AN ADAPTABLE NETWORK ARCHITECTURE FOR MULTIMEDIA TRAFFIC MANAGEMENT AND CONTROL

Sheng-Yih Wang and Bharat Bhargava

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

ABSTRACT

We have designed an adaptable network architecture, called ADNET, which provide mechanisms to allow the application adapt to the resource constraints to achieve improved QoS. In our experiments we compare three schemes (IP fragmentation, ACTP fragmentation with or without active program) of video transmissions. We find QoS is improved in ACTP scheme with active programs. Our design aims to unify different QoS control mechanisms together to provide a wide range of network services to all users and meet their specific needs.

1. INTRODUCTION

Quality of Service (QoS) provision for emerging applications is an important research topic. QoS can be provided at different layers. The application layer and the network layer are the two major layers where QoS parameters and QoS control mechanisms are studied. 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.

Intserv is a reservation-based QoS control mechanism. The objective of Intserv 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. Scalability to large number of flows and complexity in implementations are the two most serious drawbacks of Intserv approach.

Diffserv is a new effort to provide QoS support in IP networks. Diffserv tries to eliminate the need of RSVP-style resource reservation by aggregating traffic flows with similar QoS requirements into one single traffic class. The packets 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.

One drawback of the Diffserv approaches is that the QoS parameters can not be dynamically changed. Since the QoS in Diffserv is specified by Service Level Agreement (SLA) 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, sometimes the application QoS parameters are 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 of the network are.

Active network [4] is a 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 that 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. We show 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 paper, 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 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 allows 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 traffic 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.

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

Secure: ADNET employs the standardized Secure IP Protocol (IPSEC) [5] to ensure the secure communications. The security requirement is an adaptable feature in our design. It can be specified as a QoS parameter when setting up reservations.

PORTIONS OF THIS WORK WERE SUPPORTED BY A GRANT FROM NSF UNDER CCR-9901712, A GRANT FROM IBM, AND BY SPONSORS OF THE CENTER FOR EDUCATION AND RESEARCH IN INFORMATION ASSURANCE AND SECURITY (CERIAS).

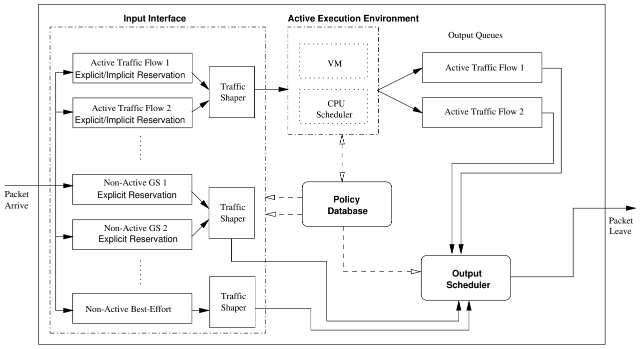

Figure 1: Architecture of an ADNET Router

Efficient: The efficiency in ADNET comes in two flavors: short-cut path for non-active traffic and primitive operations for multi-media payloads.
Scalable: Because our design address both the customized aspect and the aggregated aspect of network services, it is scalable to very large networks.
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

ADNET is inspired and partially derived from the work in Integrated Services [3], Differentiated Services [2], and Active Networks [4]. Figure 1 shows the architecture of ADNET. The design focuses on the explicit management of resources for classified traffic flows. The notion of classified traffic flow is a generalization of the usual sense of the traffic flow which encompasses a variety 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 performs classification and shaping for the traffic. AEE is composed of a virtual machine and a CPU scheduler which provides an execution environment for active network traffic. Output scheduler performs output bandwidth allocation to share fairly the output bandwidth among different types of classified traffic flows. Policy database provides policy information to other major components and interacts with AEE to update the policies.

We choose to keep a per-classified-flow state to enforce various policies on the traffic. We argue that per-classified-flow state is a necessity 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 the current design, a resource limitation tuple consists 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.

3.1. Input Interface

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

The router provides a default resource limitation tuple for every active traffic flow. For active traffic with explicit reservations (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 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 guaranteed service can reserve the required resources and become an active (or non-active) traffic with explicit reservations, 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 reservations.

The router keeps 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 traffic) into a single flow under the category "non-active traffic without explicit reservations". The non-active guaranteed service traffic flows corresponds to the notion of non-active traffic with explicit reservations.

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

3.2. Execution Environment for Active Traffic flows

Virtual Machine. We propose a new virtual machine for our design. Although Java Virtual Machine is popular, we argue that a new customized virtual machine for running programs inside the router is necessary because of the following reasons: (1) *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 may 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 [6] is not able to encode any interesting algorithms in few bytes. (2) *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 (1) work as a Bandwidth Broker (BB), (2) work as a signalling mechanism, (3) change DS Codepoint definitions and update policy database, (4) constrain resource consumptions of active flows.

In our design, VM includes some instructions that are unique: (1) Resource management (Request and Setup). (2) Specialized multimedia data processing (MPEG and H261 encoding/decoding). (3) 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. In light-weight mode, the opcode is determined by the user as a 16-bit number.

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. 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 fair queuing scheduling disciplines. Note that since every active traffic flow has its own output bandwidth limitation enforced, it is not possible for a misbehaving active traffic to shut down the output link.

3.4. Active Transport Protocols (ACTP)

Our design introduces 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 reduces the complexity of traffic and resource management tasks but also simplifies the programming interface for active network programming by providing APIs similar to BSD-sockets.

Active Transport Protocol 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 supports 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.

We have developed a fragmentation scheme on ACTP for multimedia traffic and performed a series of experiments [7].

ACTP supports built-in security mechanisms. For a secure ACTP session, the key management protocol for IP layer security is performed at ACTP layer. The authentication, encryption, and other security functions are performed at IP layer using IPSEC [5]

4. FEATURES OF ADNET

A unique feature of our design is the explicit consideration of resource management. We introduce the concept of *default resource*

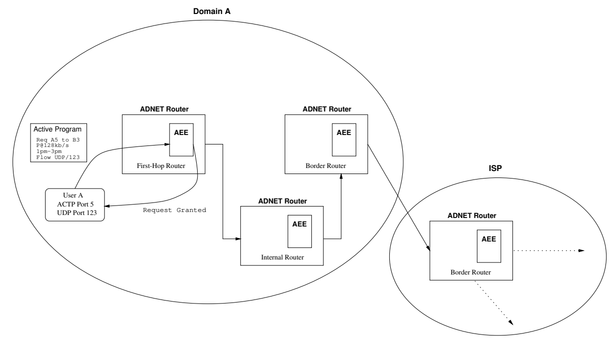

Figure 2: Active Execution Environment as Bandwidth Broker

reservation to provide better service the active traffic.

ADNET explicitly includes the CPU scheduler as one of the resource management component. Without proper management of CPU time, a router can not provide guaranteed services to the applications.

ADNET addresses simultaneously the issues of the overheads associated with per-flow accounting and customized services for individual traffic flow. ADNET 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 be provided. In the middle, aggregated QoS schemes such as Diffserv can be provided.

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

ADNET routers can play different roles, depending on where they are located: (1) As *leaf (first hop) routers*: The sender can simply send any traditional traffic as usual. The sender can send active traffic with or without reservation. If a Premium service (as in Diffserv) is required, the sender can send an active 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). (2) As *boundary (border) routers*: The egress router will issue a setup request to the ingress router of ISP. (3) 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 a part of the functionality of ADNET using *ns-2* network simulator. We have extended *ns-2* to includes certain functionalities described in this paper.

In the following experiments, we compare three different scenarios of video transmissions. They are: (1) sending the video one frame at a time at the rate of 30 frames/sec under IP fragmentation.

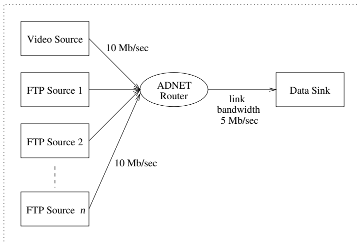

Figure 3: Simulation setup for ADNET experiments

(2) sending the video at the same rate under ACTP fragmentation. (3) sending the video at the same rate under ACTP fragmentation and with active program in each packet (50 bytes). The active program first queries the status of the current system queue. 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 a delay of 5 ms for the packet. The configuration of the experiments is shown in figure 3. We use three MPEG video clips Jurassic Park (JP), Tai Chi (TC), and Lion King (LK), which we digitalized from commercial video tapes, as the input data for the experiments. The details of these clips are described in [9].

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 traffic source exists. The experiments are repeated for the configuration of 1 to 5 competing traffic sources (FTP sources). The value γ defined in [8] are calculated for each configuration to determine the effectiveness of a particular 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 commonly used metric such as packet loss rate. The results are described as follows. Figure 4 shows the result for clip JP (from left to right, top to bottom: packet/byte loss rate under IP fragmentation, packet/byte loss rate under ACTP fragmentation, packet/byte loss rate under ACTP fragmentation plus transformation, γ values).

Some observations 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 and clip TC are similar (omitted due to space constraints, see [9] for details).

6. 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.

REFERENCES

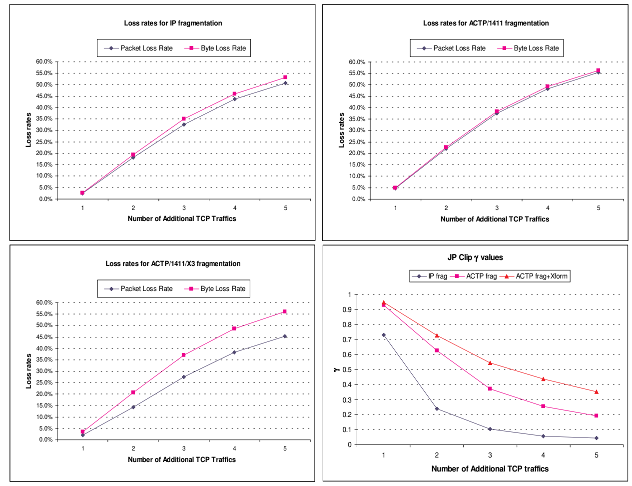

Figure 4: Result for clip JP

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

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

[6] 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 City, New York, March 1999.

[7] 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.

[8] 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.

[9] Sheng-Yih Wang and Bharat Bhargava. An Adaptable Network Architecture for Multimedia Traffic Management and Control. Technical Report CSD-99-048, Purdue University, Department of Computer Sciences, November 1999.