1999

An Adaptable Network Architecture for Multimedia Traffic Management and Control

Sheng-Yih Wang

Bharat Bhargava
Purdue University, bb@cs.purdue.edu

Report Number:
99-048

AN ADAPTABLE NETWORK ARCHITECTURE FOR MULTIMEDIA TRAFFIC MANAGEMENT AND CONTROL

Sheng-Yih Wang
Bharat Bhargava

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD TR #99-048
December 1999

Technical Report CS-99-048
An Adaptable Network Architecture for Multimedia Traffic Management and Control*

Sheng-Yih Wang and Bharat Bhargava
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
E-mail: {swang,bb}@cs.purdue.edu

December 1999

1 Introduction

Quality of Service (QoS) provision for emerging applications is a very hot research topics in recent years. QoS can be provided at different layers [10]. The application layer and the network layer are the two major layers which QoS parameters and QoS control mechanisms are focused. In network layer QoS provision, Integrated Services (Intserv) [3] and Differentiated Services (Diffserv) [2] are two paradigms which try to provide Quality of Service (QoS) to IP-based networks such as the current Internet.

Integrated Services is a reservation-based QoS control mechanism. It use RSVP, a resource reservation protocol, to reserve resources at each router along the traffic path. The objective of Interserv is to provide customized QoS to each individual traffic flow. In contrast, the existing IP networks do not provide any mechanism to support customized QoS for traffic flows. Intserv paradigm is known to have several drawbacks. Scalability to large numbers of flows and complexity in implementations are the two most serious drawbacks of Intserv approach.

Differentiated Services paradigm is a new effort to provide QoS support in IP networks. Diffserv approaches try to eliminate the need of RSVP-style resource reservation by aggregating traffic flows with similar QoS requirements into one single traffic class. The pakcets belonging to the same aggregated traffic class are marked using the same DS value in the IP header at the network boundary. The packets will be forwarded according to the per-hop behaviors (PHB) defined for this particular traffic class. Currently there are three standard PHB: the default PHB, the Expedited Forwarding (EF) and the Assured Forwarding (AF).

*This research is partly supported by a grant from NSF under NCR-9405931

One of the drawback of the Diffserv approaches is that the QoS parameters can not be dynamically changed. Since the QoS in Diffserv is specified by the Service Level Specification (SLS) (which is a subset of the Service Level Agreement (SLA) that provides the technical pecification of the service) using conventional communication mechanisms such as FAX or telephone, it is not easy to re-negotiate and adapt when communication requirements change dynamically. Another drawback of Intserv (and Diffserv) is that the QoS provided is the network-level QoS. They can not provide application-level QoS. The QoS parameters in both approaches are specified by network performance parameters such as bandwidth, delay, etc. However, sometime the application QoS parameters are very different from the network QoS parameters. For example, a video application may want to have a smooth playback (constant frame rates) no matter what the conditions the network are in.

Active network [5] is one promising paradigm for providing customized network service to the applications. Active networks allow application programs to inject specific programs to any intermediate nodes (routers). Active networks provide the flexibility for the application program to modify the default services a router can provide to suit its specific needs. Therefore it has the potential to provide the application-level QoS at the transport or network layers. Our view is that active network techniques can be used as a mechanism to extend the Diffserv techniques in providing application-level QoS. In addition, active network techniques can be used to provide dynamic re-negotiation in Diffserv approach.

In this report, we design an adaptable network architecture, called ADNET, which allows all of the active traffic, Intserv traffic, and Diffserv traffic to co-exist. Our long-term goal is to unify all the three paradigms together to provide a wide range of network services to all the users and meet their specific needs.

2 Design Goals and Requirements

- **Adaptable:** the adaptability features in ADNET includes: adaptable network services and adaptable applications. ADNET allow the application to adapt to the resource constraints. The resources available to an application are subject to maximum values for each resource type. For applications which don't make reservation in advance, the traffics generated will be aggregated into a specific class based on its attributes such as its DS label. The aggregated traffic classes will be subject to traffic management similar to Diffserv. ADNET also allow the application to choose different levels of security.

- **Safe:** ADNET explicitly include the resource management module. In contrast, SANE [1] left the resource management problem untouched, therefore can not provide true safe execution.

- **Secure:** ADNET employ the standardized Secure IP Protocol (IPSEC) [9] to ensure the secure communications. The security requirement is an adaptable

feature in our design. The security requirement can be specified as a QoS parameter when setting up reservations.

- **Efficient:** The efficiency in ADNET comes in two flavors: short-cut path for non-active traffics, primitive operations for multimedia data to facilitate the manipulation of the payload inside the packets.

- **Scalable:** Our design address both the customized aspect and the aggregated aspect of network services, therefore is scalable to very large network.

- **Interoperatable:** For applications which are based on the current best-effort paradigm, our design provides seamless interoperatability so they can run without any change.

3 Architecture Design

3.1 Overview

ADNET is inspired and partially derived from the work in Integrated Services [3], Differentiated Services [2], and Active Networks [5]. Figure 1 shows the architecture of ADNET. Our design focus on the explicit management of resources for classified traffic flows. Our notion of classified traffic flow is a generalization of the usual sense of the traffic flow which encompass a various of aggregated flows. ADNET is comprised of four major modules: the *Input Interface*, the *Active Execution Environment (AEE)*, the *Output Scheduler* and the *Policy Database*. The Input Interface receives packets from the network and perform classification and shaping for the traffics. Active Execution Environment is composed of a virtual machine and a CPU scheduler which provides an execution environment for active network traffic. Output scheduler perform output bandwidth allocation to fairly share the output bandwidth among different types of classified traffic flows. Policy database provide policy information to other major components and interact with AEE to update the policies.

We choose to keep a per-classified-flow state to enforce various policies we want to impose on the traffic. We argue that per-classified-flow state is a necessary evil for future networks because it will enhance the security and accountability of future networks. For each classified traffic flow, a resource limitation tuple is set up to limit the usage of various resources in the router. In our current design, a resource limitation tuple is consist of maximum input bandwidth, maximum output bandwidth, and maximum CPU time allowed.

Our design is IP-based. On top of the IP, we introduce a transport layer protocol called ACTP for active traffic.

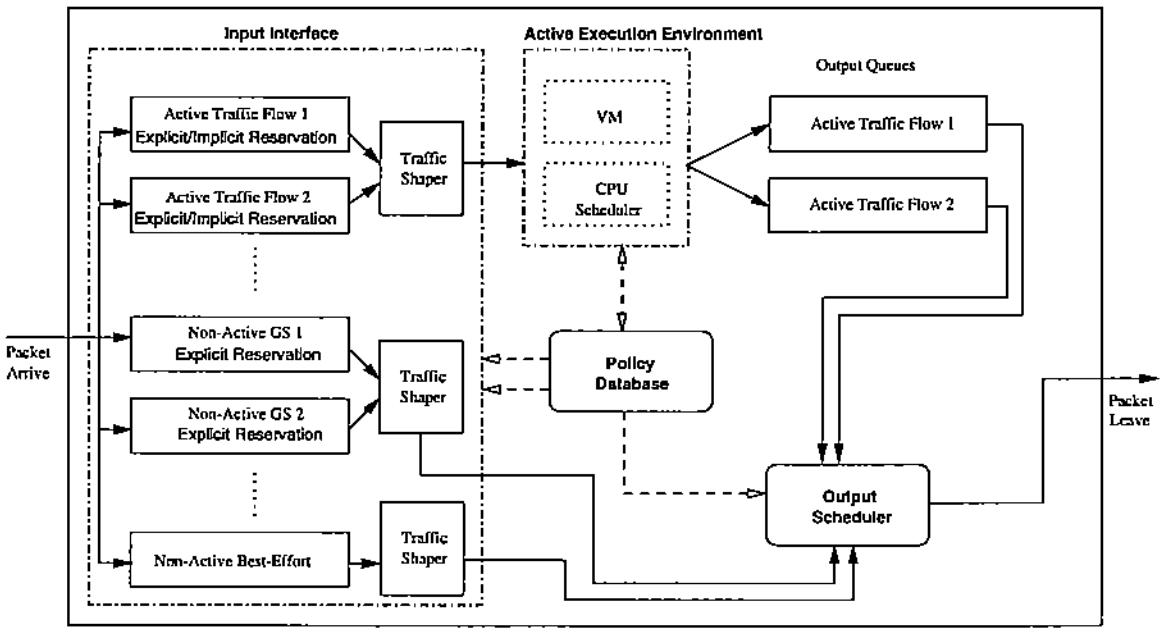

Figure 1: Architecture of an Active Router

3.2 Input Interface

3.2.1 Traffic Classifier

We classify the network traffic into four different kinds of traffic: active traffic with explicit reservation, active traffic without explicit reservation, non-active traffic with explicit reservation, and non-active traffic without explicit reservation.

The router provide a default resource limitation tuple for every active traffic flow. For active traffic with explicit reservation (which include a set-up phase before the actual data are sent), the value of the resource limitation tuple can be negotiated using primitives provided by the active router. This is very similar to the concept of `malloc()` function call in system programming. The resource request can be made at the the set-up phase, or can be made on-the-fly when necessary. This way an active traffic which needs guarantee service can reserve the required resource and become an active (or non-active) traffic with explicit reservation, while adaptable active traffic can simply request additional resource when needed. Once the resource requests are granted, the usage of the resources is up to the application.

The Diffserv traffic flows are aggregated into different super-flows based on the DS fields in their packets. These super-flows correspond to the category of non-active traffic with explicit reservation.

The router keep a flow state for each aggregated flow for the purpose of management (policing, billing, shaping, etc.). We aggregate all the best-effort traffic (traditional network traffics) into a single flow under the category "non-active traffic without

explicit reservation". The non-active guaranteed service traffic flows corresponds to the notion of non-active traffic with explicit reservation.

3.2.2 Traffic Shapers

There are three traffic shapers in our design: one for the active traffic flow, one for the non-active flow with explicit reservation and one for non-active best-effort flow. The logical separation of the traffic shapers into three pieces is for functional description only. In practice all the three traffic shapers may be implemented as one module.

3.3 Execution Environment for Active Traffic flows

3.3.1 Virtual Machine

Overview We propose a new virtual machine for our design. Although Java Virtual Machine is popular, we feel that a new customized virtual machine for running programs inside the router is necessary because of the following reasons:

- **Compactness.** The object code (or bytecode) format for currently available general-purpose virtual machines such as JVM is not very compact. The original design of these VMs is for execution in the end system or device, which usually don't need to handle a huge amount of VM programs. In contrast, the router my need to process thousands or even millions of active programs in a very short period of time. Even the existing VM specifically designed for active networks such as Spanner [13] is not able to encode any interesting algorithms in very few bytes. Our VM incorporates some specialized processing function as standard operations and include a mechanism to install user codes as standard operations.
- **Resource management.** Currently available general-purpose virtual machines such as JVM dose not include fine-grain resource management mechanisms. In particular, the VM is not tightly coupled with the OS, therefore resource management can not be done effectively. In our design, the VM includes operations dedicated to resource management tasks. Our VM plays several roles in the overall architecture. Specifically, our VM can
  - work as Bandwidth Broker (BB),
  - work as signalling mechanism,
  - change DS Codepoint definition and update policy database,
  - constraint resource consumptions of active flows.

Special Instruction Sets Here we list some VM instructions that are unique in our design:

- Resource management

- Request and Setup

  ```
  REQ UDP/123, <User B addr>, P@128 kb/s, 1pm-3pm
  ```

  This operation request a service with peak rate 128 kb/s to user B using UDP. The first-hop router act as BB and accept or reject the request. The border router of other administration domain will not honor these kind of request (yes?). The request operation can also request ACTP sessions. This will turn the active flow into Active Flow with Explicit Reservation.

- Specialized multimedia data processing
  - MPEG encoding/decoding
  - H261 encoding/decoding

- User installable operation codes

  The program can ask the virtual machine to install some particular part of the code into a single opcode. There are two modes of installable opcodes: the *secure mode* and the *light-weight mode*.

  In secure mode, the security hash algorithm MD5 is used to generate the footprint of the user code. For example,

  ```
  INS 50 ; install the following 50 bytes as a opcode.
  <program follows>
  ```

  The opcode corresponding to this fragment of code is the MD5 value of the code fragment (128 bits). A capsule can later invoke the installed opcode by

  ```
  USR1 <MD5 value>
  ```

  In light-weight mode, the opcode is determined by the user as a 16-bit number. Any number can be used in this mode. However, an unique random number is strongly suggested. Techniques to generate globally unique random numbers similar to the one described in [12] is strongly recommended. An example of light-weight mode follows.

  ```
  INS 30 1234; install the following 30 bytes as opcode 1234.
  <program follows>
  ```

  The opcode corresponding to this fragment of code is 1234 (in 16 bits). A capsule can later invoke the installed opcode by

  ```
  USR2 1234
  ```

3.3.2 CPU Scheduler

For each active traffic flow, a thread (light-weight process) of the designated execution environment (virtual machine) is created to handle the flow. The execution of the thread will be under the control of the CPU scheduler to compete for CPU time with other active traffic flows. Since all the threads are scheduled together, the CPU scheduler will ensure that each flow receive a fair share of the CPU time. After the active capsule is processed, all the active capsules generated from the processing will be put in an output queue. For each active traffic flow, there is a corresponding output queue. These output queues, together with all the output queues of the traffic shapers for non-active guaranteed service traffic flows and non-active best-effort traffic, are linked to the output scheduler.

3.3.3 Discussion

There are several issues regarding to the virtual machine that need to be investigated further. First, a mechanism is need for the installation operation INS to specify the number of bytes (the parameters) to be used as operand when USR1 or USR2 is called. Second, the INS operation may need data from several packets. A on-demand or periodic-refresh code loading scheme is needed and a mechanism to collect all the information is needed. A mechanism similar to IP fragmentation is being developed to perform this task.

3.4 Output Scheduler

The output scheduler is responsible for fairly allocating the output bandwidth among different classified traffic flows. The output scheduler will schedule the packet delivery using Weighted Fair Queuing (WFQ) or similar scheduling disciplines. Note that since every active traffic flow has its own output bandwidth limitation enforced, it is not possible for a misbehave active traffic to shut down the output link.

3.5 Active Transport Protocols (ACTP)

Our design introduce a new transport protocol called ACTP to isolate the interactions (possibly very complex) among active traffic flows and other traffic flows based on other transport protocols such as TCP or UDP. ACTP not only reduce the complexity of traffic and resource management tasks but also simplify the programming interface for active network programming by providing APIs similar to BSD-sockets.

Active Transport Protocols is a transport protocol on top of the IP protocol. ACTP provides datagram delivery similar to UDP, but with some extensions such as built-in fragmentation scheme and security mechanism to better support the active network traffic. ACTP support the concept of source/destination port similar to those in TCP or UDP to differentiate different sessions. Since the behavior of a transport

protocol can greatly influence the network performance, care must be taken when new functionality are added to it.

We have developed a fragmentation scheme on ACTP for multimedia traffic and perform some preliminary experiments [14].

ACTP support built-in security mechanisms. For a secure ACTP session, the key management protocol that will be used with IP layer security is performed at ACTP layer. The authentication, Encryption, and other security function are performed at IP layer using IPSEC [9] ("IP Authentication Header" [7] and the "IP Encapsulating Security Payload" [8]).

4 Features of ADNET

- One unique feature of our design is the explicit consideration of resource management in the design. We introduce the concept of *default resource reservation* to better service the active traffics.

- Our design explicitly include the CPU scheduler as one of the resource management component. Without proper management of CPU time, a router can not provide guarantee services to the applications.

- Our design simultaneously address the issues on the overheads associated with per-flow accounting and customized services for individual traffic flow. Our design seeks to provide a broad spectrum of services ranging from traditional stateless best-effort to Intserv-style per-flow accounting. In one extreme, per-flow customized service such as Intserv or active traffic can be provided. On the other extreme, traditional best-effort service can also be provided. In the middle, aggregated QoS schemes such as Diffserv can be provided.

- The resource reservation mechanism for traffic flows in our architecture is not active. Our current design employ the RSVP-style resource reservation mechanism. We argue that to provide safe and secure execution of active capsules a common resource reservation mechanism have to be built-in in the infrastructure. It is apparent that the current RSVP will not be suitable for this task because it is too heavy-weight and no CPU scheduling capability is included. Further research is needed to identify necessary modifications to RSVP to satisfy our need.

4.1 ADNET routers in different roles

- As *leaf (first hop) routers*:
  - The sender can simply send any traditional traffic as usual.
  - The sender can send active traffics with or without reservation. If a Premium service (as in Diffserv) is required, the sender can send an active

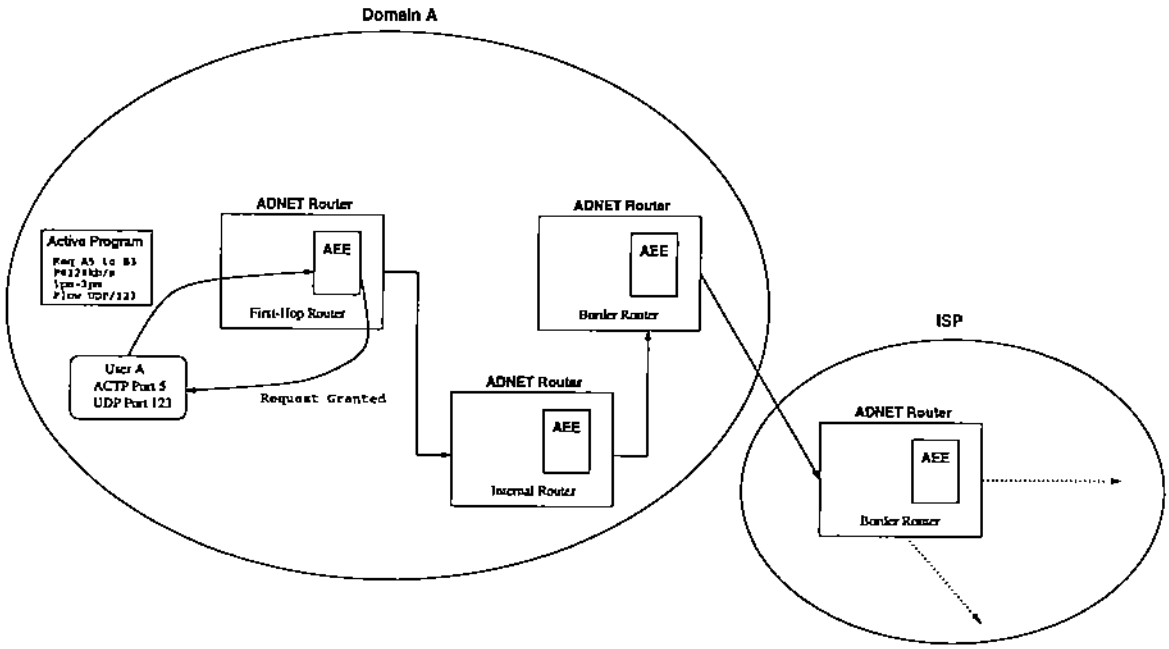

Figure 2: Active Execution Environment as Bandwidth Broker

program to the router to request resources. After validation, the request is granted by the router and the traffic from the sender will be tagged as Diffserv traffic with specific Diffserv class (figure 2).

- As *boundary (border) routers*:
  - The egress router will issue a setup request to the ingress router of ISP.
- As *core routers*:
  - The AEE only process the active flow that are authorized to go across the domain and enter the current domain.
  - The router will not accept Non-Active Explicit Reservation to ensure safety.

5 Simulation Experiments

We are simulating part of the functionality of ADNET using *ns-2* network simulator [11]. We have extended the *ns-2* to includes some functionality described in this report. Currently not all functionalities described in this chapter have been implemented yet.

In the following experiments, we compare three different scenarios of video transmissions to evaluate the architecture. They are:

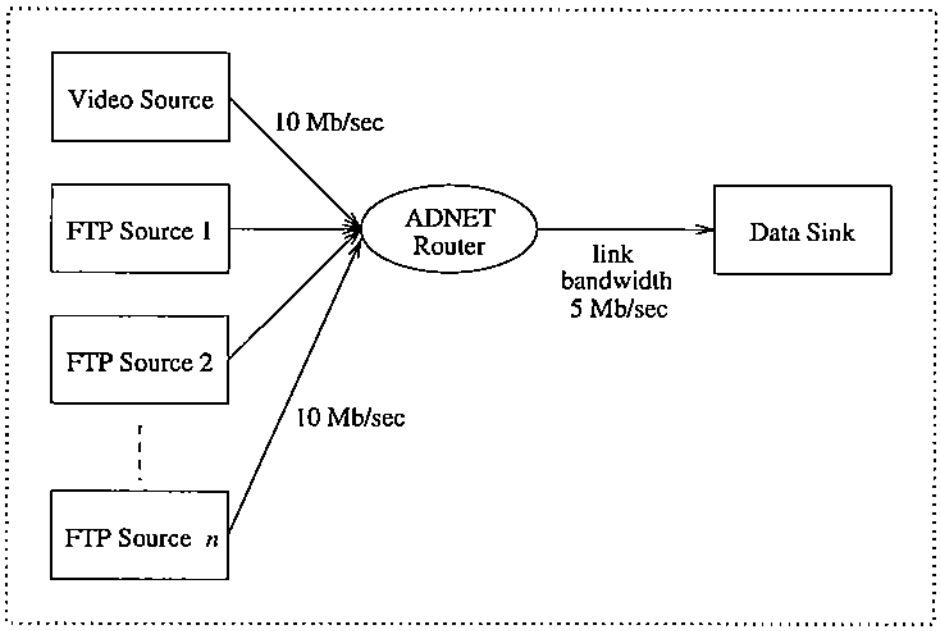

Figure 3: Simulation setup for ADNET experiments

- Case 1: sending the videos frame by frame at 30 frames/sec under IP fragmentation.
- Case 2: sending the videos frame by frame at 30 frames/sec under ACTP fragmentation.
- Case 3: sending the videos frame by frame at 30 frames/sec under ACTP fragmentation and with active program in each packet (50 bytes). The active program first query the current system queue status. If the addition of the packet will cause the queue to drop packets, the packet is forwarded to the active execution environment and transformed to 80% of its original size. The transformation will cause delay of 5 ms for the packet.

The configuration of the experiments is shown in figure 3. We use three video clips JP, LK and TC (which we digitalized from commerical video tapes) as the input data for the experiments. The details of these clips are described in appendix A.

The bottleneck link is configured at 5 Mb/sec. This value is large enough for the video source to send the video at 30 frames/sec without any loss when no competing traffics exists. The experiments are repeated for the configuration of 1 to 5 competing traffic sources (FTP sources). The value γ defined in [15] are calculated for each experiment configuration to determine the effectiveness of a particular experiment configuration. Simply speaking, γ measures the *usefulness* of the data received by the applications, which is shown to be a better QoS measure for the application than the

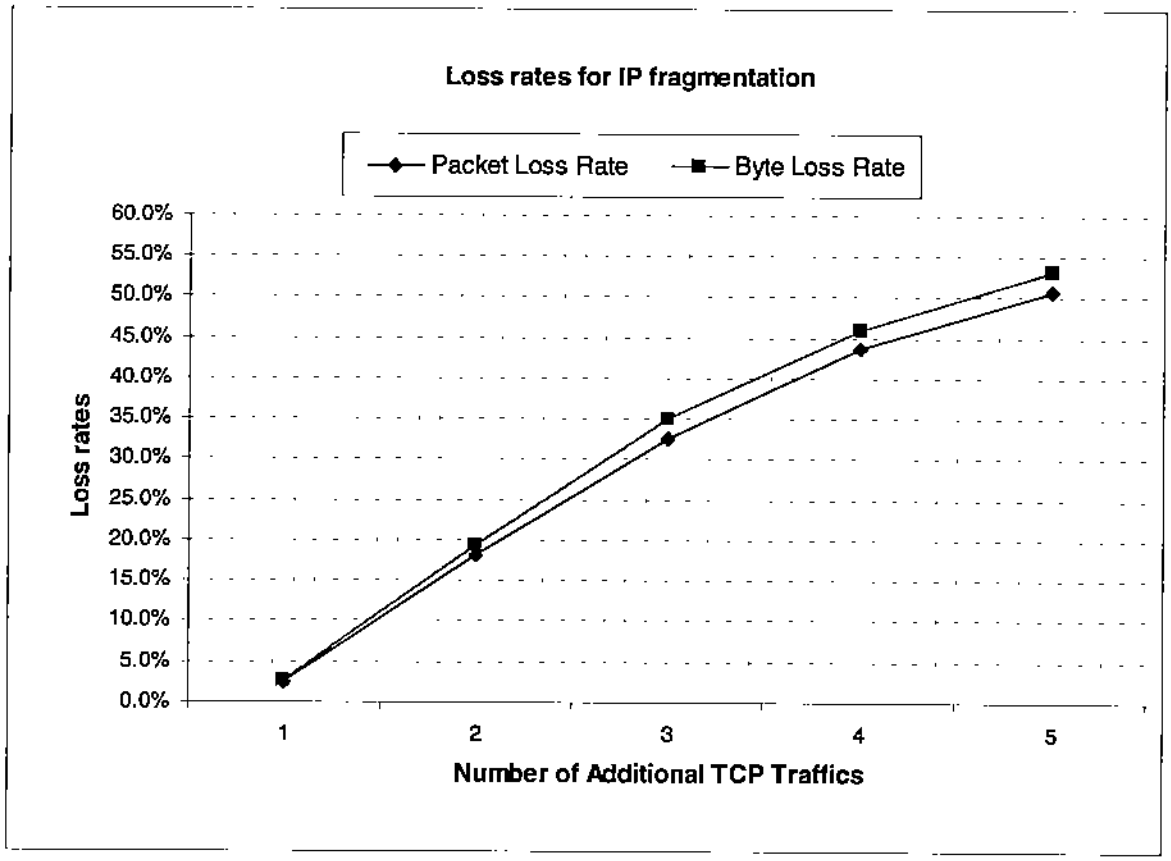

Figure 4: Packet and Byte loss rates for clip JP under IP fragmentation

commonly used metric such as packet loss rate. The results are described as follows. Figure 4 shows the packet and byte loss rate for clip JP under IP fragmentation. Figure 5 shows the packet and byte loss rate for clip JP under ACTP fragmentation. Figure 6 shows the packet and byte loss rate for clip JP under ACTP fragmentation plus transformation. Figure 7 calculates the value γ for clip JP.

Some interesting observation can be made about the results. First, the byte loss rates for the three scenarios are almost the same. Since in ADNET router the queuing discipline is DRR, it is expected to be fair to all traffic flows. However, the packet loss rates are quite different for case 3. By reducing the size of the packet, the packet has a better chance of surviving so the packet loss rates are reduced by around 10%. The most exciting results are that the QoS observed by the applications (the value γ) improves significantly for case 2, and even more for case 3 under heavy network load.

The results for clip LK (figure 8, 9, 10, and 11) and clip TC (figure 12, 13, 14, and 15) are similar.

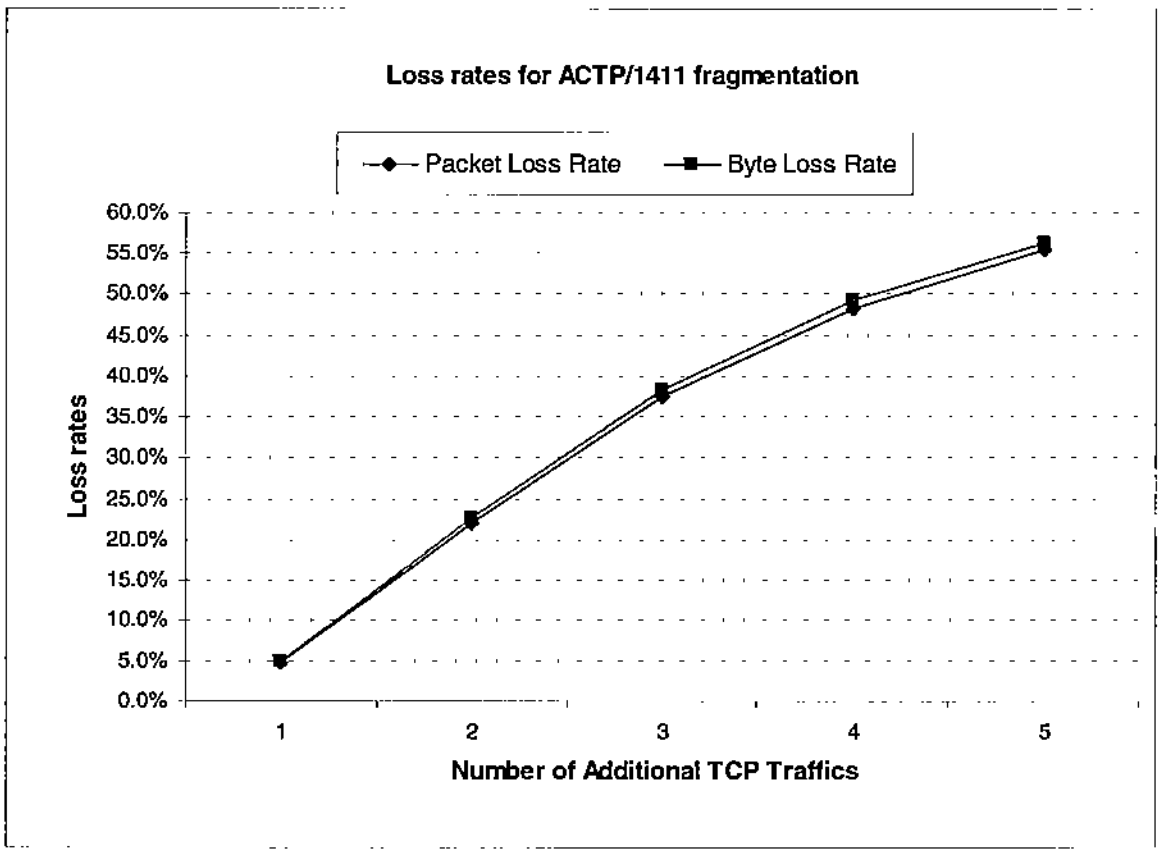

Figure 5: Packet and Byte loss rates for clip JP under ACTP fragmentation

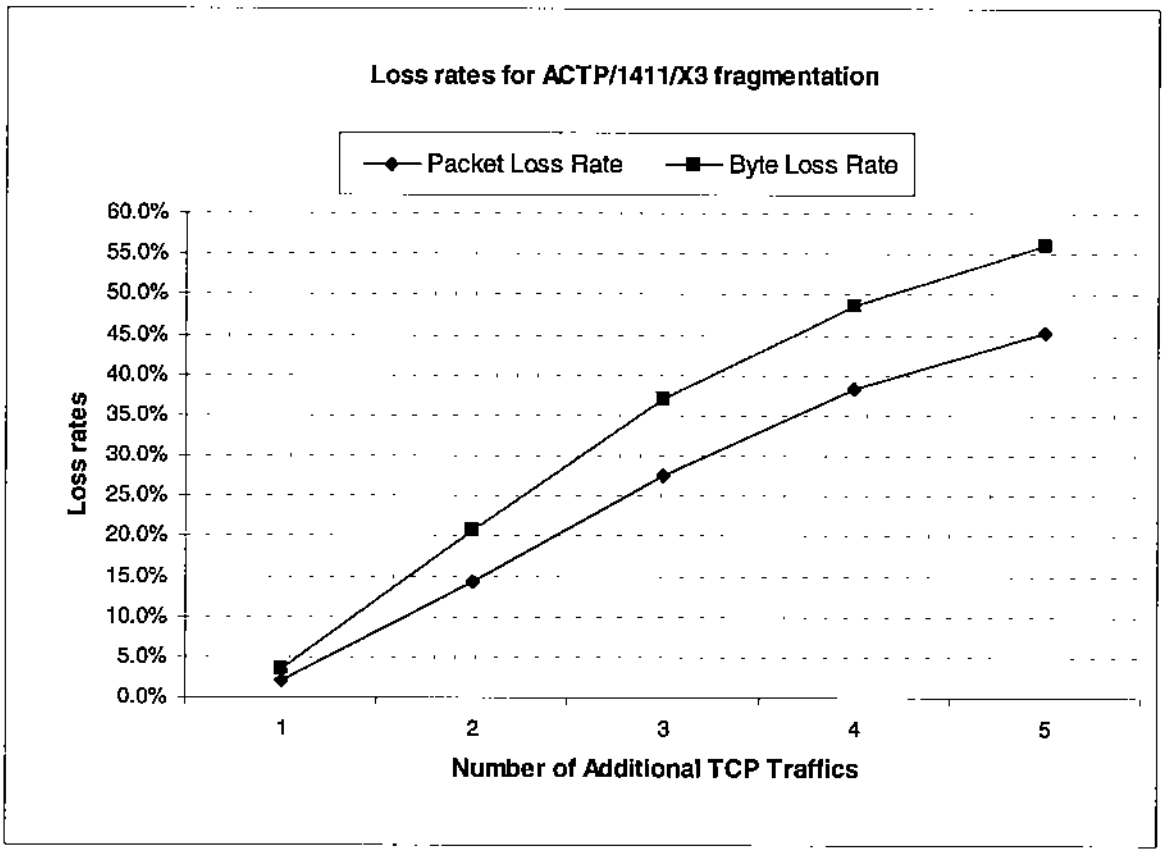

Figure 6: Packet and Byte loss rates for clip JP under ACTP fragmentation and transformation

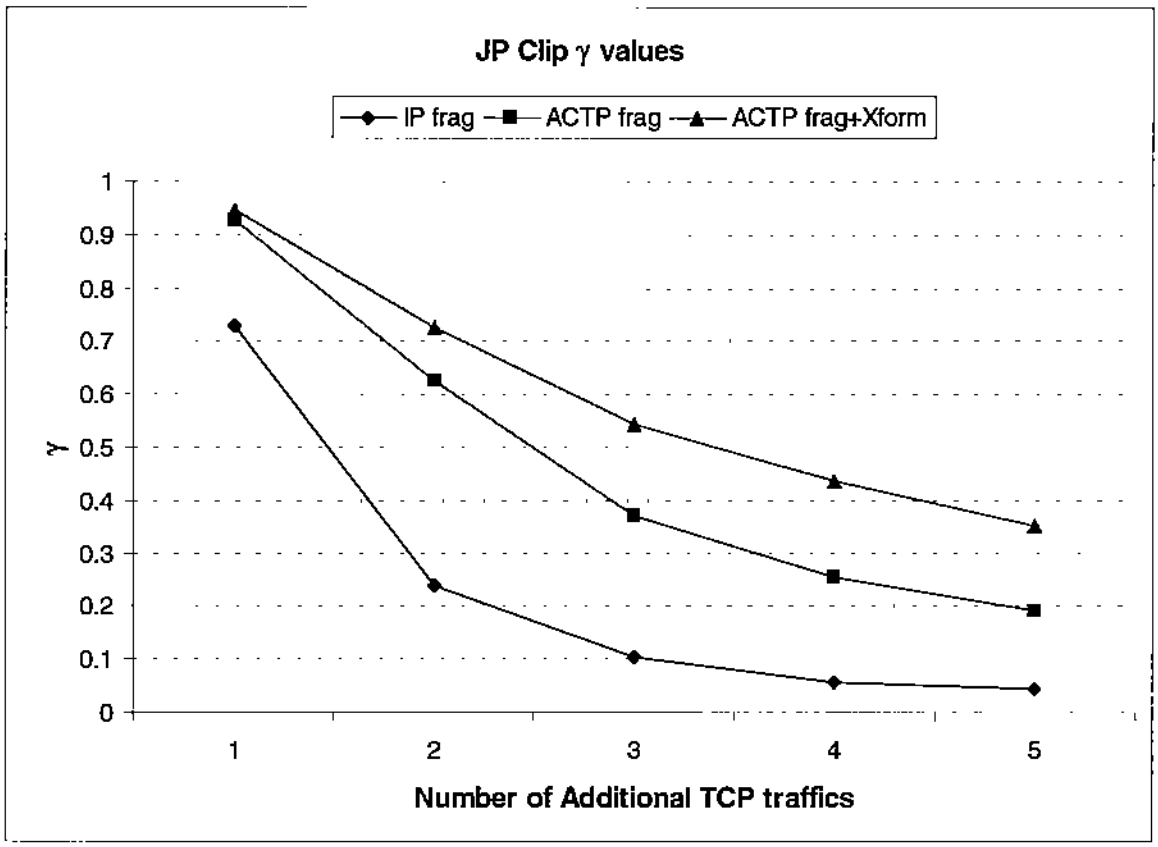

Figure 7: γ values for JP clip

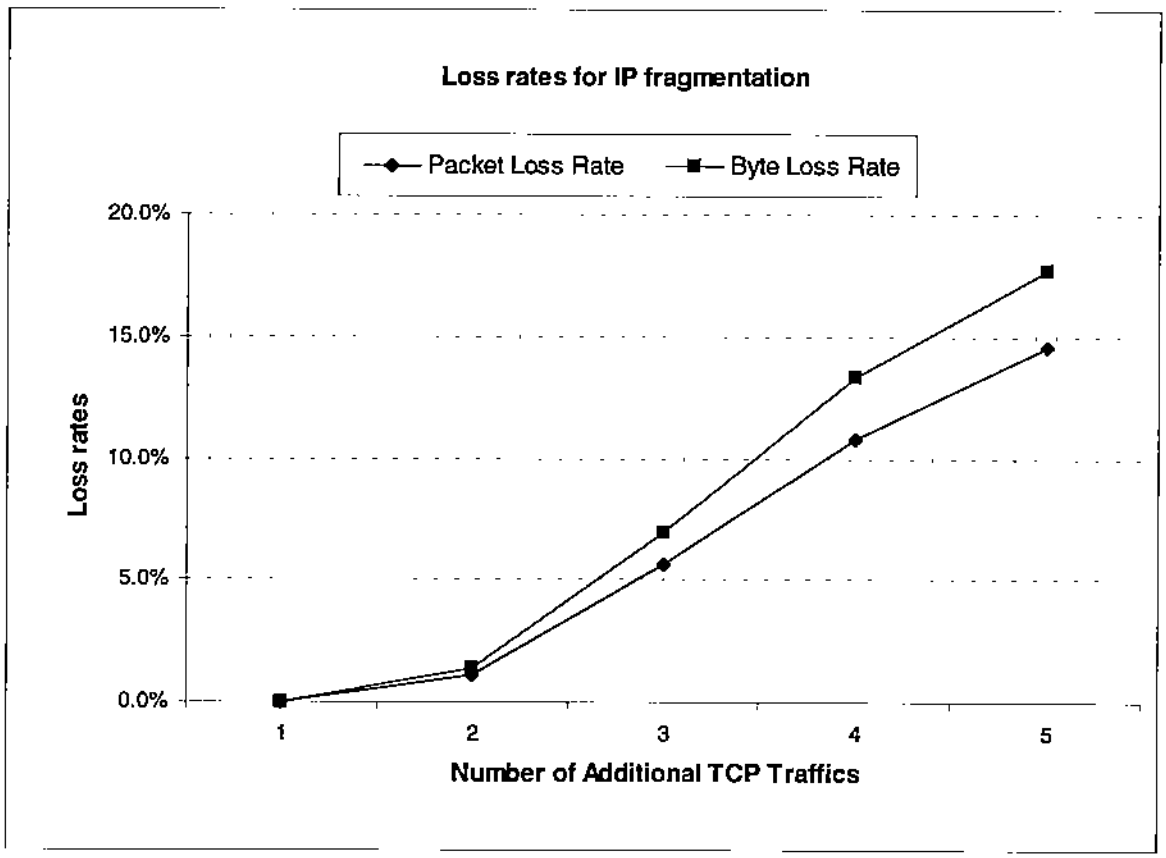

Figure 8: Packet and Byte loss rates for clip LK under IP fragmentation

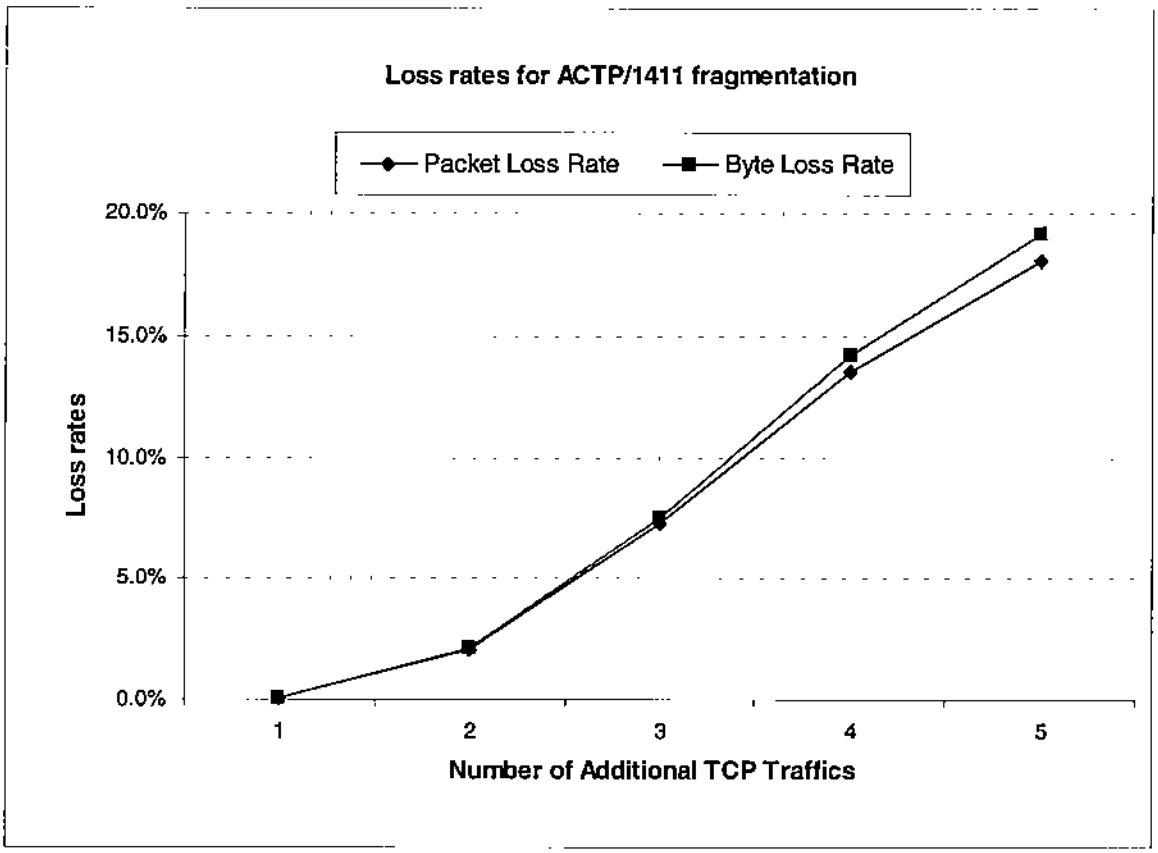

Figure 9: Packet and Byte loss rates for clip LK under ACTP fragmentation

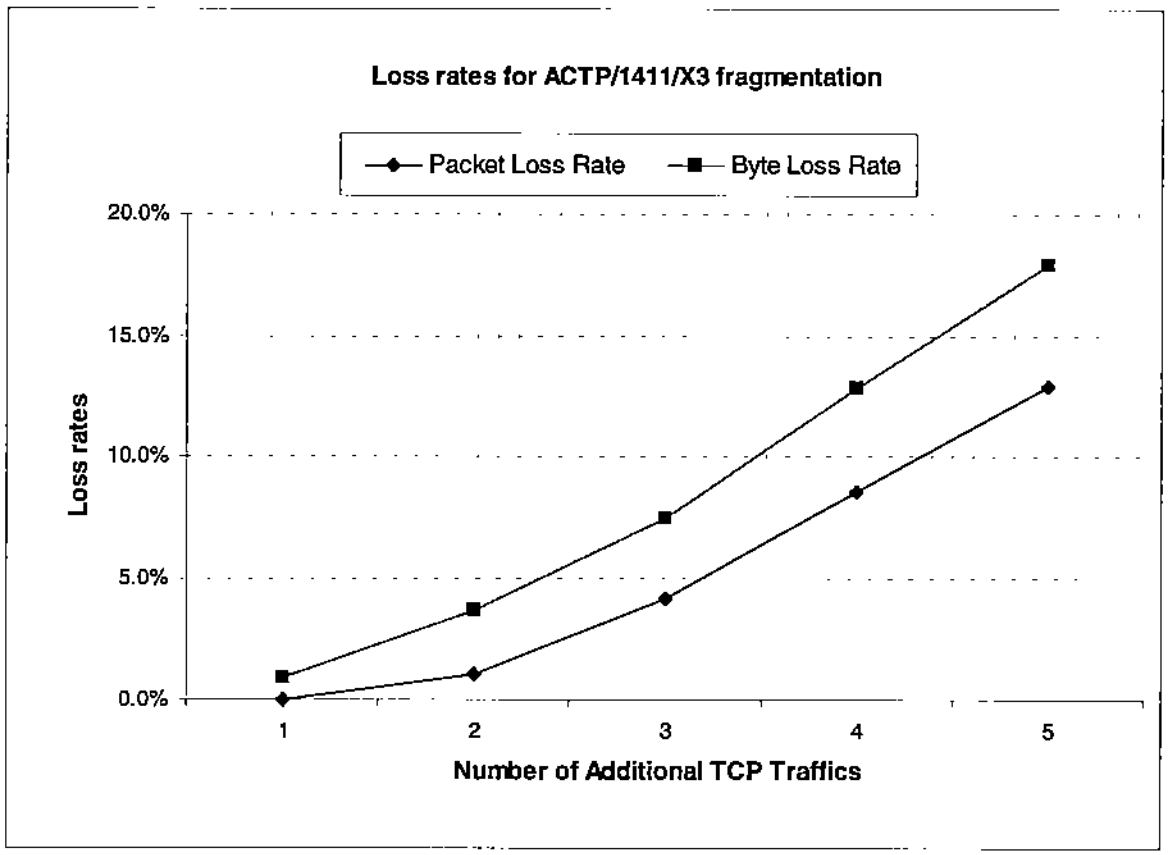

Figure 10: Packet and Byte loss rates for clip LK under ACTP fragmentation and transformation

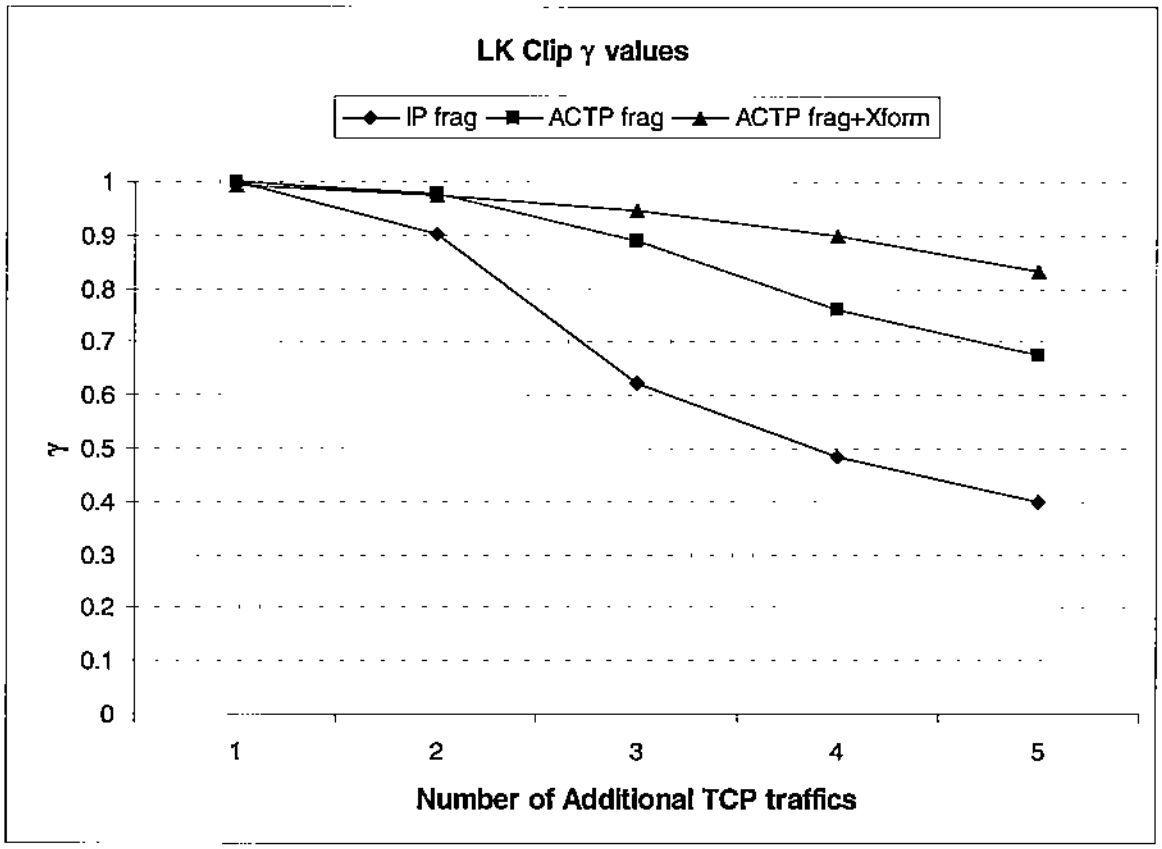

Figure 11: γ values for LK clip

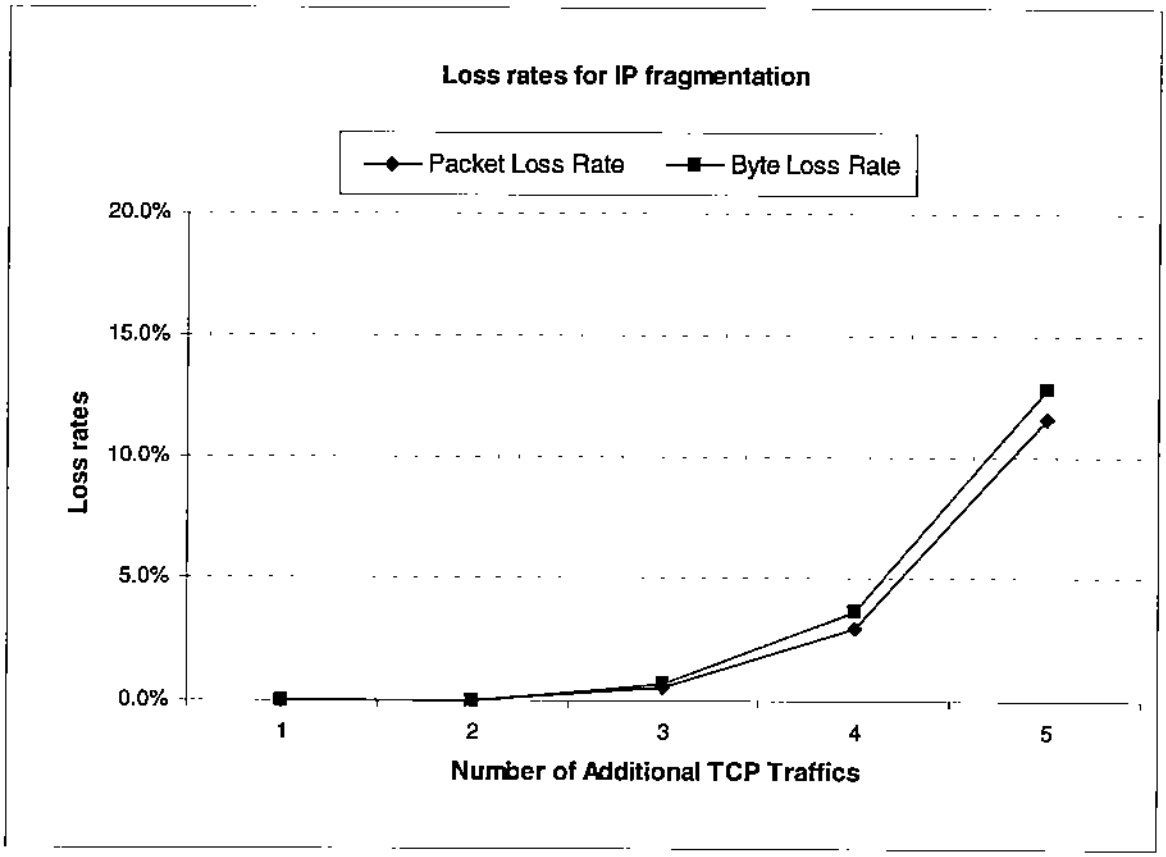

Figure 12: Packet and Byte loss rates for clip TC under IP fragmentation

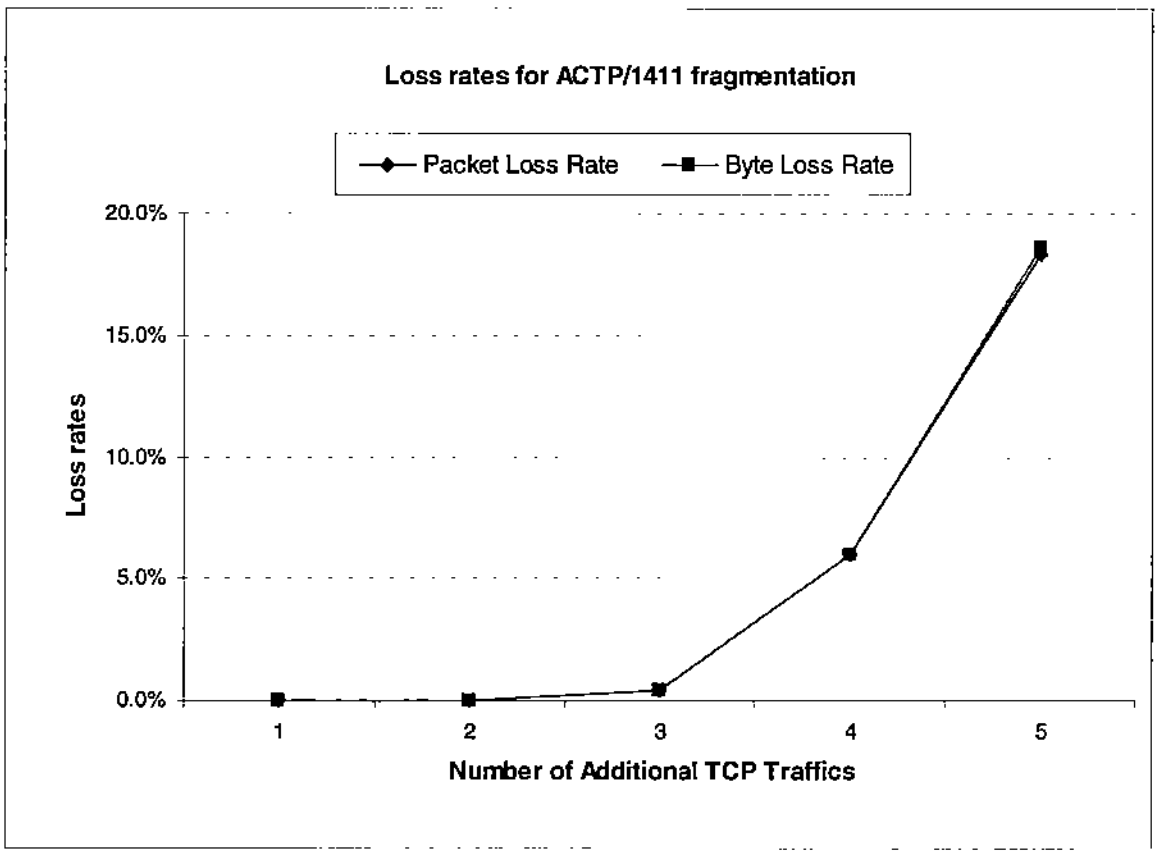

Figure 13: Packet and Byte loss rates for clip TC under ACTP fragmentation

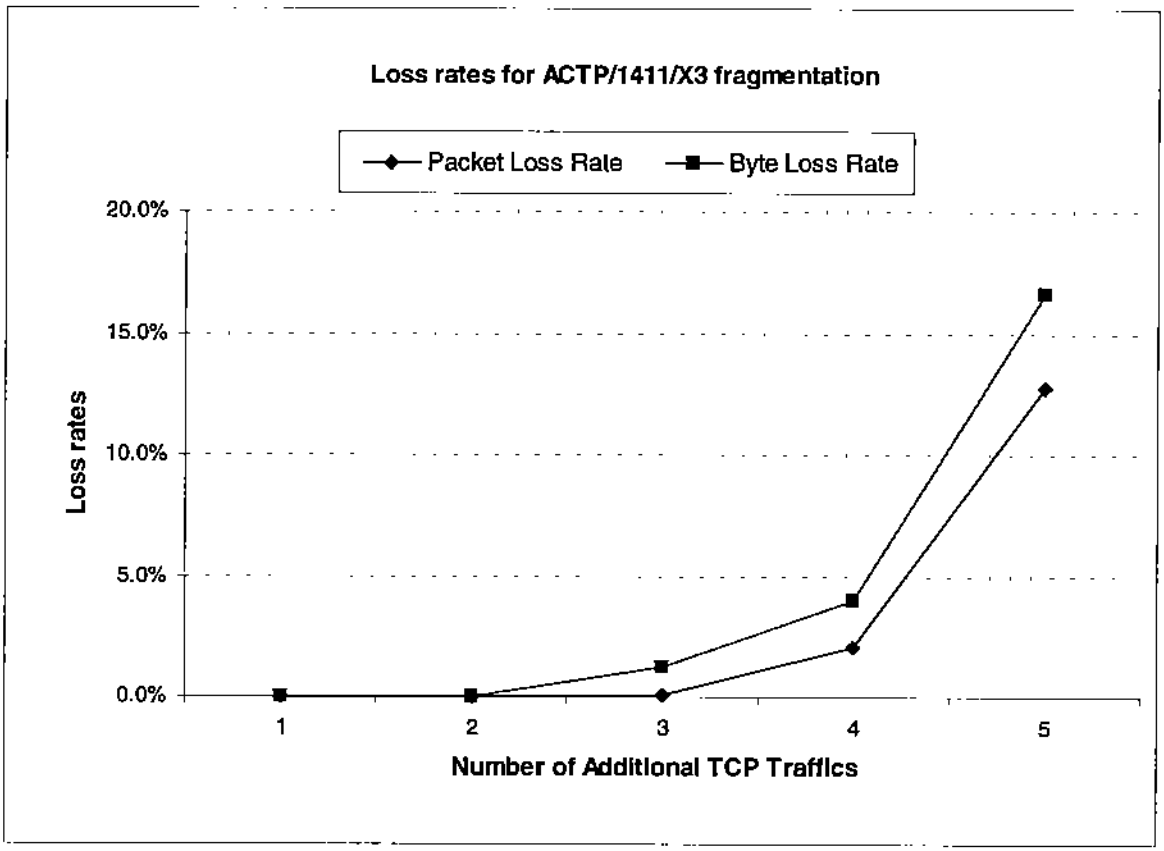

Figure 14: Packet and Byte loss rates for clip TC under ACTP fragmentation and transformation

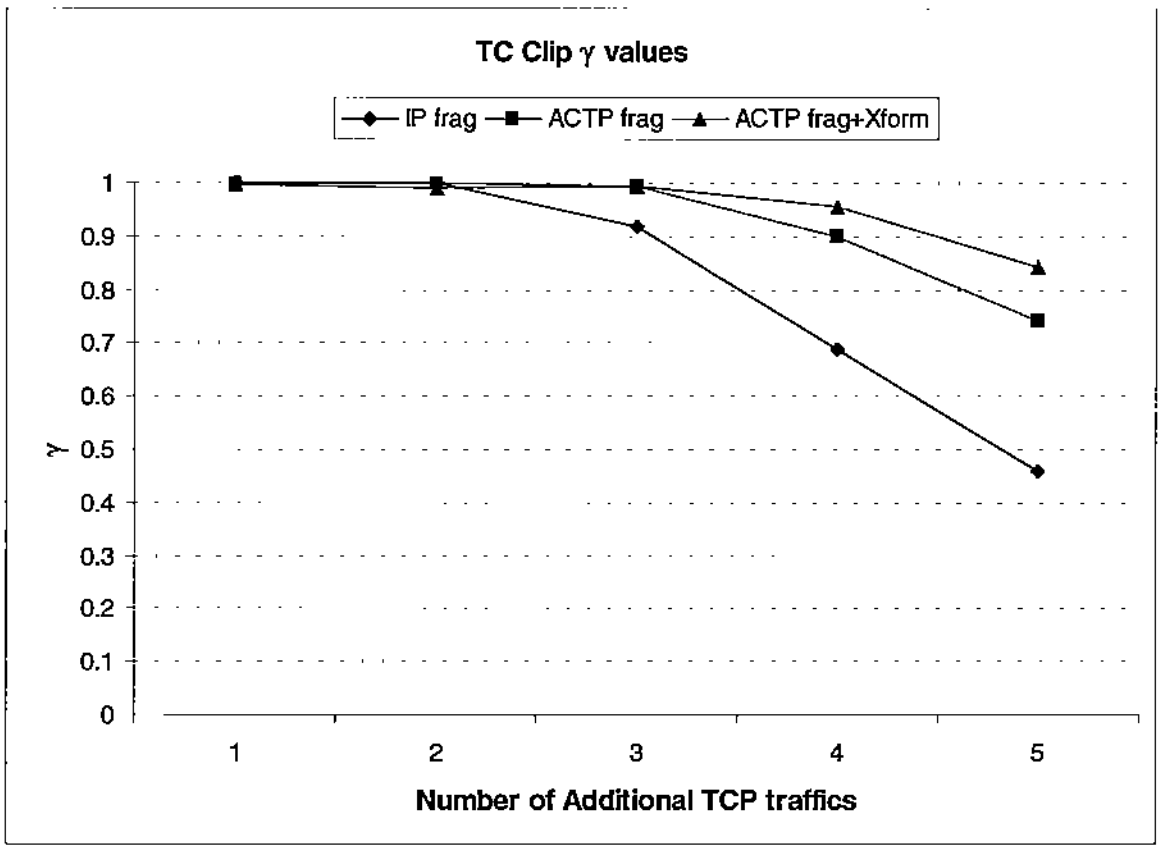

Figure 15: γ values for TC clip

6 Future Work and Conclusions

- The traffic classification in our current design is somewhat artificial because we have to consider the interoperatability with the Intserv and Diffserv architecture. The long term goal is to integrate the Diffserv architecture into the active network architecture as a special case. It may even be possible to integrate the Intserv architecture to become a unified traffic class. Some works such as Active Signalling [4] may shed some light on this aspect.

- The implementation of an efficient packet classifier is very crucial to ADNET. Currently there are some research such as [6] on this topic. It requires more research on the applicability of those approaches to ADNET and identify potential modification needed to suit our need.

References

[1] D. Scott Alexander, William A. Arbaugh, Angelos D. Keromytis, and Jonathan M. Smith. A Secure Active Network Environment Architecture: Realization in SwitchWare. *IEEE Network Magazine*, 12(3), May/June 1998.

[2] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. An Architecture for Differentiated Services. *RFC 2475*, December 1998.

[3] Bob Braden, David Clark, and Scott Shenker. Integrated Services in the Internet Architecture: an Overview. *RFC 1633*, June 1994.

[4] Bob Braden and Ted Faber. Active Networking Projects: ARP and ACC. URL `http://www.isi.edu/active-signal/`.

[5] Kenneth L. Calvert, Samrat Bhattacharjee, Ellen Zegura, and James Sterbenz. Directions in Active Networks. *IEEE Communication Magazine*, October 1998.

[6] Dan Decasper, Zubin Dittia, Guru Parulkar, and Bernhard Plattner. Router Plugins: A Software Architecture for Next Generation Routers. In *Proceedings of the ACM SIGCOMM*, Vancouver, British Columbia, Canada, September 1998.

[7] Stephen Kent and Randall Atkinson. IP Authentication Header. *RFC 2402*, November 1998.

[8] Stephen Kent and Randall Atkinson. IP Encapsulating Security Payload (ESP). *RFC 2406*, November 1998.

[9] Stephen Kent and Randall Atkinson. Security Architecture for the Internet Protocol. *RFC 2401*, November 1998.

[10] Shunge Li. *Quality of Service Control for Distributed Multimedia Systems.* PhD thesis, Department of Computer Science, Purdue University, December 1997.

[11] VINT Project. UCB/LBNL/VINT Network Simulator - ns (version 2). URL `http://www-mash.cs.berkeley.edu/ns/`.

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications. *RFC 1889*, January 1996.

[13] B. Schwartz, A. Jackson, T. Strayer, W. Zhou, R. Rockwell, and C. Partridge. Smart Packets for Active Networks. In *Proceedings of the Second IEEE Conference on Open Architectures and Network Programming*, New York, N.Y. USA, March 1999.

[14] Sheng-Yih Wang and Bharat Bhargava. A fragmentation scheme for multimedia traffics in active networks. In *Proceedings of the 17th IEEE Symposium on Reliable Distributed Systems (SRDS'98)*, October 1998.

[15] Sheng-Yih Wang and Bharat Bhargava. A Model for Active Techniques for Compressed Video Transmission. Technical Report CSD-99-046, Purdue University, Department of Computer Sciences, September 1999.

A Description of Video Clips Used in the Experiments

One major problem in many multimedia research is that the testing data are not representative for real-world situation. For example, many research papers test their video-related algorithms using small video clips created in the research laboratory. Since the characteristics of these non-professional video clips are very different from commercial production videos, their test results can not be easily generalized to apply to real-world problems. In order to test our approaches and algorithms under realistic scenario, we created three video clips from commercial video tapes. They are:

- *Jurassic Park* (JP) - A typical movie with different type of scenes. The characteristics of this video clip is similar to most of the other videos in the market.
- *Tai Chi* (TC) - An instruction video which teach Tai Chi Chun (one form of Chinese Martial Art). This video is a typical instruction video. The feature of a typical instruction video is that usually there are no fast changing scenes and background. The changes in the video is slow because most of the scenes involve only motions of the instructor.
- *Lion King* (LK) - A cartoon video. The feature of a cartoon video is that there are many sharp edges which usually don't occur in real world video. The compression of cartoon is usually more difficult because MPEG scheme works better on real-world objects.

Table 1 gives the details of these video clips.

Clip Name	Jurassic Park	Tai Chi	Lion King
Resolution	320 x 240	352 x 240	320 x 240
# of frames	3000	2996	3750
Total Size	23091162	12497528	10273216
Avg. I frame size	19458.98 (251)	7684.57 (334)	8056.66 (313)
Avg. P frame size	14423.05 (750)	5632.38 (666)	4752.44 (938)
Avg. B frame size	3695.67 (1999)	3094.44 (1996)	1317.00 (2499)
Overall avg. frame size	7696.38	4170.34	2738.85
GOP pattern	IBBPBBPBBPBB	IBBPBBPBB	IBBPBBPBBPBB

Table 1: Profiles of Three Video Clips