Flow Labelled IP: Connectionless ATM Under IP*

Peter Newman, Tom Lyon, and Greg Minshall†
Ipsilon Networks Inc.‡

Abstract

IP traffic on the Internet and private enterprise networks has been growing exponentially for some time. This growth is beginning to stress the traditional, processor based design of current day routers. Switching technology offers much higher aggregate bandwidth but presently only offers a bridging solution. Various proposals are under way to support IP routing over ATM switching technology; however, these proposals hide the real network topology from the IP layer by treating the data-link layer as a large, opaque, network cloud. We argue that this leads to complexity, inefficiency and duplication of functionality in the resulting network.

We propose an alternative in which we discard the connection oriented nature of ATM and integrate fast ATM hardware directly with IP, preserving the connectionless nature of IP. We use "soft" state in the ATM hardware to cache the IP forwarding decision. This enables further traffic on the same IP flow to be switched by the ATM hardware rather than forwarded by IP software. We claim that this approach combines the simplicity, scalability, and robustness of IP with the speed, capacity, and multiservice traffic capabilities of ATM.

1. Introduction

In the early days, before the advent of Asynchronous Transfer Mode (ATM), life was easy and communication protocols were, on the whole, mostly straight forward. Two philosophical approaches to switching evolved: *capitalist switching* and *socialist switching*. According to the capitalist approach, you specify exactly how much bandwidth you require, and if it is available the network gives it to you. The bandwidth is yours, use it or waste it as you will — you don't have to share it with anyone. Socialist switching takes a completely different viewpoint. In socialist switching we have a huge pool of bandwidth — and we share it! You want some bandwidth, you take it, and when you are done, kindly return it to the pool. The capitalists were fond of connections so they knew where to send the bill. Socialists, however, distrusted connections because they break, and bandwidth was free anyway. No-one tried particularly hard to make phone calls over the Internet, but equally there were not too many web browsers on ISDN.

ATM evolved as an attempt to combine both fundamental approaches to switching (packet switching and circuit switching) into a single integrated switching mechanism — the grand unified network. It has recently received much attention because of its high capacity, its bandwidth scalability, and its ability to support multiservice traffic. However, in the control plane, ATM takes a somewhat capitalist approach in its reliance upon the connection oriented paradigm. The vast majority of modern data networking protocols are connectionless and take a thoroughly socialist view of communication. This mismatch has led to complexity, inefficiency, and duplication of functionality in attempting to apply ATM technology to data communication.

The Internet Protocol (IP) has also seen very rapid growth in the last several years though its underlying design philosophy is much more socialist in nature [Clark88]. Current research suggests that given a suitable implementation, IP is no less capable of supporting real-time and multimedia traffic than ATM [Clark92, rfc1633]. Much attention is being focused on the use of IP multicast for multimedia and conferencing applications [Thyag95]. Furthermore, many believe that the connectionless model on which IP is based, with the addition of soft-state for traffic management, is a much more robust and flexible

* Networld+Interop, Las Vegas, April 1996.
† {pn,pugs,minshall}@ipsilon.com
‡ <http://www.ipsilon.com>

basis on which to construct an integrated services network.

In this paper we investigate the implementation of IP directly on top of ATM hardware while preserving the connectionless model of IP. We discard the connection oriented nature of the ATM protocol stack and couple the fast ATM switching hardware directly to IP. This has the particular advantage of not requiring a signalling protocol, or any address resolution protocol, and requiring only the standard IP routing protocols — protocols that have been well debugged and heavily tested. It also directly supports IP multicast which is currently incompatible with the ATM implementation of multicast. Of course, it rather assumes that IP, instead of ATM, be the underlying universal, ubiquitous, integrated services protocol, but some would claim that this is already the case.

2. IP: Necessary and Sufficient …

The Internet is currently connected to approximately 5 million hosts on 45 thousand interconnected networks covering 86 countries and continues to grow exponentially. The last year and a half has seen rapid growth in the number of homes with personal computers connected to the Internet for entertainment and information in addition to business use. The web browser is perhaps the first real widely distributed multimedia networking application and the world wide web continues to enjoy dramatic growth. This explosive growth is fuelled by user demand which seems likely to continue.

However, the traditional design of packet switch (router) on which the Internet is based is beginning to run out of steam. Routers are expensive and of limited throughput when compared to switches. To support the continued traffic demand of the Internet, IP needs to go faster and cost less. To support the increasing demand for real-time and multimedia applications IP also needs to support quality of service (QOS) selection. We believe that both can be met by the application of ATM switching technology to the design of an IP router. Various approaches are under way in the industry and the standards bodies to implement IP over ATM. We discuss the difficulties that beset these efforts and propose an alternative. Our objective is simply to make IP go fast and offer QOS support by integrating state-of-the-art switching technology with IP routing and forwarding. We aim to combine the flexibility of IP with the speed of switching.

Switching offers the ability to support high bandwidth links at wire speed and switches of very high aggregate throughput may be implemented. A switch achieves this high capacity by implementing the data path completely in hardware. Thus there is a tradeoff between the flexibility of the router, which can make an independent forwarding decision for every packet, and the speed of the switch, which requires state to be established in its forwarding tables.

ATM offers scalability of both link bandwidth and switch capacity. It is also well suited to the application of VLSI implementation and the widespread, almost frenzied, interest in ATM promises the rapid decrease in cost that comes from volume production. We are interested in using ATM for the switching technology because the hardware is now standardized and available, it is fast, and the price tag is falling. However, ATM is a connection-oriented switching technology.

The debate regarding the pros and cons of the datagram approach versus the virtual circuit approach is an old one and is now as much a matter of religious principle as technical virtue. IP is connectionless and since our goal is to make IP go fast at low cost we seek the most efficient integration of the connectionless IP protocol and ATM switching hardware. Yet the considerable success of the Internet is in large part due to the connectionless nature of IP.

IP is built on a very low level building block — datagram forwarding. No assumptions are made regarding the services provided by the underlying network beyond the ability to forward a datagram in the direction of the destination. This has permitted IP to operate over a very wide range of underlying network technologies. Multiple types of communications service are offered by enhancement of the basic forwarding service. Datagram forwarding requires no state to be maintained for individual connections. This has proven extremely robust in the presence of failures.

In a connection oriented network, possibly as much as 90% of the signalling code is there to handle error conditions. This code is impossible to thoroughly test and is almost never correct. A soft state approach in which state in the network is periodically refreshed covers a large spectrum of possible error conditions with a very simple recovery mechanism. Also, the delay involved in connection establishment and the limited number of connection setups per second available in current switches motivate us to explore the use of ATM switching hardware in a connectionless manner.

3. Obscured by Clouds

Since ATM is itself connection oriented, the heart of the problem is to make use of the speed and capacity of the switching hardware without sacrificing the scalability and flexibility that come from the connectionless nature of IP. A number of approaches to the implementation of IP over ATM have been proposed in the literature and in the standards bodies [Alles95]. These include: LAN emulation [LANE] from the ATM Forum; classical IP over ATM [rfc1577, Cole95] from the IPATM working group of the IETF; [NARP] and [NHRP] from the Routing Over Large Clouds (ROLC) working group of the IETF; and Multiprotocol Over ATM (MPOA) currently under discussion at the ATM Forum [MPOA]. All of these approaches obscure the real topology of the underlying network from the internetwork layer routing protocol. To IP the physical network becomes a large opaque cloud which results in some significant problems.

First there is a duplication of functionality. Both IP and ATM each require routing protocols. Not only does this imply duplication of the routing protocols but it also leads to duplication of the maintenance and management functions. In addition, management functions are required to handle the interaction between the two. This makes it much more difficult to locate problems. When connectivity is lost it is much more difficult to determine where the fault lies if two separate routing protocols are involved. It is also possible for undetected routing loops to be formed in certain situations [Cole95].

For efficient multicast capability IP requires knowledge of the underlying network topology. Without this information, a multicast packet arriving at a router for transmission to leaf nodes attached to the cloud must be replicated at the router and each copy must be transmitted separately across the cloud. This is clearly inefficient as multiple copies of the same packet will be transmitted unnecessarily across data links within the cloud. The replication function would be far better implemented at the appropriate forks in the multicast tree within the cloud. For low bandwidth datagram traffic this is simply inefficient but it could prove very disruptive for higher bandwidth real-time traffic with quality of service requirements, such as IP voice and video multimedia applications.

All routers and route servers connected to a large cloud may be considered to be logically one hop away from each other. Conventional routing protocols have N^2 scaling difficulties in the case where each router has many neighbors, arising from the routing table size, the amount of routing update processing, and the amount of routing update traffic generated. But if the routers are not logically fully meshed the reliability of the network is reduced because it is possible that two hosts that are physically connected have lost logical connectivity. Introducing logical connectivity on top of a physical network can only reduce the reliability since more systems need to be functioning to achieve connectivity.

In a physical network, such as Ethernet, it is a simple matter for a router to discover its neighbors and then fire up a routing protocol to connect to the network. It is also simple for a host to discover what services are available on the network. In a cloud environment, however, the cloud can be of any arbitrary size and topology, so a router cannot discover its neighbors, it must be assigned them. It also makes it much more difficult for a host to automatically discover network services. This lack of support for auto-configuration leads to greatly increased management and manual configuration requirements. Also, in a cloud model, routers are required to interconnect multiple logical subnets. This requires the configuration of multiple logical interfaces on a single physical interface — the “one-armed router” — which also increases the configuration requirements.

4. Connectionless Connections

The concept of a flow has emerged within the IP community over the past few years. A flow is a sequence of packets sent from a particular source to a particular (unicast or multicast) destination that are related in terms of their routing and any local handling policy they may require [Deering95]. It performs a similar function in a connectionless network to the role the connection plays in a connection oriented network. In IP version 6 the inclusion of a flow label in the packet header allows the forwarding process to be enhanced by caching routing decisions. Also network resources may be reserved on behalf of a flow to offer quality of service guarantees.

IP is connectionless but many applications above IP employ a connection oriented transport protocol. The most efficient mapping of IP onto ATM must consider the characteristics of the application, or at least the transport protocol, in deciding whether to establish an end-to-end ATM connection on behalf of any specific flow [Rekhter95, Cole95]. Flows carrying real-time traffic, flows with quality of service requirements, or flows likely to have a long holding time, will be handled most efficiently by mapping them into an individual ATM connection. Short duration flows, and database queries would best be handled by connectionless hop-by-hop packet forwarding between IP routers using shared, pre-established ATM connections between the routers. This is particularly true for exchanges such as DNS lookups that consist of a single packet in each direction. Establishing an end-to-end ATM connection for every IP packet flow would impose a heavy load on the ATM signalling protocol and impose unnecessary delay on query-response traffic.

One of the reasons that IP scales well to large networks is due to its connectionless nature. If a router or a link fails in a moderately well connected network, IP simply routes around the failure. If we establish end-to-end connections across an ATM cloud the failure of a link or router will invalidate all associated connections. This will exert a heavy load on the signalling protocol to re-establish all of the ATM connection state. Also, many

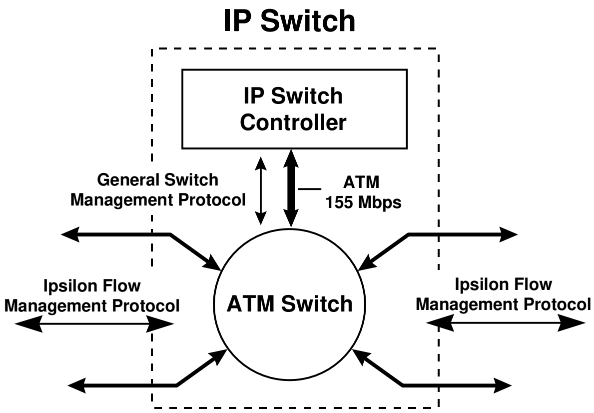

Figure 1: Structure of an IP Switch

connections that are not directly associated with the failed component will become sub-optimal, perhaps highly sub-optimal when the topology changes. It is also possible that routing loops can form after a topology change until the old routing information is purged from the address resolution servers and route servers [rfc1620].

It is clear that in order to take advantage of the efficiency of switching at the data link layer, and to offer quality of service guarantees, state information must be maintained within the switches. However, the simplicity and robustness of IP is much more likely to be preserved if the state is maintained locally rather than on an end-to-end basis and if the state is "soft" rather than "hard". Soft state is state information that is installed within a network for reasons of performance enhancement but is not crucial to the correct operation of the network [Clark88, rfc1633]. It is typically designed to be refreshed periodically such that many possible error conditions may be corrected by simply timing out old state. This leads us to consider the possibility of a connectionless implementation of ATM.

5. Flow Labelled IP

Our goal is to achieve the most efficient implementation of IP on top of fast switching hardware. We now consider using standard ATM hardware but completely changing the control software in order to operate each switch in a connectionless manner. The result will be a router with attached switching hardware that has the ability to cache routing decisions in the switching hardware. We call this an *IP switch* since it allows packet flows to be switched, bypassing the router, once the routing information has been cached in the switch.

5.1 A Switch By Any Other Name …

To construct an IP switch, fig. 1, we take the hardware of an ATM switch as it stands, without any modification, but completely remove the software resident in the control processor above AAL-5. Thus we remove the signalling, any existing routing protocol, and any LAN emulation server or address resolution servers, etc. In place of the ATM software we load a simple, low-level control protocol, called the Generic Switch Management Protocol [GSMP], to give the IP switch controller access to the switch hardware. The IP switch controller is a high-end processor running a standard IP router software package with extensions that allow it to make use of the switching hardware. These extensions include a simple flow management protocol (IFMP) to associate IP flows with ATM virtual channels, a flow classifier to decide whether to switch each flow, and GSMP to control the switch hardware.

At system startup a default forwarding ATM virtual channel is established between the IP routing software running on the IP switch controller and that of each of its neighbors. The default forwarding channel is used for the hop-by-hop connectionless forwarding of IP datagrams. Thus we now have the ability to forward IP packets but to gain the benefit of the switching hardware we need a mechanism to associate an IP flow with a specific ATM label (virtual path and virtual channel identifier VPI/VCI).

5.2 Flow Classification

We characterize an IP flow according to the fields in the IP/TCP/UDP header that determine the routing decision such as: type of service, protocol, source address, destination address, source port, destination port, etc. Two packets belong to the same flow if the values of these fields are identical. Several different flow types may be defined, each characterized by a different set of header fields (though the set of flow types must be ordered so that a most specific match operation may be performed.)

When a packet is received across a default forwarding channel it is reassembled and submitted to the control processor for forwarding. The processor forwards the packet in the normal manner but it also performs a flow classification on the packet to determine whether future packets belonging to the same flow should be switched directly in the ATM hardware or continue to be forwarded hop-by-hop by the router software.

Flow classification is a local policy decision. The flow classifier inspects the contents of the fields that characterize the flow and makes its decision based upon a local policy expressed in a table. For example, by looking for well known source or destination port numbers one can identify the application. Flows belonging to FTP data connections may be configured to be switched, but DNS queries could be forwarded as datagrams. (The performance of such a classification is investigated in the following section.)

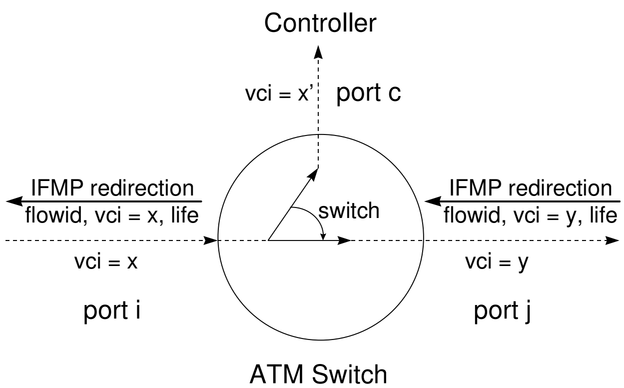

Figure 2: Establishing a switched flow

5.3 Flow Management Protocol (IFMP)

If the processor decides that the flow should be switched it selects a free label (label x) from the label space of the input port (port i) on which the packet was received, fig. 2. We make the assumption that virtual channels are unidirectional so the ATM label space (VPI/VCI range) of the incoming direction of each link is owned by the input port to which it is connected. The processor also selects a free label (label x′) on its control port (port c). (The control port is the port, either real or virtual, by which the control processor is connected to the switch.) The switch driver is then instructed to map label x on input port i to label x′ on the control port c.

After making the entry in the translation table of the switch input port the processor sends an IFMP [IFMP] Redirection message upstream to the previous hop from which the packet came. The redirection message contains the label x, a flow identifier, and a lifetime. The flow identifier contains the set of header fields that characterize the flow. The redirection message requests the upstream host or router to transmit all further packets with header fields that match those specified in the flow identifier on the ATM virtual channel specified by the label. The lifetime field specifies the length of time for which this redirection is valid. Unless the flow state is refreshed, before the lifetime expires, this binding of flow and label should be deleted and further packets belonging to the flow will be transmitted on the default forwarding channel.

From this point packets belonging to the flow will arrive at the switch controller, port c, with the ATM VPI/VCI label x′. The packets will still be reassembled and forwarded by the IP forwarding software but the process is speeded up because the previous routing decision for this flow was cached in the router software and can be indexed by the label x′.

The real benefit of switching comes when the downstream router or host also runs the same redirection algorithm. When the router receives a redirection message from its downstream neighbor on port j, redirecting the flow to label y, it can switch all further traffic belonging to that flow directly within the ATM hardware. The router does this by instructing the switch to map label x on port i to label y on port j. Thus the traffic is no longer sent to the control processor but is switched directly to the required output port.

When an IP switch accepts a redirection message it also changes the encapsulation it uses for the redirected flow. The encapsulation used for IP packets on the default forwarding channel is the standard LLC/SNAP encapsulation over AAL-5. The encapsulation used for each IP packet on a flow redirected to a specific virtual channel removes all of the header fields that characterize the flow from the header of each packet [ENCAPS]. The IP packet with the resulting compressed header is then encapsulated in AAL-5 and transmitted on the specified virtual channel. The fields that are removed are stored by the router that issued the redirection and are associated with the specified ATM virtual channel. The complete packet, including TTL and checksum fields, may be reconstructed using the incoming label to access the stored header fields. This approach is taken for security reasons. It allows an IP switch to act as a security firewall without having to inspect the contents of each packet. It prevents a user from establishing a switched flow to a permitted destination or service behind a firewall and then changing the IP packet header to gain access to a prohibited destination.

Conceptually, each IP switch maintains a background refresh timer. When the background refresh timer expires, the state of every flow is examined. If a flow has received traffic since the last refresh period its state is refreshed. Flow state is refreshed by sending a redirect message upstream with the same label and flow identifier as the original and a new lifetime. If a flow has received no traffic since the last refresh period its cached state is removed. This will involve issuing an IFMP Reclaim message upstream to reclaim the label for reuse. The flow state is not deleted until an IFMP Reclaim Ack message is received to acknowledge release of the requested label. (Reclaim messages may also be used to release labels in use if the free label space is close to exhaustion.) For flows that are labelled, but not switched, the control processor can examine its own state to see whether the flow has received any traffic in the previous refresh period. For flows that are switched the control processor must query the switch hardware to discover whether a specific channel has recently been active.

The flow management protocol is advisory in nature. The decision to accept a redirection request is local and redirection messages may be ignored. Redirection messages are not acknowledged since the first packet arriving on the new virtual channel will indicate acceptance of the request. The protocol is also symmetric in that no distinction is made between a user interface (UNI) and a network interface (NNI). This leads to a very simple implementation.

5.4 Point-to-Point

LAN emulation, and classical IP over ATM, etc. seek to establish a logical shared medium network model on top of ATM. However, we propose a point-to-point network model — a much more natural model for ATM. All routing protocols deal well with point-to-point links. Point-to-point links existed in IP before the advent of Ethernet multi-access broadcast links and there may be as many point-to-point links (SLIP and PPP) in the Internet today as there are shared medium links.

We have adopted a point-to-point network model rather than a cloud model. The Internet is proof that IP can scale to very large networks without requiring the concept of a data-link cloud. Also we have been careful to separate the act of labelling a flow from that of switching a flow. Choosing to switch a labelled flow is a purely local decision. From outside an IP switch, one cannot determine whether a particular flow has been switched or forwarded other than it's increased performance. This separation of labelling and switching, and the local nature of the switching decision, ensures scalability to large networks. The labelling or switching decision for any particular link has no effect on the rest of the network.

5.5 Multicast

An IP switch can support IP multicast without any modification to the Internet Group Management Protocol (IGMP). Flow redirection proceeds in exactly the same manner as for unicast traffic. At an IP switch, where an incoming multicast flow is replicated into a number of branches, each branch may be individually redirected by its downstream neighbor. If the incoming multicast flow is labelled, the multicast capability of the ATM switch may be used directly on those outgoing branches that have redirected the flow onto a specific virtual channel. The switch can also send a copy of the multicast flow to the control processor so that branches that have not decided to redirect the flow may receive their copies of the traffic over the default forwarding channel.

IP multicast offers a multipoint-to-multipoint service. Any sender can transmit traffic to the multicast group. Individual flows, however, are point-to-multipoint since each flow is specific to a single source. ATM hardware only offers a point-to-multipoint multicast service. The result of the flow redirection process for a multicast group will be to establish a point-to-multipoint virtual channel from every sender that has recently transmitted traffic to the group.

5.6 Quality of Service

Each IP switch can make its own quality of service decisions according to local policy. Each flow is classified as part of the forwarding operation and quality of service information may be included in the flow classification decision based upon the application, the type of service field in the IP header, the protocol, etc. Each IP switch may also base its quality of service decision on the capabilities of the underlying ATM switch hardware. For current generation switches, separating the traffic into real-time and best-effort flows may be all that can be supported. Future switch designs are likely to be able to offer sophisticated scheduling capabilities [Clark92, Floyd95, Wake94].

Individual quality of service requests for each flow may also be supported using the resource reservation protocol (RSVP) [Zhang93, RSVP]. RSVP performs a similar function to the ATM signalling protocol in that it can reserve network resources for a particular flow. The major differences are that RSVP is receiver initiated and that it uses a "soft" state approach rather than the "hard" state of ATM signalling. RSVP allows an application to specify the traffic characteristics of a flow using a *flowspec*, similar in nature to the traffic descriptor of ATM traffic management. A reservation request may be accepted or denied by each IP switch in the path using an admission control policy. Resources are reserved by configuring the queueing and scheduling hardware within the ATM switch, and the flow may be policed by configuring the UPC hardware in the ATM switch according to the flowspec. An IP switch should be configured to redirect all flows requesting bandwidth reservation in order to switch them. This will allow the traffic management capabilities of the ATM hardware to be employed to guarantee the requested quality of service.

5.7 Latency

Transmission of the first packet on a new flow across a network of IP switches has the effect of leaving an ATM connection in its wake if all of the IP switches are configured to switch that type of packet. For a connection oriented transport protocol such as TCP the first packet on a new flow will be the SYN packet in the forward direction and the SYN ACK packet in the return direction. This exchange is used in the three-way handshake that establishes the transport connection. In the typical case, for a network of IP switches, by the time the SYN ACK has returned to the source a specific ATM virtual channel will have been established from source to destination in the forward direction. Thus as soon as the first data packet

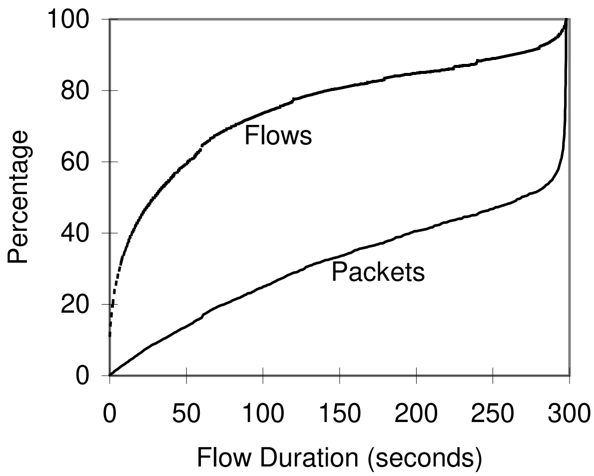

Figure 3: Cumulative Distribution of Flows and Packets vs. Flow Duration

is sent on the new transport connection it will be carried on an end-to-end ATM virtual connection. If the virtual connection is still in process of being established on any link, data is forwarded by the routing software on the default forwarding channel.

If a link or router fails at any time the normal process of connectionless dynamic routing will establish a new route. When routes or routing policy changes any existing related state will be invalidated and flushed. Affected traffic will once again be forwarded over the default forwarding channel and new virtual connections will be rebuilt from the point of failure across the new path. Traffic will continue to flow over the new path until the flows go idle for a sufficient time for each virtual connection to time out.

6. Simulation Results

The performance gain that results from the integration of an ATM switch with an IP router is somewhat dependent upon the characteristics of the incident traffic. If all of the traffic consists of single packet queries and single packet responses between a very large population of sources and destinations, the ATM switch will add very little. However, even in this situation, our approach will offer better performance than connection based IP over ATM since it can offer connectionless datagram forwarding. Many applications, such as file transfer and real time audio or video, transmit a significant quantity of information after establishing a connection. Also some applications such as remote login, while not transferring large quantities of information, have long holding times and tend to transmit a large number of small packets. These will benefit from the establishment of a connection within the switch both by removing the per packet processing overhead and by reducing the transfer delay by allocating a higher quality of service to such applications.

To investigate the benefit of our approach we obtained a traffic trace from the Internet backbone[§]. The trace contains five minutes of traffic taken at 5:15pm on Sep 25, 1995. It was taken by monitoring an FDDI ring that connects traffic from the San Francisco Bay Area to and from the Internet backbone. The trace includes a timestamp, IP source and destination addresses, the packet length, and source and destination port numbers for each packet. Backbone traffic is most likely to present a worst case for flow switching because of the large number of independent conversations that are multiplexed together. If we can demonstrate a performance enhancement with Internet backbone traffic there will be a far greater enhancement possible for campus and enterprise backbone traffic.

We first performed a simple flow analysis on the trace. For this purpose two packets belong to the same flow if they have the same IP source and destination addresses. A flow is deleted after it has remained idle for 60 seconds although the duration of the flow is recorded as being the time from when it was created until the time of the last packet transmitted on the flow. The traffic flow analysis presented in [Claffy95] suggests that a timeout value of the order of 60 seconds is a reasonable compromise between the size of the flow table and the probability of deleting flows that will shortly become active again. Fig. 3 presents the cumulative distribution of flows and packets against the flow duration. From fig. 3 we see that a high percentage of flows are of relatively short duration but carry only a small percentage of the packets. Also, only a small percentage of flows are of long duration but these flows account for a large percentage of the packets. For example, 64 percent of the flows have a duration less than 60 seconds but these flows only account for 16 percent of the packets. These are exactly the statistics that we would like to see. We can forward packets on the short duration flows but switch the long duration flows.

Next we investigated the flow characteristics of each of the protocols present in the trace. The protocol is defined as either the IP protocol number of the packet, or if that indicates TCP or UDP, the well-known port number from either the source port or the destination port numbers. For each packet in the trace we check to see if a suitable flow is available. If so, the statistics of the flow are updated else a new flow is created. For this purpose a

[§] We are grateful to K. Claffy and Hans-Werner Braun, Applied Network Research, San Diego Supercomputer Center, for making the trace available to us. The trace is available at `<ftp://www.nlanr.net/Traces>`.

Protocol	port		%flows	%pkts	%bytes	flows/s	pkts/s	duration	pkts/flow	bytes/pkt
IP in IP		✓	0.04	2.73	2.57	0.09	456	173.1	2307	253
TCP ftp-data	20	✓	0.76	12.09	15.18	2.17	2018	118.2	525	338
TCP ftp-cntrl	21		1.55	0.74	0.23	6.50	124	38.6	16	83
TCP telnet	23	✓	1.39	4.81	1.61	4.24	803	114.3	114	90
TCP smtp	25		10.26	4.80	2.82	49.49	802	18.2	15	158
UDP dns	53		45.30	5.57	3.04	216.56	929	15.4	4	147
TCP gopher	70	✓	0.45	0.54	0.55	1.87	91	43.3	40	275
TCP http	80	✓	17.94	40.21	41.53	72.98	6717	56.5	74	278
TCP pop-v3	110		0.08	0.05	0.03	0.41	9	27.0	21	148
TCP authent	113		2.12	0.19	0.05	10.54	32	9.0	3	64
TCP nntp	119	✓	0.35	6.56	6.59	0.68	1096	176.7	627	270
UDP ntp	123		5.01	0.20	0.06	25.02	33	1.37	1.3	83
TCP netbios	139	✓	0.03	0.08	0.15	0.11	14	69.8	82	501
UDP snmp	161		1.35	0.26	0.11	6.14	43	17.9	6	115
TCP login	513	✓	0.09	0.24	0.14	0.31	41	88.1	92	156
TCP cmd	514	✓	0.01	0.13	0.07	0.06	21	49.1	316	149
TCP audio	1397	✓	0.00	2.20	2.62	0.01	367	167.9	15653	321
TCP AOL	5190	✓	0.18	0.46	0.38	0.51	77	129.8	84	223
TCP X-11		✓	0.08	0.66	0.53	0.18	111	160.6	276	217

Table 1: Flow Statistics per Protocol

flow is defined as suitable if it was established between the same source and destination IP addresses and for the same protocol as the packet being processed.

The results are presented in table 1 for all protocols with a recognizable protocol or port number that contributed more than 0.05% of the total packets in the trace. (The table accounts for about 82% of the total number of packets.) For each protocol the table gives the percentage of the total number of flows, packets, and bytes contributed by that protocol. It gives the mean number of flows/s after the initial startup phase, the mean number of packets/s, the average duration of each flow, the average number of packets transmitted across each flow, and the mean number of bytes per packet. Protocols with characteristics for which it appears worth establishing a flow are marked "✓". These are protocols with an average flow duration in excess of about 20 seconds and which transmit an average of more than about 40 packets per flow (an arbitrary limit). If we assume that these flow characteristics are a property of the application behind the protocol, and the manner in which people use the applications, then for any individual protocol the results should remain relatively independent of the position in the network that the measurement is made. The characteristics of each protocol should also change only slowly over time. Thus we can use this information in deciding whether to establish a flow for any given packet.

It is interesting to note that http (web traffic) shows an average of 74 packets per flow, much higher than the value typically quoted (about 15–20 packets per flow). This is because we are looking at host pair flows, which allows multiple TCP connections between the same two IP addresses to share a single flow, rather than assuming a separate flow for each TCP connection.

In another experiment each packet is first classified according to its protocol. If it belongs to a protocol marked "✓" in table 1 it is suitable for switching. If neither the source nor destination port numbers are well known (less than 1024 or a recognized registered number) we assume the packet is suitable for switching if belongs to TCP but not if it is UDP. (This is the best guess we can make for packets that do not have a recognizable port number.) For those packets classified for switching we check to see if a suitable flow exists. If the search fails a flow is created. In this experiment a flow is suitable if it has the same source and destination IP address as the packet being processed. Flows are deleted after a timeout of 60 seconds. In this experiment 84% of the packets and 91% of the bytes in the trace are recognized as suitable for switching. A mean of 92 flows per second are established after an initial startup phase of 60 seconds, with a 95th percentile of 116 flows per second. The average number of established flows in the flow table is 15,500.

The experiment was repeated with all packets classified for switching. Thus, a suitable flow must exist, or be established, for every packet. This simulates the purely connection-oriented approaches to IP over ATM. In this case a mean of 422 flows/s must be established with an average of 42,000 entries in the flow table. This

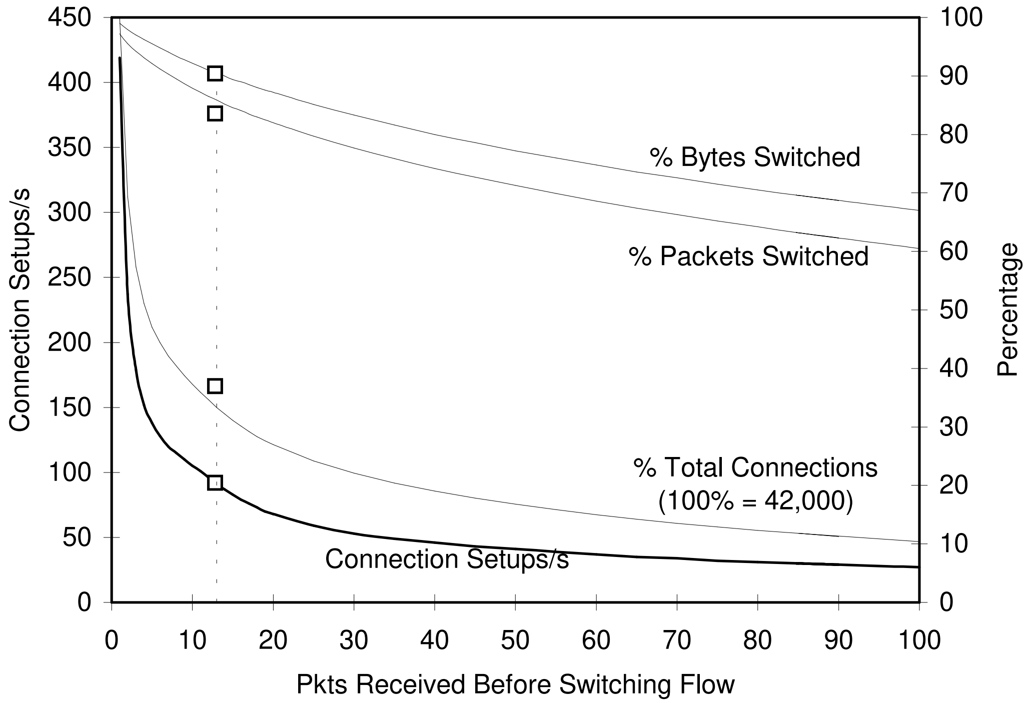

Figure 4: Variation in Switching Performance vs. Number Packets Received Before Switching

clearly demonstrates the advantage of connectionless forwarding for short lived flows.

The trace contains an average of 16,700 incoming packets per second of which, in the second experiment, about 14,100 are recognized as suitable for switching and the remaining 2,600 must be forwarded by the processor. We may assume that in an efficient implementation the work required to establish a flow is approximately equivalent to the number of packets that must be sent to set it up and tear it down. This we may assume to be about 9 packets plus the original packet that must be forwarded before the flow is established. Thus, the amount of work required to establish 92 flows is approximately equivalent to the work required to forward 920 packets. So, if flows are established for all of the packets marked as suitable for switching, 16,700 packets per second may be handled with an amount of work equivalent to that required to forward 3,520 packets per second. By adding the ATM switch we are able to handle approximately 4.5 times more traffic. This is a very encouraging result given that we are looking at Internet backbone traffic.

In a further experiment we investigated an alternative method of selecting long duration flows. In this method the number of packets on each flow is counted and when it reaches a threshold the flow is switched. Fig. 4 presents the results. The x-axis gives the threshold, the number of packets observed on a flow before switching the flow, and the y-axes give the number of connection setups/s observed and the percentage of packets switched, bytes switched and total table size. We see that the number of connection setups/s and the total table size falls rapidly as the threshold is increased beyond 1 but the percentage of packets and bytes switched decreases much more slowly. A value of 13 for the threshold yields the same number of connection setups/s as the previous method, selection by protocol type, but gives slightly better performance for total connections and packets switched. The threshold gives a very simple but effective parameter to control the ratio of flows forwarded to flows switched. It allows the number of connection setups per second and the connection table size to be adjusted to the capabilities of the hardware and to the average traffic mix.

To investigate the optimum setting of the threshold parameter we defined a cost function. The cost function is the total equivalent work done by the processor, which is the sum of the average number of packets forwarded per second by the processor and the average number of connections setup per second. If we define a single unit of work to be that required by the processor to forward one packet per second, fig. 5 shows the variation of the cost function with the threshold parameter for various assumptions of the equivalent work required to set up one

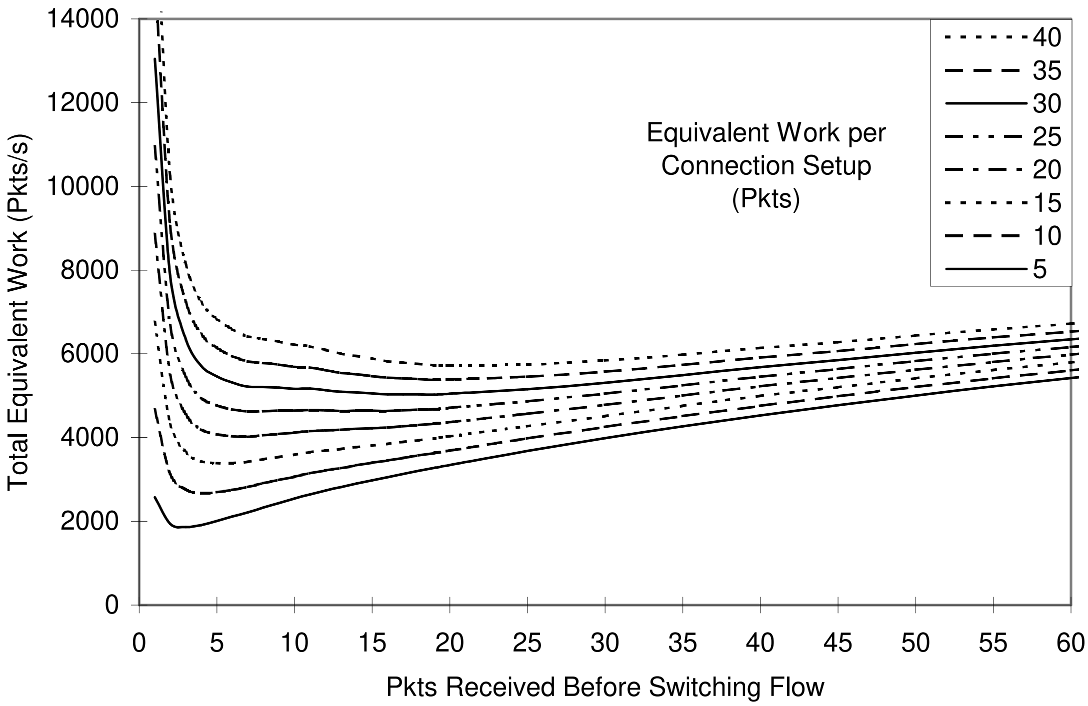

Figure 5: Cost Function of Total Equivalent Work vs. Packets Received Before Switching Flow

connection per second. Depending on the assumption of the amount of work required to set up a connection we see a minimum in the cost function for a threshold parameter between about 3 and 20. As the threshold parameter is increased, all of the curves will rise asymptotically to 16,700 which is the average number of packets per second in the trace. The cost function curves also give some idea of the gain that comes from using flow switching because the total average work required for a solution using only packet forwarding is 16,700.

7. Conclusion

Growth in IP traffic on the Internet and private enterprise networks is beginning to stress the traditional, processor based, design of current day routers. Switching technology offers much higher aggregate bandwidth but currently only offers a bridging solution. Various proposals are under way to support IP routing over ATM switching technology, however, these proposals hide the real network topology from the IP layer by treating the data link layer as a large opaque cloud. This leads to complexity, inefficiency and duplication of functionality. The cloud approach is untested and its scaling properties unproven, yet the Internet is proof that IP can scale to very large networks without requiring the concept of a cloud at the data-link layer.

We have proposed a connectionless approach to integrate IP with fast ATM switching hardware. The IP routing decision is cached as soft state in the ATM switch such that future packets belonging to the same flow may be switched in hardware rather than forwarded by software. We believe that this approach combines the simplicity and robustness of IP with the speed and capacity of ATM.

Simulation results using a traffic trace from the Internet backbone indicate that for this trace 84% of the packets and 91% of the bytes were recognized as suitable for switching. An average of 92 connection setups per second were required with an average number of established connections in the table of 15,500. This is well within the capabilities of current ATM switch hardware. A simple method of varying the ratio of packets forwarded by the processor to those switched by the switching hardware has been demonstrated. This allows the number of connection setups per second and the connection table size to be adjusted to the capabilities of the hardware and to the average traffic mix. In contrast, a purely connection-oriented approach, such as that proposed in current standards efforts for IP over ATM, required over 400 connection setups per second and an average connection table size (equivalent to the number of VPI/VCIs) in excess of 40,000 to handle a traffic load of a mere 36 Mbps. This is well beyond the number of calls per second of which the signalling is capable in many current ATM switches.

Assuming an efficient implementation this suggests that for our connectionless ATM approach, with this traffic trace, the addition of an ATM switch could increase the traffic capacity of the routing software by up to 4.5 times. For campus and enterprise backbone traffic the increase in capacity is likely to be much higher. While one cannot dwell too heavily on the results from a single traffic trace this is certainly a very encouraging result.

Acknowledgement

We would like to thank Bob Hinden, Fong-Ching Liaw, and Eric Hoffman, for their contribution to the architecture presented here.

References

[Alles95] A. Alles, “ATM Internetworking,” May 1995, <http://www.cisco.com>.

[Claffy95] K. C. Claffy, H.-W. Braun, and G. C. Polyzos “A parameterizable methodology for Internet traffic flow profiling,” IEEE J. Selected Areas in Commun. 13(8), Oct. 1995, 1481–1494.

[Clark88] D. D. Clark, “The design philosophy of the DARPA Internet protocols,” Proc. ACM SIGCOMM, Comp. Commun. Review 18(4), Aug. 1988, 106–114.

[Clark92] D. D. Clark, S. Shenker, and L. Zhang, “Supporting real-time applications in an integrated services packet network,” Proc. ACM SIGCOMM, Comp. Commun. Review 22(4), Sep. 1992, 14–26.

[Cole95] R. G. Cole, D. H. Shur, and C. Villamizar, “IP over ATM: A framework document,” IETF Internet Draft, draft-ietf-ipatm-framework-doc-03.ps, Jun. 1995.

[Deering95] S. Deering, R. Hinden, “Internet protocol, version 6 (IPv6),” IETF Internet Draft, draft-ietf-ipngwg-ipv6-spec-02.txt, Jun. 1995.

[ENCAPS] “The transmission of flow labelled IPv4 on ATM data links,” <http://www.ipsilon.com>.

[Floyd95] S. Floyd and V. Jacobson, “Link-sharing and resource management models for packet networks,” IEEE/ACM Trans. Networking, 3(4) Aug. 1995, 365–386.

[GSMP] “The GSMP protocol specification,” <http://www.ipsilon.com>.

[IFMP] “The IFMP protocol specification for IPv4,” <http://www.ipsilon.com>.

[LANE] “LAN emulation over ATM,” The ATM Forum, version 1.0, Jan. 1995.

[MPOA] C. Brown, “Baseline text for MPOA,” ATM Forum/95-0824r5, Jan. 1996.

[NARP] J. Heinanen and R. Govindan, “NBMA address resolution protocol (NARP),” IETF RFC 1735, Dec. 1994.

[NHRP] D. Katz and D. Piscitello, “NBMA next hop resolution protocol (NHRP),” IETF Internet Draft, draft-ietf-rolc-nhrp-04.txt, May 1995.

[Rekhter95] Y. Rekhter and D. Kandlur, “IP architecture extensions for ATM,” IETF Internet Draft, draft-rekhter-ip-atm-architecture.txt, Jan. 1995.

[rfc1577] M. Laubach, “Classical IP and ARP over ATM,” IETF RFC 1577, Jan. 1994.

[rfc1620] B. Braden, J. Postel, and Y. Rekhter, “Internet extensions for shared media,” IETF RFC 1620, May. 1994.

[rfc1633] R. Braden, D. Clark, and S. Shenker, “Integrated services in the Internet architecture: An overview,” IETF RFC 1633, Jly. 1994.

[RSVP] R. Braden et al., “Resource ReSerVation Protocol — Version 1 functional specification,” IETF Internet Draft, draft-ietf-rsvp-spec-05.ps, Mar. 1995.

[Thyag95] A. S. Thyagarajan, S. L. Casner, and S. E. Deering, “Making the Mbone real,” Proc. INET, Honolulu, Jun. 1995, 465–473

[Wake94] I. Wakeman et al., “Implementing real time packet forwarding policies using streams,” Usenix 1995 Technical Conference, New Orleans, Jan. 1995, 71–82.

[Zhang93] L. Zhang et al., “RSVP: A new resource ReSerVation Protocol,” IEEE Network Mag., Sep. 1993, 8–18.