcome of this scheme are:

1. Better results are achieved, when queue size is between 800kbps to 1200kbps
2. In general, average packet loss is found to be 2.18%
3. Media packet loss alone is identified as 1.13%
4. Average play-out delay varies from 78ms to 82ms for media streaming applications
5. End to end delay is found to be in average as 120ms.
Table 5.5 Video Play-out Setup – Experiment 1

Test Environment

<table>
<thead>
<tr>
<th>Movie List</th>
<th>Date:</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL – Silence of Lambs</td>
<td>1st set of movie - 12th March 2003</td>
</tr>
<tr>
<td>DH – Die Hard</td>
<td>2nd set of movie - 23rd March 2003</td>
</tr>
<tr>
<td>PA – Project A</td>
<td>3rd set of movie - 5th April 2003</td>
</tr>
<tr>
<td>JK – Jungle King</td>
<td></td>
</tr>
</tbody>
</table>

Movie List:
SL - Silence of Lambs
DH - Die Hard
PA - Project A
JK - Jungle King

A : MAT Systems Inc, Denver, US
B : MAT Systems Inc, Coimbatore, India

<table>
<thead>
<tr>
<th>Source</th>
<th>Movie Selected</th>
<th>% Packet Loss</th>
<th>Average Delay (ms)</th>
<th>% Packet Loss</th>
<th>Average Delay (ms)</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A,B</td>
<td>SL</td>
<td>6</td>
<td>120</td>
<td>11</td>
<td>174</td>
<td>12.30.05</td>
</tr>
<tr>
<td>B,A</td>
<td>DH</td>
<td>4</td>
<td>93</td>
<td>11</td>
<td>150</td>
<td>15.30.20</td>
</tr>
<tr>
<td>A,B</td>
<td>DH</td>
<td>4</td>
<td>89</td>
<td>7</td>
<td>125</td>
<td>00.10.04</td>
</tr>
<tr>
<td>B,A</td>
<td>SL</td>
<td>0</td>
<td>113</td>
<td>4</td>
<td>167</td>
<td>03.25.30</td>
</tr>
<tr>
<td>A,B</td>
<td>JK</td>
<td>0</td>
<td>70</td>
<td>3</td>
<td>98</td>
<td>20.30.00</td>
</tr>
<tr>
<td>B,A</td>
<td>PA</td>
<td>0</td>
<td>72</td>
<td>3</td>
<td>107</td>
<td>23.22.30</td>
</tr>
</tbody>
</table>
Fig-5.10 Percentage of Packets Lost - Experiment-1
Fig-5.11 Play-out Delay - Experiment-1
Table 5.6: Packet Loss Statistics gathered from various routers – Experiment 2

<table>
<thead>
<tr>
<th>ROUTER</th>
<th>LOCATION</th>
<th>Best-Effort</th>
<th>Best-Effort</th>
<th>Me-TRAP-Q</th>
<th>Me-TRAP-Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>sl-gw24-ana-5-3-ts15.sprintlink.net</td>
<td>Anaheim</td>
<td>536</td>
<td>72</td>
<td>143</td>
<td>0</td>
</tr>
<tr>
<td>atl1-core1-10.atlas.algx.net</td>
<td>Atlanta</td>
<td>651</td>
<td>90</td>
<td>62</td>
<td>0</td>
</tr>
<tr>
<td>atl-core-02.inet.qwest.net</td>
<td>Atlanta</td>
<td>707</td>
<td>95</td>
<td>88</td>
<td>0</td>
</tr>
<tr>
<td>atl-brdr-03.inet.qwest.net</td>
<td>Atlanta</td>
<td>527</td>
<td>91</td>
<td>86</td>
<td>0</td>
</tr>
<tr>
<td>pos1-0-0-155m.ar1.bos1.gblx.net</td>
<td>Boston</td>
<td>784</td>
<td>20</td>
<td>65</td>
<td>1</td>
</tr>
<tr>
<td>chi-core-03.inet.qwest.net</td>
<td>Chicago</td>
<td>476</td>
<td>17</td>
<td>42</td>
<td>1</td>
</tr>
<tr>
<td>chi-edge-08.inet.qwest.net</td>
<td>Chicago</td>
<td>469</td>
<td>9</td>
<td>34</td>
<td>0</td>
</tr>
<tr>
<td>ord2-core1-10.atlas.algx.net</td>
<td>Chicago</td>
<td>476</td>
<td>35</td>
<td>33</td>
<td>4</td>
</tr>
<tr>
<td>loopback0.gw9.chi2.alter.net</td>
<td>Chicago</td>
<td>455</td>
<td>50</td>
<td>36</td>
<td>0</td>
</tr>
<tr>
<td>mail.rmail.mdsg-pacwest.com</td>
<td>Colorado</td>
<td>371</td>
<td>53</td>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>router.mitchell.edu</td>
<td>Connecticut</td>
<td>532</td>
<td>42</td>
<td>41</td>
<td>2</td>
</tr>
<tr>
<td>core-c7fe00.aspware.net</td>
<td>Dallas</td>
<td>496</td>
<td>73</td>
<td>45</td>
<td>4</td>
</tr>
<tr>
<td>dfw3-core1-10.atlas.algx.net</td>
<td>Dallas</td>
<td>616</td>
<td>89</td>
<td>42</td>
<td>8</td>
</tr>
<tr>
<td>denver-br2.bbnplanet.net</td>
<td>Denver</td>
<td>738</td>
<td>40</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>loopback0.gw2.den4.alter.net</td>
<td>Denver</td>
<td>539</td>
<td>21</td>
<td>77</td>
<td>0</td>
</tr>
<tr>
<td>router.vortechhosting.com</td>
<td>Florida</td>
<td>616</td>
<td>6</td>
<td>88</td>
<td>0</td>
</tr>
<tr>
<td>atm4009.c7507.bfs.fsu.edu</td>
<td>Florida</td>
<td>624</td>
<td>14</td>
<td>78</td>
<td>0</td>
</tr>
<tr>
<td>sagonet.net</td>
<td>Florida</td>
<td>672</td>
<td>10</td>
<td>84</td>
<td>0</td>
</tr>
</tbody>
</table>
Fig. 5.12 - Delay (ms) from various routers - Experiment-2
Fig-5.13 Packet Loss - Experiment-2

Bytes Lost

Best-Effort
Me-TRAP-Q
DiffServ

% Bytes Lost

Locations A B C D E F G H

Fig-5.13 Packet Loss – Experiment-2