Implementation and Performance Evaluation of Tunneled Aggregated RSVP Architecture

Karthikeyan T Nathillvar

Masters Thesis Defense
University of Kansas, Lawrence
February 14th 2003

Committee:
Dr. Douglas Niehaus (Chair)
Dr. Victor Frost
Dr. Jerry James

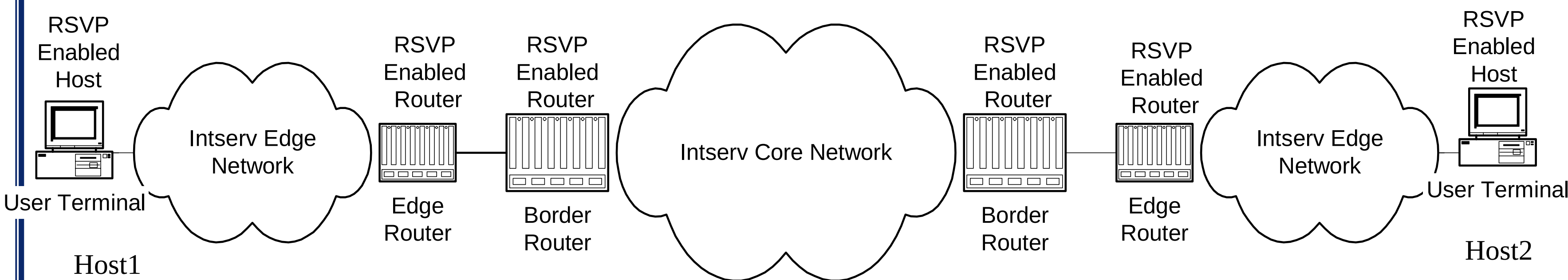
Contents

- Introduction
- Related Work
- Tunneled Model – Design, Implementation
- Performance Evaluation and Comparison
- Management Complexity Analysis
- Conclusion

Introduction – Problem

- Today's trend in Internet - Converged Networking
  - Voice, Data and Video are transmitted over common IP infrastructure
  - Different applications demand different service guarantees
  - Internet offers just best-effort QoS
- Current QoS Models - Drawbacks
  - Integrated Services – Not Scalable
  - Differentiated Services – No Signaling Mechanism
- No means to measure the Management Complexity

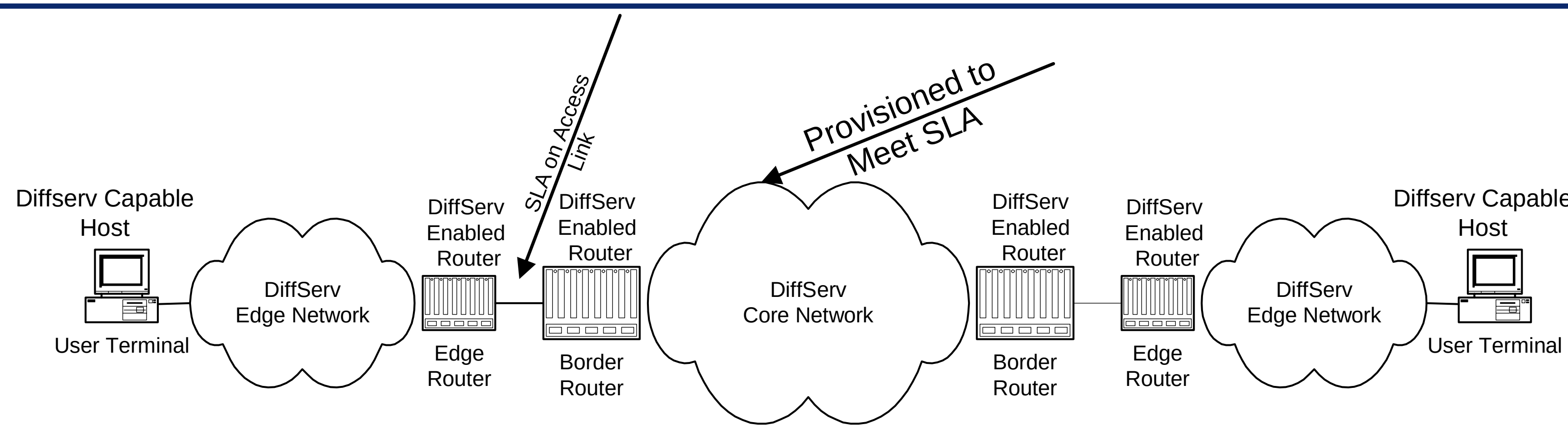
Introduction – Integrated Services



Intserv Internetwork

- Approach – QoS on Microflow basis
- Components
  - Signaling Protocol – ReSerVation Protocol
  - Services
    - Controlled Load Service – Ex. Adaptive real time applications
    - Guaranteed Load Service – Ex. Audio and Video playback applications.
- Disadvantage – Scalability, sparse RSVP deployment, etc

Introduction – Differentiated Services

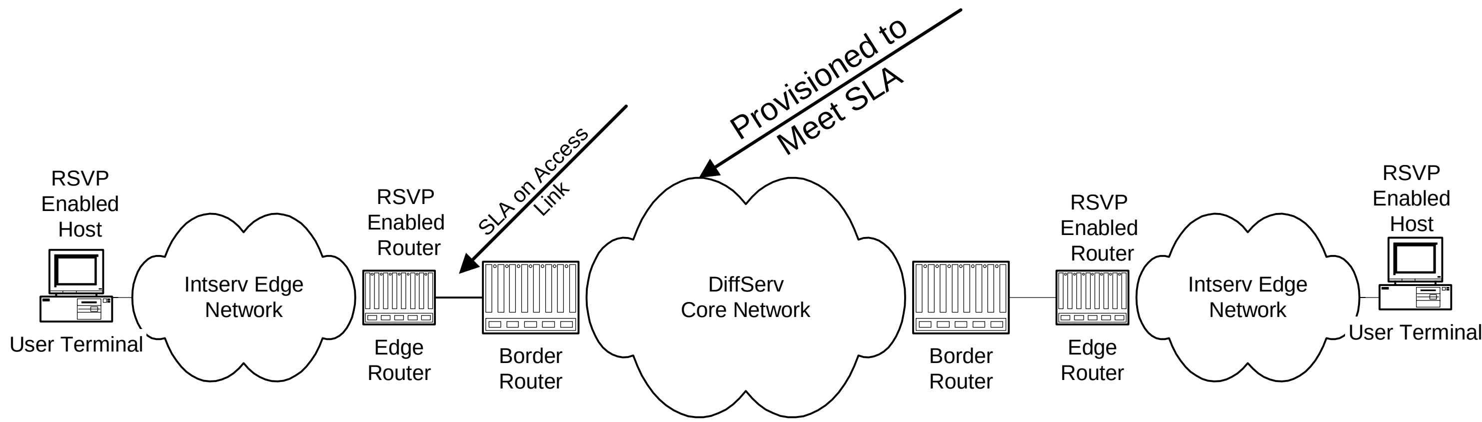


DiffServ Internetwork

- Class of Service – Based on ToS byte in IP
- Components
  - SLA – Exchanged between customer and provider
  - Assured forwarding and Expedited forwarding PHB
- Disadvantages – Static Configuration, no QoS protection, etc

TOS	CLASS
001	AF1
010	AF2
011	AF3
100	AF4
101110	EF

Introduction - Hybrid Models



Hybrid Intser-Diffserv Internetwork

- Diffserv considered a node in the Intserv network
- Intserv is a customer of the Diffserv network
  - SLA exchanged between the edge and core network
- Models
  - Microsoft model – Static aggregate resource allocation in Diffserv
  - Tunneled Model – Dynamic aggregate resource allocation in Diffserv
  - Per-flow Model – Per-flow dynamic resource allocation in Diffserv

Introduction - Goals

Goals:

- Design and implement Tunneled Aggregated RSVP architecture
- Evaluate data plane and control plane performance against Intserv and Microsoft QoS models
- Management complexity comparison of the tunneled model with several other QoS models.

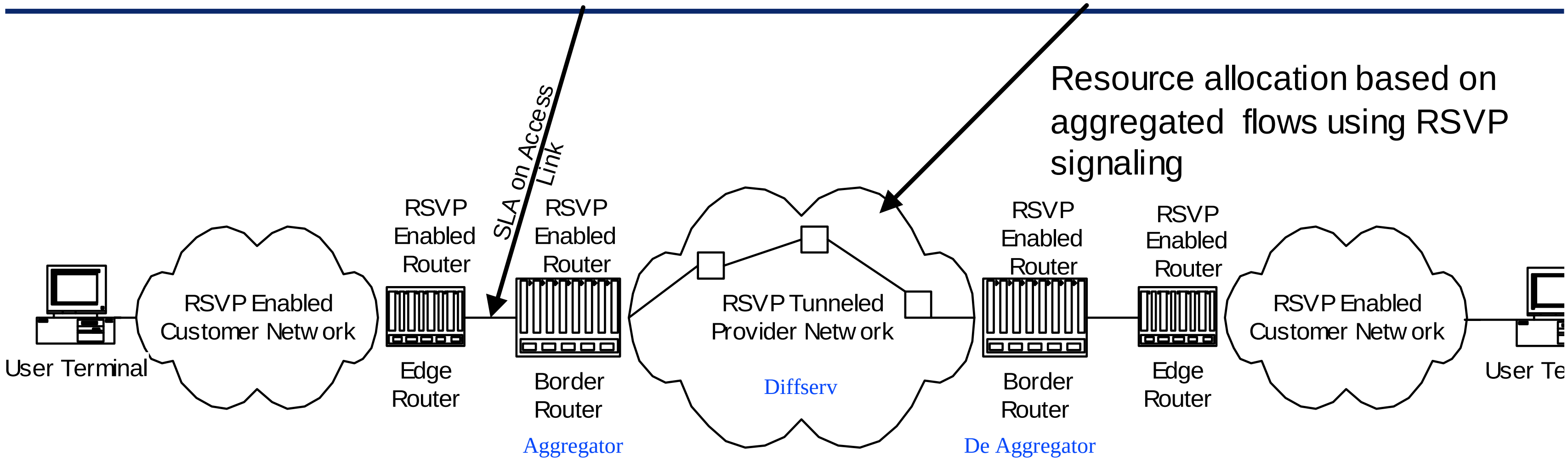
Related Work

- Integrated Services
  - RSVP – Signaling software developed by ISI
  - Traffic Control Layer – Alexy Kuznetsov
  - No complexity and performance analysis
- Differentiated Services
  - Ecole Polytechnique Federale de Lausanne (EPFL) – Provides kernel patch for 2.2.14 kernels
  - No data plane analysis
- Tunneled RSVP architecture was built over these softwares.

Related Work

- Ericsson Prototype
  - Studies the requirements involved in designing the boundary routers
  - Microsoft hybrid model – Non-optimal network utilization.
  - Scalability issues were not considered
- Institute of Computer Communication and Applications (ICA) Prototype
  - All Intserv flows are mapped to a single EF instance
  - Restricted mapping choice
- Tampere University of Technology(TUT) Prototype
  - Similar to ICA prototype.
  - Provides complete data plane performance results
- Scalability issues are not considered
- No Complexity comparison

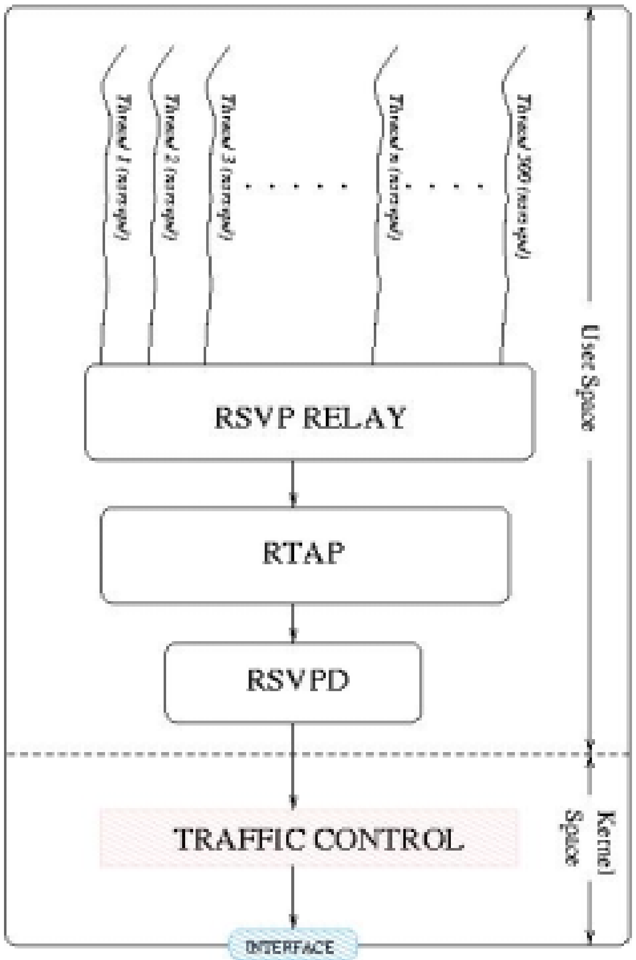
Tunneled model - Architecture



RSVP/RSVP Aware DiffServ Internetwork

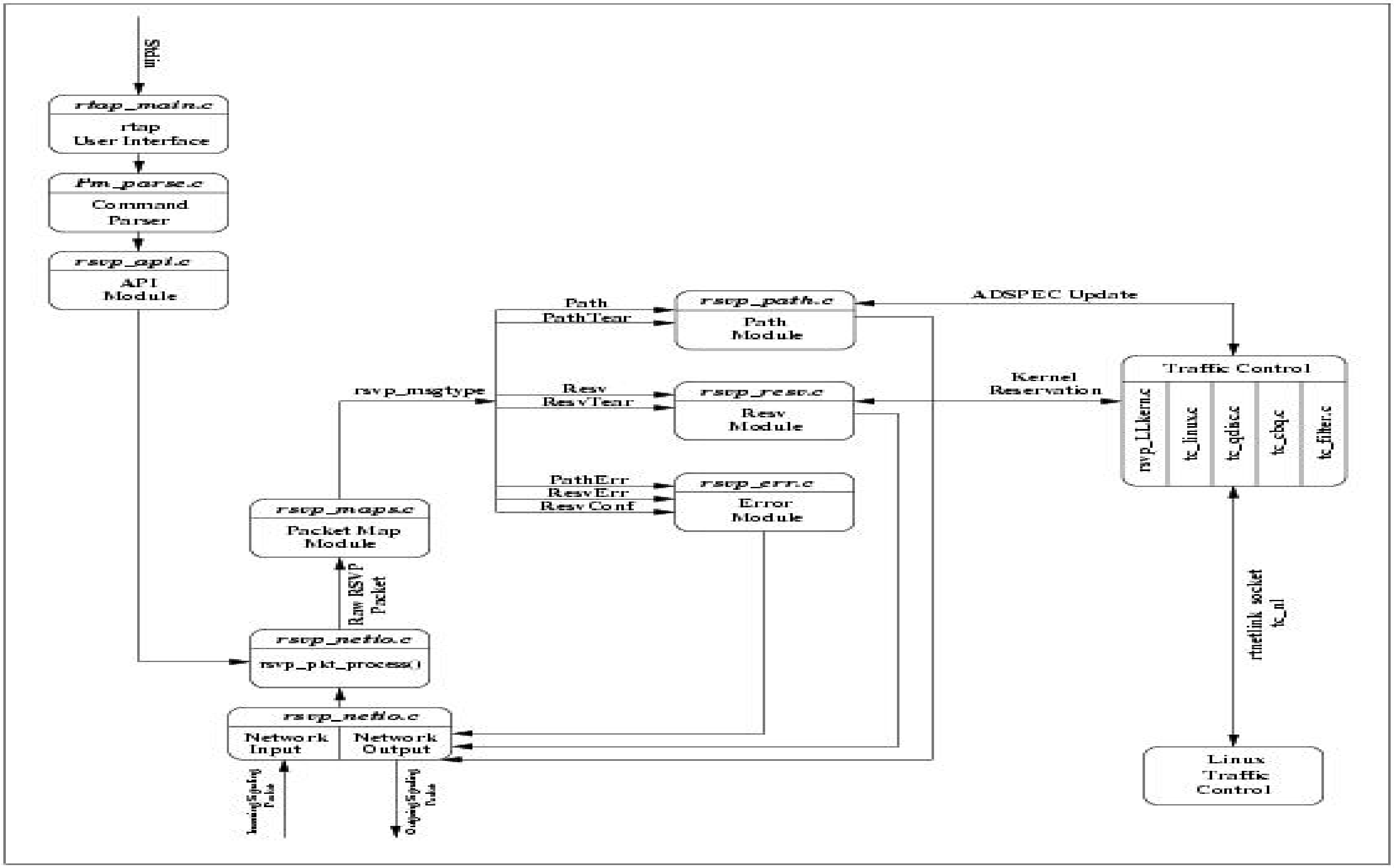
- Diffserv remarking is done at the aggregator
- Admission control and service mapping is done at the deaggregator
- Mapping: Intserv Controlled Load Service to Diffserv AF PHB
- Reservation is done on aggregate basis

Design and Implementation – Edge Router



- Pure Intserv operation
- Hosts are simulated by threads
- NetSpec is used for call generation and traffic generation
- Capable of generating varying Intserv traffic requests at varying call arrival rates

RSVP Process – Modular View



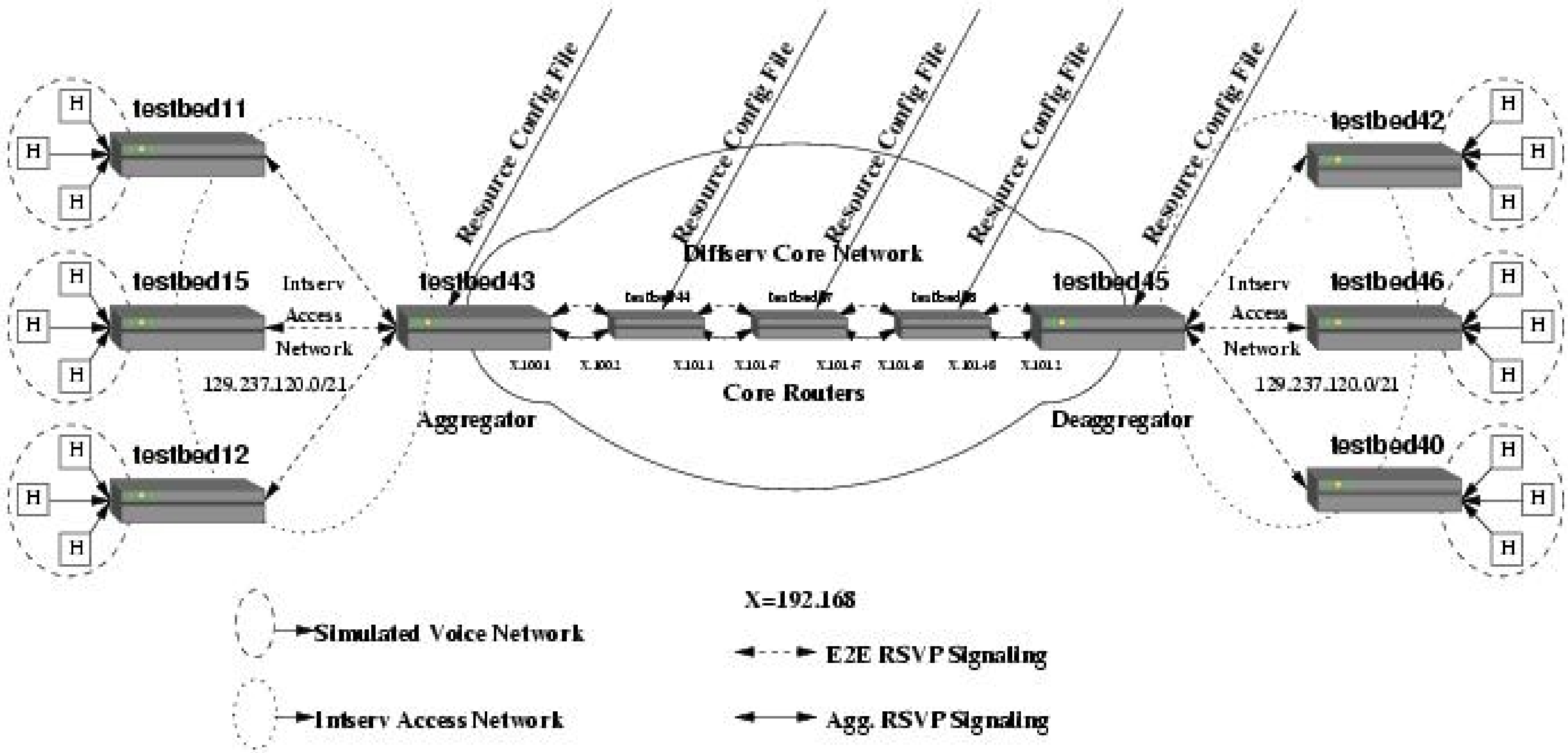
Design and Implementation – Aggregator

- Path Module
  - Installs the end-to-end PSB
  - Tunnels the Path message across the Diffserv region
  - Receives the aggregate PathErr message from the Deaggregator
  - Installs an aggregate API session by contacting the API module
- Resv Module
  - Processes end-to-end and aggregate reservation messages
  - Orders the traffic control module to establish the end-to-end filter for steering the traffic into the AF instance
  - Orders the traffic control module to modify the AF instance parameters upon receiving an aggregate reservation message

Design and Implementation - Deaggregator

- Path Module
  - Triggers aggregate PathErr messages upon the arrival of the tunneled end-to-end path message
  - Orders the API module to install the aggregate session
  - Calculates the minimum resource availability inside the Diffserv region
- Resv Module
  - Does End-to-end reservation→AF instance mapping
  - Performs aggregate admission control
  - Triggers aggregate reservation messages based upon the hysteresis policy

Tunneled Model – Emulated Network



- Hysteresis policy for triggering aggregate reservations.
- Aggregate signaling

Performance Evaluation - Topology

- Topology
  - Number of hosts per enterprise network = 256. [Limited by number of active threads]
  - Number of Diffserv routers (Varied from 3-7)
  - Number of border routers = 2 [Depends upon the network topology]
  - Access link bandwidth
    - 100Mbps for control plane & 10Mbps for data plane.
  - Core link bandwidth
    - 100Mbps for control plane & 10Mbps for data plane.
  - Routers – Emulated on high speed Pentium III, 1GHz, 1GB RAM Linux systems.
  - Voice Gateways – Emulated on low speed Pentium II, 400MHz, 128MB RAM Linux systems.

Performance Evaluation – Parameters

- Traffic Pattern
  - Call inter-arrival time exponentially distributed.
  - Average call arrival rate is varied from 5 calls/second to 50 calls/second.
  - Call duration (constant) = 5seconds.
- Diffserv Parameters
  - Real time class divided into 4 AF instances, each with a bandwidth of 20Mbps.
  - Best Effort class bandwidth=20Mbps.
  - RED Parameters = 60KB/15KB/45KB (limit/min/max).
- Intserv Parameters
  - Bandwidth assigned to real-time class = 80Mbps.
  - Bandwidth assigned to best-effort class = 20Mbps.
  - Served by CBQ queuing discipline.
- RSVP Daemon parameters
  - Refresh Interval (REFRESH_DEFAULT) – 30seconds.
  - Number of controlled load flows – 1024.

Performance Evaluation – Parameters

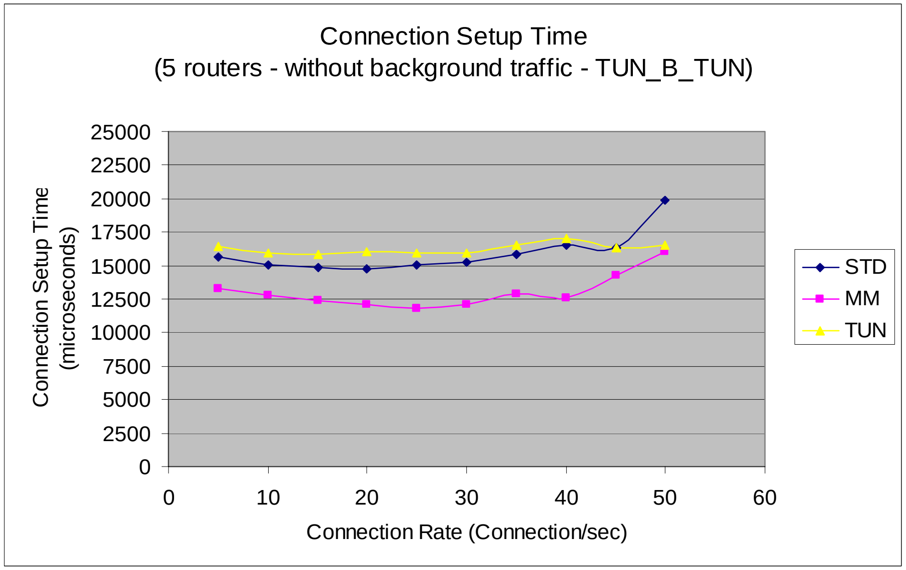
- RSVP Daemon Parameters
  - Maximum number of API sessions (MAX_RAPI_SESS) = 1024.
  - Maximum number of elements in the timer queue (MAX_TIMER_Q) = 1024.
  - Maximum number of active sessions (MAX_SESSIONS) = 1024.
  - Maximum number of flow descriptors in a packet (MAX_FLWDS) = 1024.
- Hysteresis Parameters – Only in Tunneled Model
  - Increment percentage (hyst_incr) = 15%.
  - Increment difference (hyst_add_diff) = 10%.
  - Decrement percentage (hyst_decr) = 10%.
  - Decrement difference (hyst_sub_diff) =15%.
- Miscellaneous Parameters
  - b/r ratio – AF instance mapping.

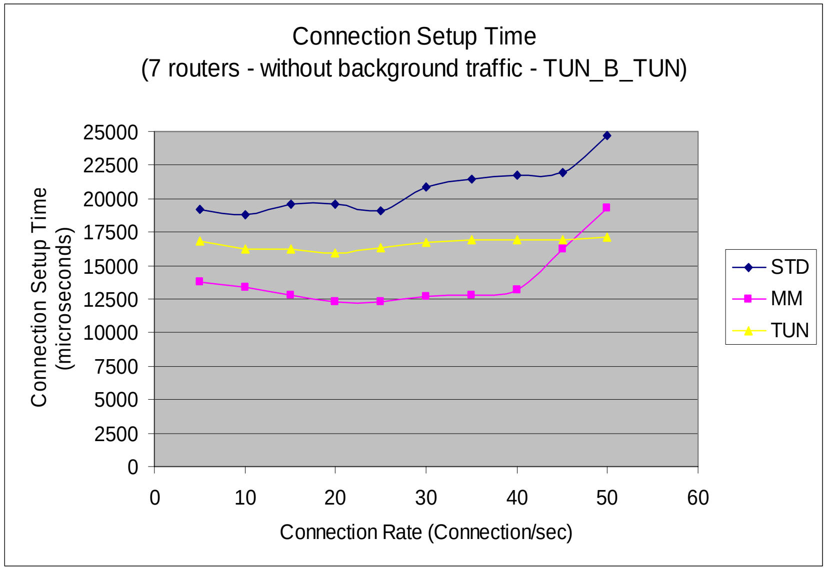
b/r ratio (r)	**AF Instance**
r < 1ms	AF1
1ms <= r < 2ms	AF2
2ms <= r < 3ms	AF3
3ms <= r < 4ms	AF4

Performance Evaluation - Metrics

Data Plane	•Throughput •Delay
Control Plane	•Connection Setup Time (CST) •CPU utilization •Memory utilization
Management Plane	Management complexity, sum of •Static configuration complexity •Dynamic configuration complexity •Messaging complexity •Tuning complexity

Control Plane Test Results – Connection Setup Time





- STD_CST > TUN_CST > MS_CST
- Tunneled Model – Higher system overhead at aggregator and de-aggregator.
- Microsoft Model – No signaling overhead in core routers.
- Both tunneled and Microsoft models scale well with increase in number of routers.

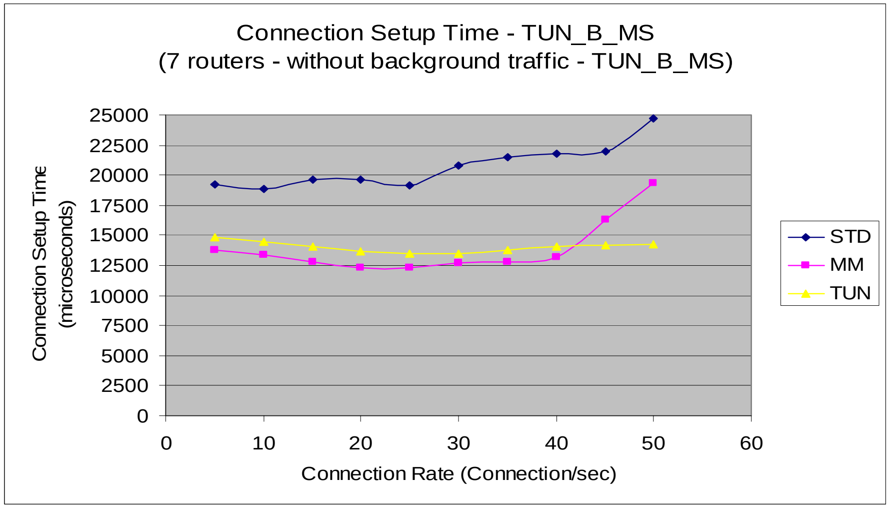
Tunneled Model – Various Behaviors

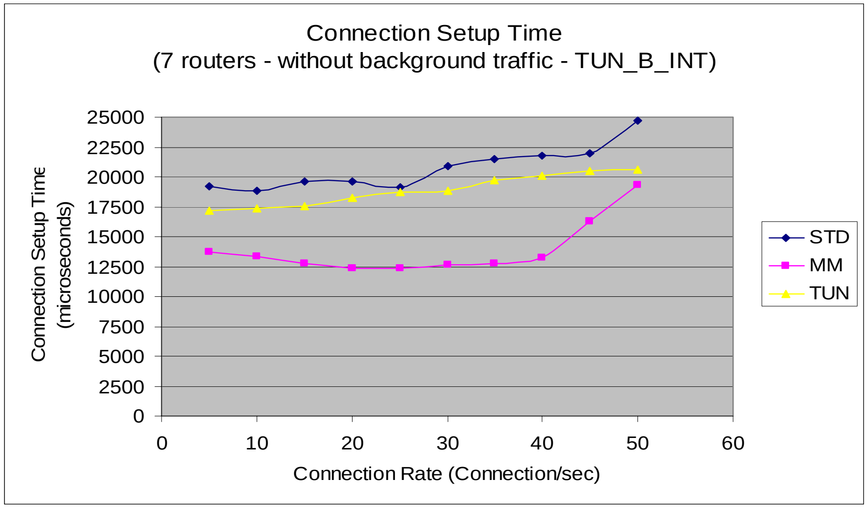
- Motivation
  - Presence of tunable parameters
    - Minimum bandwidth allocated to AF instances to support aggregate reservation.
    - Hysteresis policy parameters.
  - Parameters can be tuned to enable tunneled model to emulate
    - Microsoft model.
    - Intserv model.
- Microsoft model and Intserv model represent two extremes of tunneled model.
- Major advantage – Flexibility
  - Carrier can choose any of the three QoS models with just a single implementation.

Tunneled Model – Various Behaviors

- Tunneled Model Emulating Microsoft model (TUN_B_MS)
  - Minimum reservation at the core exceeds the sum of end-to-end reservations.
  - Aggregate reservation messages are never triggered.
  - Hysteresis policy – Never comes into effect as minimum reservation is always greater than sum of end-to-end reservations.
- Tunneled Model Emulating Intserv Model (TUN_B_INT)
  - Minimum reservation is set to a value equal to token bucket rate requested by single connection.
  - Hysteresis policy - Every end-to-end reservation request and tear down causes triggering of aggregate reservation messages by the deaggregator.

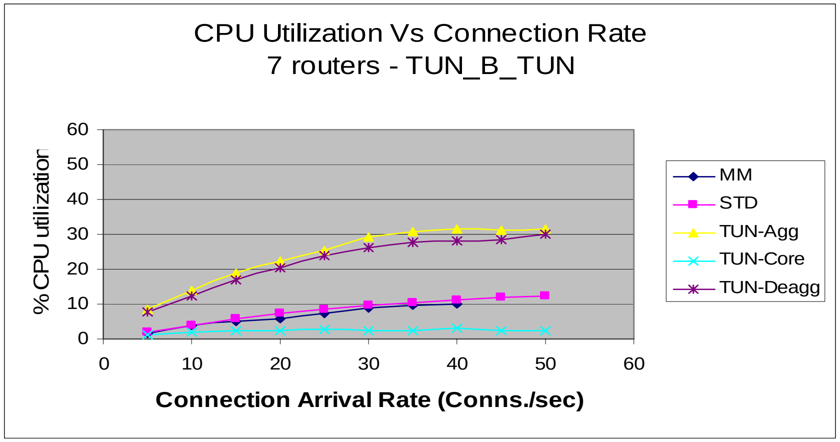
Control Plane Test Results – CST – Other Possible behaviors of Tunneled Model

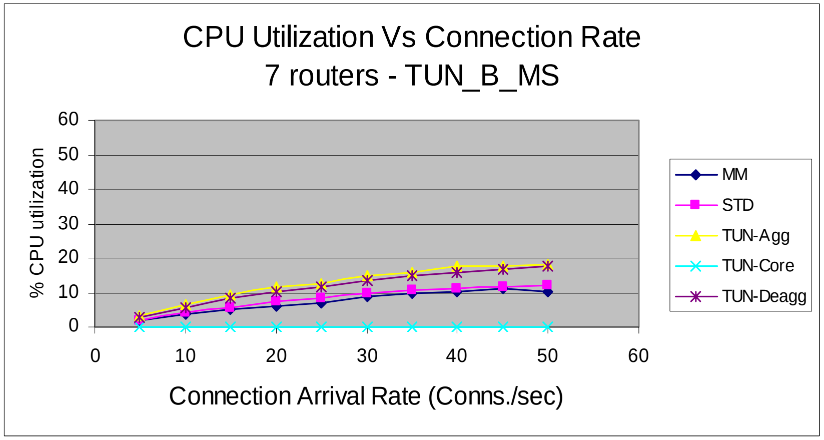


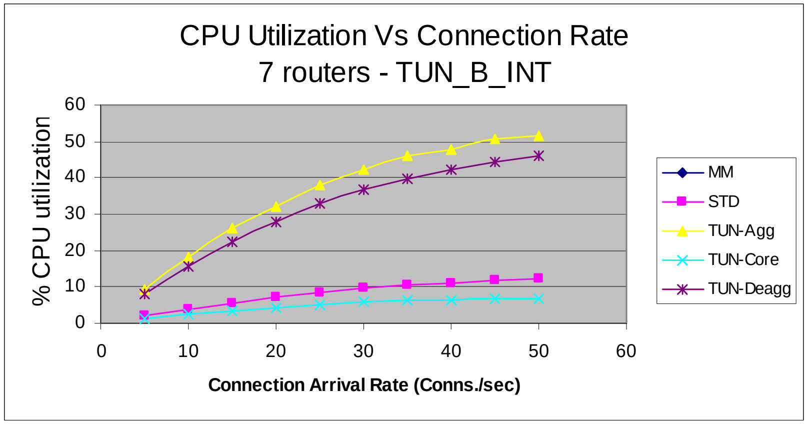


- TUN_BEHAVE_MS – Tunneled model emulating Microsoft model – CST curve follows closely the CST curve in pure Microsoft model.
- TUN_BEHAVE_INT – Tunneled model emulating Intserv model – CST curve follows closely the CST curve in pure Intserv model. To achieve this aggregator and deaggregator consume extra CPU.
- Both these behaviors introduce less overhead at the core routers.

Control Plane Test Results – CPU Utilization







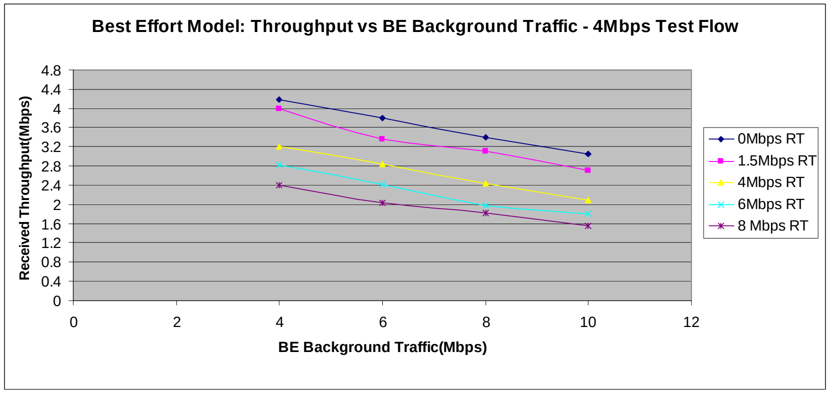
- Negligible CPU Utilization at core in both Tunneled and Microsoft Model.
- TUN_CPU_AGG > STD_CPU_AGG, MS_CPU_AGG
- TUN_CPU_AGG and TUN_CPU_DEAGG in Tunneled model emulating Intserv model is higher than same in any other models.
  - Aggregate Session Maintenance.
  - Hysteresis Policy and Admission Control.
  - Renegotiation for every end-to-end reservation request and teardown.

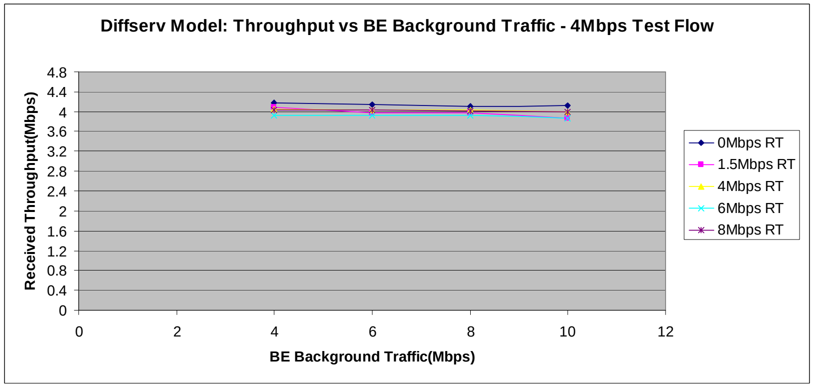
Control Plane Test Results – Memory Utilization

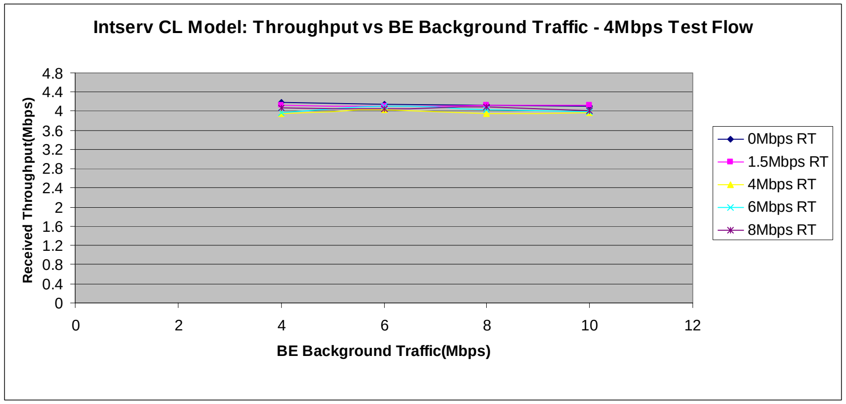
Router	Maximum Memory Utilization (MB)		
	TUN_B_TUN	TUN_B_INT	TUN_B_MS
Intserv Router	1.72	1.72	1.72
MS Aggregator	1.78	1.78	1.78
Tunneled Model Aggregator	3.37	3.86	3.10
Tunneled Model Core Router	2.05	2.15	1.37
Tunneled Model Deaggregator	3.39	3.39	9.87

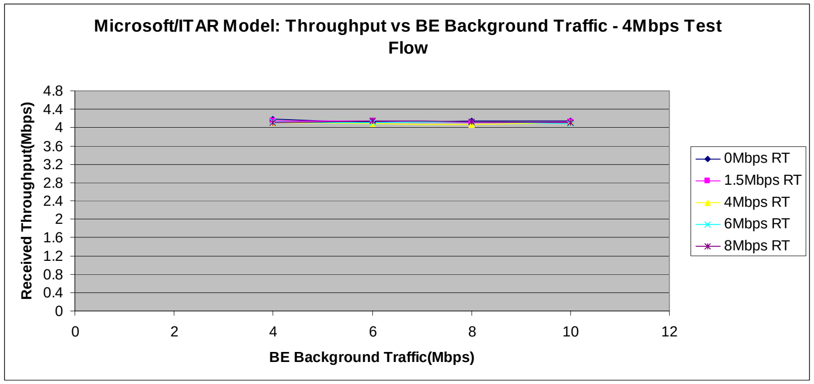
- De-aggregator is the point of interest.
- Memory Utilization at Tunneled Model's De-aggregator is higher than that of any other routers in other models.
  - Extra state information for Aggregate Session, Hysteresis Policy, etc.
  - Measured using a Pentium III, 1GHZ, 1GB system.
- Overall kernel memory utilization follows same pattern.

Data Plane Performance – Throughput Comparison

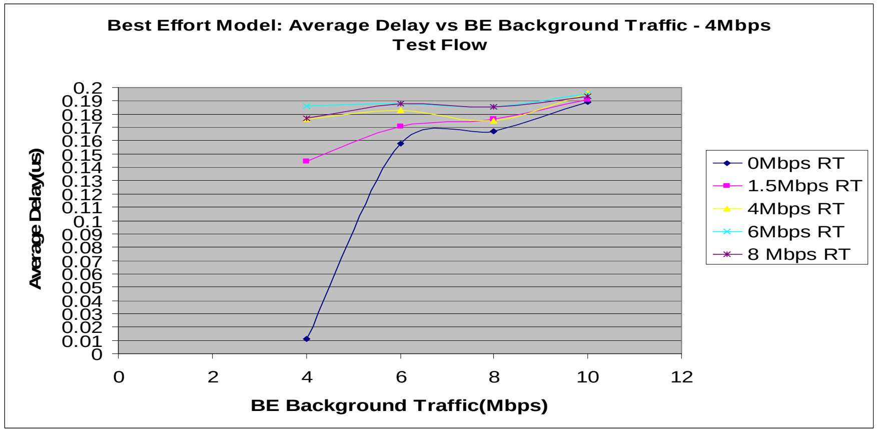


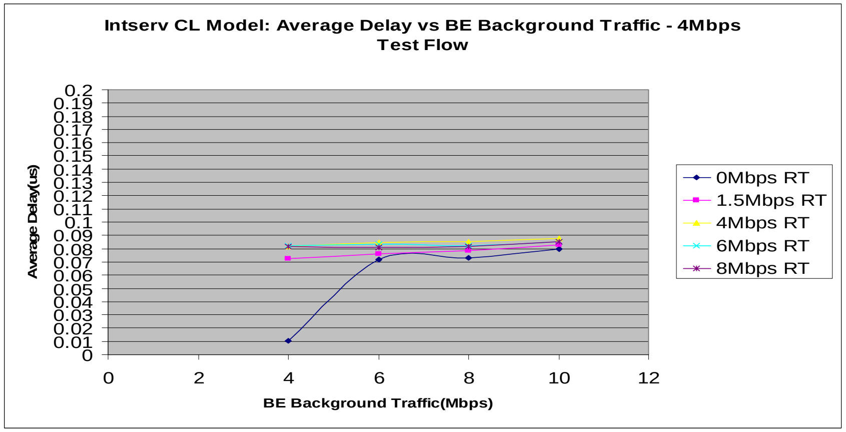


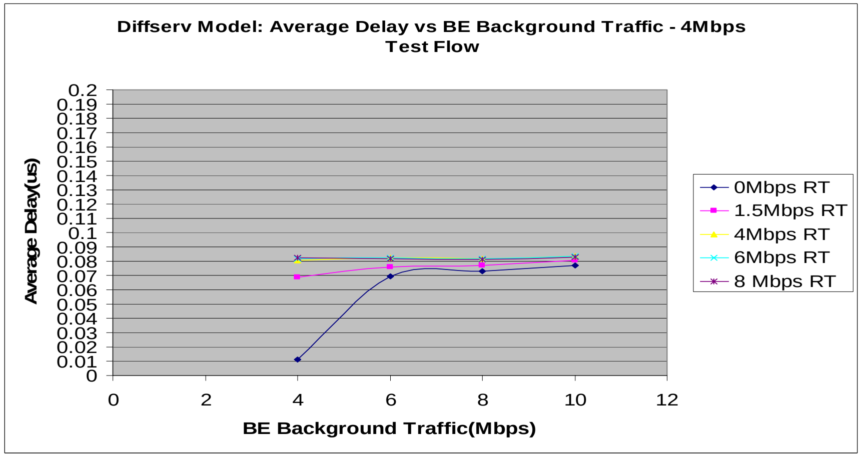


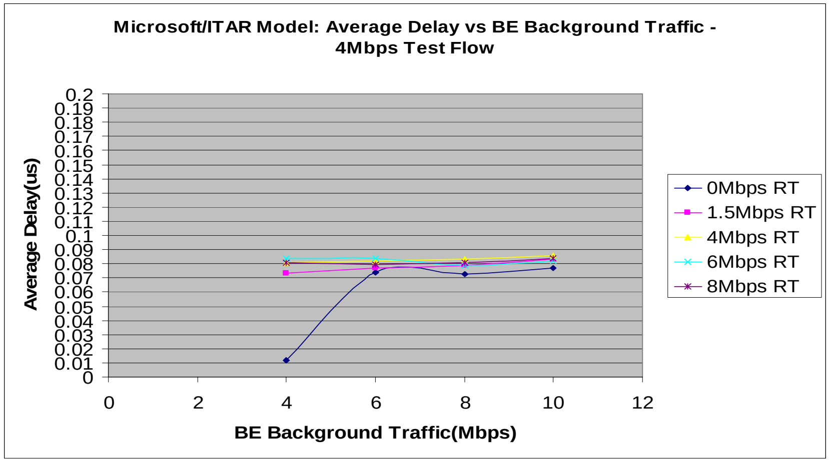


Data Plane Performance – Delay Comparison









Performance Evaluation – Results

- Hybridization does not result in performance deterioration.
- Throughput Comparison – Data flow's performance is similar in all QoS architectures.
- Delay Comparison – Data flow exhibits similar performance in all QoS architectures.
- Results are based only on empirical measurements.

Management Complexity - Introduction

- Managing a QoS network (Management Complexity) involves:
  - Establishing appropriate traffic control structure at all routers.
  - Choosing the values for various tuning parameters – Very important as they affect the network performance.
  - Verifying the established reservation:
    - Done by observing the traffic control filters established at all routers.
  - Looking for any malfunctions.
    - Done by observing the signaling messages exchanged in various architectures.
- Architecture of choice – Obtained by observing complexity comparison between different QoS models.

Management Complexity - Components

Management Complexity	Issues Touched	Equation
Static Management Complexity M_s	• Training complexity • Testing and debugging	$M_s = \sum W_{si} M_{si}$
Dynamic Management Complexity M_d	• Testing and debugging • Kernel message processing	$M_d = \sum W_{di} M_{di}$
Messaging Complexity M_m	• Monitoring the messages exchanged during the entire duration of the connection	$M_m = \sum W_{mi} M_{mi}$
Tuning Complexity M_t	• Network sensitivity • Network stability	$M_t = \sum W_{ti} M_{ti}$

Management Complexity - Methodology

- **Methodology Summary**
  - Identifying complexity metrics.
  - Evaluated as a function of two network attributes – Size and Number of active connections.
  - Assigning weights to individual metrics.
  - Mathematical model for complexity analysis.
- **Steps Involved**
  1. Determine the tuning parameter values – Measure the tuning complexity.
  2. Emulate the network – Measure the static configuration complexity.
  3. Establish QoS connections.
     1. Observe the debug messages, filter establishment, AF instance modification from the emulation.
     2. Measure the dynamic configuration and messaging complexity from the emulation.
  4. Assign weights to different complexity metrics based on their importance to carrier.
  5. Determine the overall complexity.

Management Complexity – Tunneled Model Analysis

- Analysis for Tunneled Model Emulating Microsoft Model.
  - Static Configuration Complexity = $M_{ts} = 408g + 234r$
    - 408 – Number of parameters present in network configuration scripts + number of parameters present in traffic control scripts required for initial configuration at border routers.
    - r = Number of Diffserv core routers.
    - 234 – Total number of lines of configuration scripts required for initial configuration at Diffserv core routers.
  - Messaging Complexity = $M_{tm} = (4t+2)*2*g*n + ((n/\lambda + t)*8*[r+2*g])$
    - 4 Represents No. Agg. PathErr messages exchanged initially.
    - t =Average connection duration in minutes.
    - g = number of aggregators – de-aggregator pairs.
    - r = number of core routers
    - n = number of connections
    - λ = Connection Arrival Rate

Management Complexity – Tunneled Model Analysis

- Dynamic Configuration Complexity = $M_{td} = (7*2)g*n$
  - 7 – Number of AF instance parameter modifications when renegotiation is triggered.
- Tuning Complexity = $M_{tt} = 35$
  - 35 – Number of tuning parameters in tunneled model.

- Assigning Weights
  - Depends on the network administrator.
  - Constraint: $\sum w_i = 1$
  - Lowest weights (0.0125 and 0.0125) to messaging and dynamic configuration complexity – requires no interference or training.
  - Highest weight (0.95) to tuning parameters – Tuning parameters affect the overall network performance.
  - Medium weight (0.025) to static configuration complexity – involves training of personnel, etc.

- Overall Complexity = $M_{to} = w_{ts}*M_{ts} + w_{td}*M_{td} + w_{tm}*M_{tm} + w_{tt}*M_{tt}$

$$= 0.025(408+234r) + 0.0125*(4t+2)*2*g*n + ((n/\lambda + t)*8*[r+2*g])$$
$$+ 0.0125((7*2)g)*n + 35*0.95$$

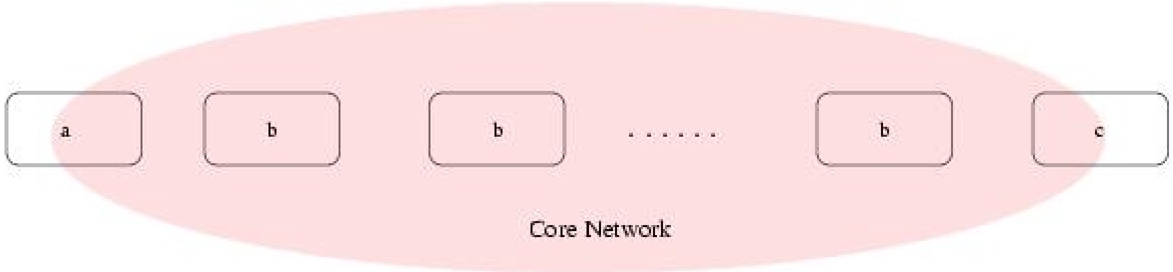
Management Complexity – Equation Chart

Parameters	Diffserv		Intserv		Microsoft Model	
	w	Value	w	Value	w	Value
Static Management Complexity	0.025	$28g+184r$	0.025	$49r$	0.025	$358g+184r$
Dynamic Management Complexity	0.0125	0	0.0125	$7r*n$	0.0125	$(7*2)g*n$
Messaging Complexity	0.0125	5*n	0.0125	$(4t+2)rn$	0.0125	$(4t+2)g*2*n$
Tuning Complexity	0.95	10	0.95	14	0.95	35

Management Complexity – Equation Chart

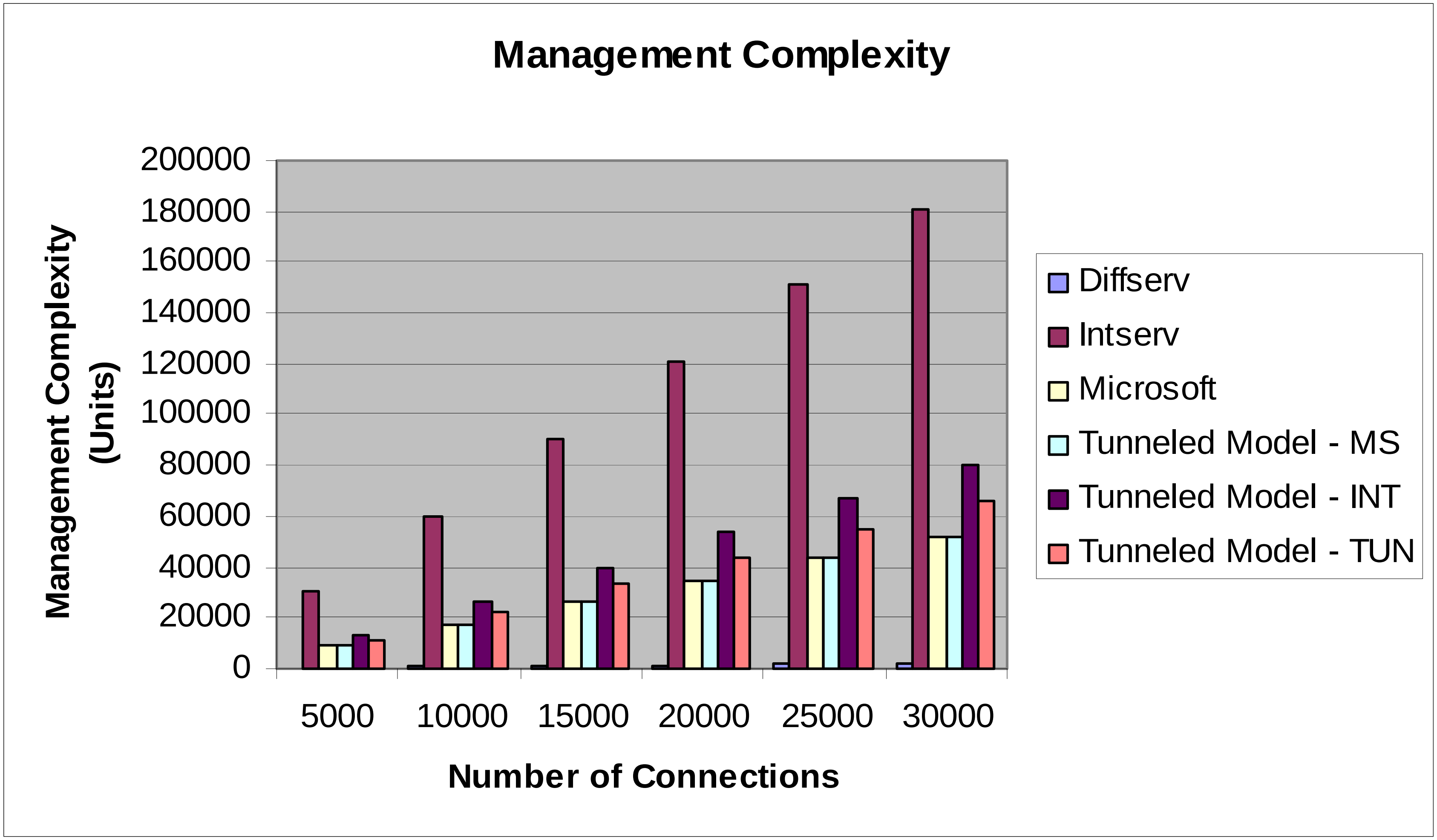
Parameters	Tunneled Model Emulating Microsoft Model		Tunneled Model Emulating Intserv Model		Tunneled Model Acting As A Pure Tunneled Model	
	w	Value	w	Value	w	Value
Static Management Complexity	0.025	$408g+234r$	0.025	$408g+234r$	0.025	$408g+234r$
Dynamic Management Complexity	0.0125	$(7*2)g*n$	0.0125	$(7*2)*g*n$ $+5*(g+r)*2*n$	0.0125	$(7*2)*g*n$ $+5*(g+r)*((x/100)*2*n)$
Messaging Complexity	0.0125	$(4t+2)*2*g*n$ $+((n/\lambda+t)*8$ $*[r+2*g])$	0.0125	$(4t+2)*2*g*n$ $+(n/\lambda+t)*8*[r+2*g]$ $+2*n*[r+2*g]$	0.0125	$(4t+2)*2*g*n$ $+(n/\lambda+t)*8*[r+2*g]$ $+((x/100)*(2*g+r)*2*n)$
Tuning Complexity	0.95	35	0.95	35	0.95	35

Management Complexity – Comparison



- **Topology**
  - Derived from Cisco's Intserv/Diffserv integration model.
- **Complexity evaluation**
  - Varying number of connections.
  - Fixed number of core routers (5).
  - One aggregator-deaggregator pair.
- **Tunneled Model Behaviors**
  - Microsoft Model.
  - Intserv Model.
  - Tunneled Model – 50% renegotiations.
- **Evaluated for connection duration of 15 minutes and arrival rate of 50 connections/minute.**

Management Complexity - Results



Management Complexity – Analysis

- Diffserv - Least complex to manage.
- Intserv - Most complex to manage due to heavy exchange of signaling messages by the routers.
- Pure Microsoft model and emulated Microsoft model have equal management complexity.
- Management complexity of the tunneled models lie in between Diffserv and Intserv models.
  - Processing of only aggregate signaling messages by the Diffserv core routers.
  - Complex traffic control structure at the aggregator and core.
- Complexity results purely depend upon the assigned weights.

Conclusion

- Tunneled Aggregated RSVP QoS architecture was successfully implemented.
- Control plane complexity comparison was done
  - Tunneled model introduces marginal overhead in border routers.
  - At a carrier's core Tunneled model introduces negligible amount of system overhead when compared to Intserv model.
  - Tunneled model scales well with increase in number of core routers.
- Data Plane Performance – Hybridization does not result in performance deterioration
- Management Complexity
  - Tunneled model's management complexity lies in between conventional Diffserv and Intserv models.
  - Tunneled model – Most flexible model. Can behave like Microsoft and Intserv model.
- Tunneled model is the career's choice for providing flexible and scalable QoS.

Thank You