IP Switching: ATM Under IP*

Peter Newman, Greg Minshall, and Tom Lyon†

Ipsilon Networks‡

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 layer-2 bridging solution. Various proposals are under way to support IP routing over an ATM network. 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 end-to-end ATM connection 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

Asynchronous Transfer Mode (ATM) has received much attention because of its high capacity, its bandwidth scalability, and its ability to support multiservice traffic. However, ATM is connection-oriented whereas the vast majority of modern data networking protocols are connectionless. 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. Research suggests that IP is no less capable of supporting real-time and multimedia applications than ATM. Also, much attention is being focused on the use of IP multicast for multimedia and conferencing applications [Turletti96, Perkins97].

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 end-to-end ATM connection and couple fast ATM switching hardware directly to IP. This has the particular advantage of not requiring end-to-end signalling, or address resolution, and requiring only the standard IP routing protocols. Our approach directly supports IP multicast, and because it is based upon IP, it is easily integrated into existing networks.

In section 2 we introduce the motivation for this work. Section 3 considers the difficulties that arise if the physical network topology is hidden from the internetworking layer. The concepts of flows and soft-state are discussed in section 4. Section 5 presents IP switching in detail; section 6 gives simulation results based upon several traffic traces from the Internet and one from a corporate backbone; and section 8 discusses related work.

2. IP: Necessary and Sufficient …

The Internet is growing rapidly. However, the traditional design of packet switched router on which the Internet is based is beginning to run out of steam. Existing routers are expensive and of limited throughput when

*Preprint of IEEE/ACM Transactions on Networking, 6(2), April 1998, pp. 117–129.

†{pn,pugs}@ipsilon.com, minshall@acm.org

‡ <http://www.ipsilon.com>

compared to switches. For example, a well-known manufacturer of both routers and ATM switches currently charges $187,000 for a 1 Gbps router and $41,000 for a 5 Gbps ATM switch. Thus, in this example, for the same aggregate throughput, routing costs over 20 times as much as switching. Efforts are under way to implement IP routers at speeds commensurate with that of current ATM switches. However, the greater complexity of IP forwarding is likely to ensure that routing remains more expensive than switching for some time.

ATM offers scalability of both link bandwidth and switch capacity. It is well suited to the application of VLSI implementation. Also the widespread 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 standardized and available, it is fast, and the price tag is falling. However, ATM is a connection-oriented switching technology.

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, a substantial fraction of the signalling code is there to handle error conditions. This code is very difficult to thoroughly test under all possible conditions. A soft-state approach, where state in the network is periodically refreshed, covers many possible error conditions using a very simple recovery mechanism. Also, in current switches, there is significant delay involved in connection establishment and a limited number of connection setups per second available [Niehaus97]. This motivates us to explore the use of ATM switching hardware without requiring end-to-end connections.

3. Obscured by Clouds

ATM is connection-oriented. So the heart of the problem is to make use of the speed and capacity of the switching hardware, without sacrificing the scalability and flexibility of connectionless IP. A number of approaches to the implementation of IP over an ATM network have been proposed in the literature and in the standards bodies [Alles95]. These include: LAN emulation [LANE]; classical IP over ATM [rfc1577, rfc1932]; address resolution (NARP) [rfc1735] and next hop resolution protocols (NHRP) [NHRP]; and Multiprotocol Over ATM (MPOA) [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 require their own 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 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 [rfc1932].

All routers in a cloud are interconnected physically by the link-layer. However, because the number of routers in the cloud may be very large, the next-hop routing tables can become unmanageable. For N routers in the cloud, the routing table size, the routing update traffic, and the routing update processing all grow with N^2. The conventional solution is to overlay a logical network in which not all the routers are interconnected. Thus, datagrams between two physically adjacent routers may be required to traverse multiple hops. At the very least this is inefficient. Worse still, a complex protocol is required to maintain connectivity when links or routers fail.

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. In a cloud environment, however, the cloud can be of arbitrary size and topology. So a router cannot easily discover its neighbors. They must be assigned by manual configuration. 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 datagrams between a source machine (or application process) and one or more destination machines (or application processes). The datagrams all follow the same route through the network, and all receive identical service policies at the routers [rfc1883, RSVP]. A flow in a connectionless network is very similar to a connection in

a connection-oriented network. If sufficient temporal locality exists in the traffic, a router can cache the routing decisions for a flow, hence accelerating the forwarding process. 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. An efficient mapping of IP onto ATM must consider the characteristics of the application and the transport protocol in deciding whether to establish an end-to-end ATM connection on behalf of any specific flow [rfc1932]. 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 will best be handled by connectionless packet forwarding between IP routers connected via shared, pre-established ATM connections. 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 would result in unnecessary delay for 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 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 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. The cost of this simplicity is that the messages required to maintain the soft-state might in some cases impose a significant load on the network controllers.

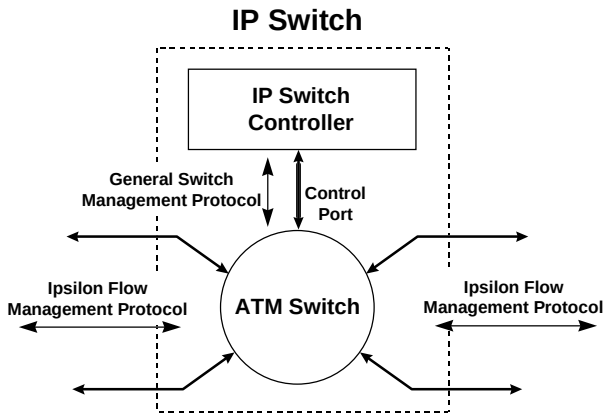

Figure 1: Structure of an IP Switch

5. IP Switching

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 General Switch Management Protocol (GSMP) [rfc1987], to give the IP switch controller access to the switch hardware. The IP switch controller is a high-end processor running standard IP router software 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 circuits, a flow classifier to decide whether to switch each flow, and GSMP to control the switch hardware.

At system startup, a default ATM virtual circuit, on a well-known VPI/VCI, is established between the IP software running on the IP switch controller and that of each of its neighbors. The default channel is used for the hop-by-hop connectionless forwarding of IP datagrams. To connect IP switching networks across a public ATM network, a virtual path may be established across the public network with a configured VPI. Thus, we 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 virtual circuit.

5.1 Flow Classification

We characterize an IP flow according to the fields in the IP/TCP/UDP header that determine the routing decision including: type of service, protocol, source

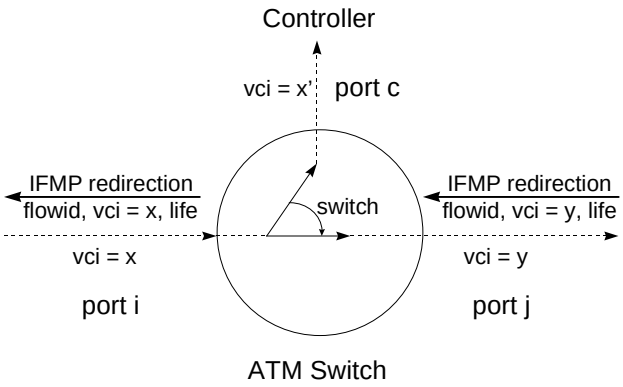

Figure 2: Establishing a switched flow

address, destination address, source port, and destination port. 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.

Currently, we have defined two flow types: a port-pair flow type (Flow Type 1) and a host-pair flow type (Flow Type 2). The port-pair flow type is for traffic flowing between the same source and destination TCP/UDP ports on the same source and destination IP addresses. The type of service, protocol, and time to live fields must also be the same. The port-pair flow type allows quality of service differentiation between flows with the same source and destination. The host-pair flow type is for traffic flowing between the same source and destination IP addresses with the same time to live field.

When a packet is received across a default channel it is reassembled and submitted to the IP switch controller for forwarding. The controller forwards the packet in the normal manner, but it also performs a flow classification on the packet. The flow classification determines 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 IP switch controller.

Flow classification is a local decision. The flow classifier inspects the contents of the fields that characterize the flow, and makes its decision based upon a local policy. For example, a flow classifier might look for well-known source or destination port numbers to identify the application. Flows belonging to FTP data connections might be configured to be switched, but DNS queries could be forwarded as datagrams. Another example of a flow classifier might count the number of packets received on each flow. If the number of packets received within a specified time interval exceeds a threshold the flow is switched. (The performance of these flow classifiers is investigated in the following simulation study.)

5.2 Flow Management Protocol (IFMP)

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

After making the entry in the translation table of the switch input port (via GSMP) the IP switch controller sends an IFMP [rfc1953] redirection message upstream to the previous node. The redirection message contains the label (VCI = x), a flow identifier, and a lifetime. The flow identifier contains the set of header fields that specifies the flow. The redirection message requests the upstream node to transmit all further packets with header fields that match the flow identifier, on the ATM virtual circuit specified by the label field (VCI = x). The lifetime field specifies the length of time for which this redirection is valid. Unless the flow state is refreshed, this binding of flow and label should be deleted when the lifetime expires.

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 accelerated because the previous routing decision for this flow was cached and is indexed by the label x′.

The real benefit of switching comes when the downstream node also redirects the flow to a specific VCI. While the flow labelling process runs independently on each link, the flow classification policy must be consistent within an administrative domain. So the downstream node is likely to redirect the flow at almost exactly the same time. When the IP switch 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 IP switch does this by instructing the switch to map label x on port i to label y on port j. Thus, traffic on this flow is no longer processed by the IP switch controller in a store-and-forward manner, but is switched directly to the required output port.

When traffic is cut over to the switched path, it is possible for packet misordering to occur. The first packet on the switched path may be delivered to the destination before the last packet on the store-and-forward path. This

possibility is investigated in [Lin97] and it is reported that for a single IP switch, packet misordering is about as likely to occur as in current networks. Because packets are transmitted as cells it is also possible that the cut over to the switched path will occur in the middle of a packet. This will result in the loss of that packet. To avoid this cause of packet loss we have biased our implementation to establish the switched path backward from the destination through the network towards the source. This can be achieved simply by making the flow classification algorithm at the destination end system likely to make the switching decision before nodes within the network. Thus the switched path is established while traffic is still flowing over the store-and-forward path, and the source may cut the traffic over to the switched path on a packet boundary.

IFMP may be viewed as a signalling protocol. However, it runs independently on each link and simply associates a local label with an IP flow. As such, it is much less complex than an end-to-end, hard state signalling system with network-wide addressing. Implementation experience indicates that Q.2931, the ATM signalling protocol, requires at least 10 times as many lines of source code as IFMP.

When an IP switch accepts a redirection message, it also changes the encapsulation it uses for packets belonging to the redirected flow. The encapsulation used for IP packets on the default 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 circuit removes all of the IP header fields specified by the flow identifier from the header of each packet [rfc1954]. The IP packet with the resulting compressed header is then encapsulated in AAL-5 and transmitted on the specified virtual circuit. The fields that are removed are stored by the router that issued the redirection and are associated with the specified ATM virtual circuit. The complete packet 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 simple flow-based 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 submitting packets with a different header to gain access to a prohibited destination.

5.3 TTL and Header Checksum

One of the basic requirements of IP is that the Time To Live (TTL) field of the IP header in a packet be decremented at each node. If the TTL reaches zero, the packet must be discarded (and an ICMP Time Exceeded error message must be sent to the source of the packet).

We include the TTL in the flow identifier to ensure that a packet exits a switched flow with the same TTL that it would have if it were forwarded hop-by-hop. This ensures that only packets with a single, specific, TTL value may be included in a switched flow. It also ensures that a packet with a TTL of zero will never be switched through a node. Thus at the end of a switched flow, the TTL of packets on that flow must be correct; the TTL field is not transmitted in the packet, but is recovered from information stored at the destination. The price of this solution is an increase in the number of flows created, as two packets that are identical, except for the value of the TTL field, will be transmitted in two separate flows.

In order to preserve the value of the header checksum, the value of the TTL field is subtracted from the header checksum of packets at the origin of a switched flow. (Any node where packets arrive non-switched and depart switched is the origin of a switched flow.) The header checksum is reconstructed at the end of a switched flow by adding the value of the TTL field to the checksum when the packet header is reconstructed. This operation is necessary because the number of upstream IP switch nodes is unknown at the destination of a switched flow, and may indeed change if more upstream IP switches decide to switch a particular flow. The effect of this operation, assuming no errors are introduced in the transmission path, is that the header checksum contains the value it would have contained had the packet been forwarded hop-by-hop (and the TTL decremented and the checksum updated by all the upstream nodes). Any errors introduced into the IP header along the transmission path will result in an incorrect header checksum, because the subtraction and addition is performed in an error preserving way. This will cause the packet to be rejected the next time the header checksum is checked.

5.4 Robustness

It is important that the protocol be robust in the face of network or node failure. While we strive to avoid introducing inconsistent state into the network, we also seek to limit the amount of time that bad or inconsistent state exists. The specific hazard our design introduces in the network is that an upstream node may be sending a flow on a label that is unknown to the downstream node. There are two cases: first, the downstream node thinks a different flow is being transmitted on that label; second, the downstream node thinks no flow is being transmitted on that label. In the first case, packets sent on the label will be mis-routed and corrupted; in the second case, packets will be dropped at the downstream node.

In order to reduce this failure from being introduced into the network, IFMP includes a simple adjacency protocol. This protocol enables a node to be sure that its

neighbor has not changed. IFMP messages are only sent or processed once adjacency has been established. If adjacency is lost, all state associated with flows redirected over the link is cleared. In addition, redirection messages are sent with monotonically increasing sequence numbers. This allows the message receiver to process messages in the correct order.

We recognize that even with the adjacency protocol, two neighboring nodes might still occasionally contain inconsistent state. We attempt to limit the lifetime of such bad or inconsistent state in the network by associating lifetimes with redirection messages. After the lifetime expires, upstream nodes are required to stop sending messages on the redirected label. Typically, lifetimes are on the order of one or two minutes; these values correspond to typical flow lifetimes in the Internet [Claffy95].

Each IP switch controller periodically examines every flow. If a flow has received traffic since the last refresh period, the controller sends another redirect message upstream to refresh the flow. If a flow has received no redirect messages for a period in excess of its lifetime, it is removed. This may 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 if the free label space is close to exhaustion.) For flows that are labelled, but not switched, the IP switch controller 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 controller must query the switch hardware to discover whether a specific channel has recently been active.

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 state related to a changed route will be invalidated and flushed. Affected traffic will once again be forwarded over the default channel and new virtual connections will be rebuilt from the point of failure across the new path. Thus the connections are repaired only within the locality of the failure. This is far more efficient than deleting all affected connections and establishing them afresh, end-to-end, as is the case for connection-oriented networks.

While the routing protocols are converging to a consistent state, it is possible for temporary routing loops to exist. A switched flow that is established while a loop exists in the routing state may traverse the same link multiple times, using a different VCI each time. Eventually the TTL will be decremented to zero and the flow will not be switched further. When the routing state becomes stable, a switched loop will be detected and removed. While a switched loop exists, it consumes VCIs that could have been used for other traffic. However, to detect a looped flow would require global path information in IFMP. It is not possible to implement loop detection by noting the presence of previously switched flows that differ only in TTL, because this situation could arise from causes other than routing loops. Load sharing or route flaps upstream of the IP switch could cause this situation, as could applications that select different initial TTL values.

5.5 Receiver Initiated

One of the reasons for the success of the connectionless, datagram model used by IP is the relatively weak coupling between adjacent nodes. This decoupling permits substantial autonomy for nodes, and allows them to make decisions based on local policy, yet still provide the basic internetwork service. To preserve this local autonomy, 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 circuit will indicate acceptance of the request.

From a purely theoretical point of view, either neighbor (upstream or downstream) could choose the label to be used for sending redirected packets. However, in IFMP, the downstream node chooses the label and sends the redirection message. There are two reasons for this decision. First, a flow can be reliably redirected with just a single message. An acknowledgment would be required if the upstream node chose the VCI. Second, in at least the case of today's ATM hardware (adapter cards and switches), it is common that the ability to receive on a label is a more precious resource than the ability to transmit on a label. Thus, it seemed prudent for the receiver to choose the label.

Upstream label allocation can simplify label allocation for multicast connections because the same label may be used on each output branch. However, most current ATM switching hardware is capable of separately labelling each output branch. To take care of the case where the upstream node is only able to transmit packets within a certain range of labels, we have provided a label range message. This message is used to inform the downstream node of the available label range.

5.6 Point-to-Point

LAN emulation and classical IP over ATM, etc. seek to establish a logical shared medium network model on top of an ATM network. 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. Also, the IFMP protocol is symmetric in that no distinction is made between a user interface (UNI) and a network interface (NNI). This leads to a very simple implementation.

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 by observing 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.

5.7 Multicast

An IP switch can support IP multicast without any modification to the Internet Group Management Protocol (IGMP) or the multicast routing protocols. Flow redirection proceeds in exactly the same manner as for unicast traffic. At an IP switch, when an incoming multicast flow is replicated into a number of branches, each branch may be individually redirected by a downstream neighbor. If an incoming multicast flow is labelled, the hardware multicast capability of the ATM switch can be used to replicate packets on those output branches that have been redirected by a downstream neighbor. The switch can also send a copy of the multicast flow to the IP switch controller, if there are branches that have not redirected the flow. Such branches may receive their copies of the traffic from the IP switch controller over the default 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 circuit from every sender that has recently transmitted traffic to the group.

5.8 Quality of Service

Quality of service may be considered to be a means of forwarding packets from different flows with controlled unfairness. Some packets receive faster service than others according to some established policy. Typically, this policy is established by a user requesting specific resources from the network using some form of signalling protocol. This is contract-based quality of service, as the user makes a contract with the network to reserve the specified resources. Q.2931 is the signalling protocol used in ATM to request resource reservation from the network. RSVP is the corresponding signalling protocol used in IP [Zhang93, White97, RSVP]. This approach to the provision of differentiated quality of service assumes that the user (or application): understands what resources are required; is able to describe the resources required; has a signalling protocol to request the desired resources; and can be trusted. This is not necessarily the case, particularly for existing IP applications. An alternative approach is to use policy-based quality of service differentiation.

In policy-based quality of service, the quality of service requirements for different flows are established by policy within the administrative domain. This policy is established by the network manager (or service provider). Policy-based quality of service differentiation will not provide the same control granularity that is available from the contract-based approach, but it does offer the network manager significant capabilities to control the use of resources in the network. Specific users can be limited to a specified maximum bandwidth. For example, this capability would permit an Internet service provider to equate dollars per month in access fee to bits per second of access rate. Interactive applications can be given higher priority than bulk transfer applications. The bandwidth available to multimedia applications can be constrained to avoid overloading network links.

IP switching can support both policy-based and contract-based quality of service. Each IP switch can make a policy-based quality of service decision, according to the policy established within the administrative domain. Each flow is classified as part of the forwarding operation, and quality of service information may be included in the flow classification decision. This decision may be based upon any of the fields within the packet header, for example: the application (specified by the TCP/UDP port numbers), the type of service field, the source and destination IP addresses, etc. Each IP switch must interpret the quality of service policy according to 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, Zhang95].

Contract-based quality of service requests for individual flows may also be supported using the resource

reservation protocol (RSVP). RSVP allows an application to specify the required service and 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 may be reserved by configuring the queueing and scheduling hardware within the ATM switch, or in software within the IP switch controller. Also, the flow may be policed by configuring the policing hardware in the ATM switch according to the flowspec.

6. Simulation Results

To investigate the benefit of our approach, we obtained several packet traces from the Internet[§]. The first 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. The trace includes a timestamp, IP source and destination addresses, the packet length, and source and destination port numbers for each packet. The other two traces were each of 10 minutes duration taken from the same location on Feb 28, 1996 at 8:30am and 1:45pm. These traces contain a timestamp and the full IP and TCP/UDP header.

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. With a flow timeout of 60s, and total trace duration of only 5 or 10 minutes, it is difficult to make an accurate estimate of the average flow duration. Some flows will already be in progress when the trace begins, and some will still be in progress when the trace ends. The effect will be to slightly underestimate the amount of traffic switched, because flows that were already in progress when the trace started are assumed to begin at the start of the trace. To avoid overestimating the number of new flow arrivals/s, we waited 60s before counting the flow arrival rate. This period was the approximate time required for the number of new flow arrivals/s to decline to a stable value.

We performed a simple flow analysis on the Sep. 1995 trace to select those protocols with traffic characteristics suitable for switching. The results are reported in [Newman96], and the set of protocols in the trace deemed worth switching is given in Table 1. 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 threshold selected by inspection of the trace).

Protocol	port
IP in IP	
TCP ftp-data	20
TCP telnet	23
TCP gopher	70
TCP http	80
TCP nntp	119
TCP netbios	139
TCP login	513
TCP cmd	514
TCP audio	1397
TCP AOL	5190
TCP X-11	

Table 1: Protocols Selected for Switching

We first investigated the performance of the Sep. 1995 trace when the decision to switch a flow was based upon its protocol. If a packet belonged to a protocol listed in Table 1, it was classified as suitable for switching. If neither the source nor destination port numbers were well known (less than 1024 or a recognized registered number) we assumed the packet was suitable for switching if it belonged to TCP but not if it was UDP. (This was the best guess we could make for packets that did not have a recognizable port number.) For those packets classified for switching we checked to see if a suitable flow existed. If the search failed a flow was created. In this experiment, a flow was suitable if it had the same source and destination IP address as the packet being processed. (The definition of Flow Type 2 also requires the TTL field to match, however, the TTL field was not preserved in the packet trace. This will cause the number of flows to be underestimated, but will not affect the estimate of the ratio of traffic switched to traffic forwarded.) Flows were deleted after a timeout of 60 seconds.

In this experiment, 84% of the packets and 91% of the bytes in the trace were recognized as suitable for switching. A mean of 92 flows per second was observed 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 was 15,500.

The experiment was repeated for the Sep. 1995 trace with all packets classified for switching. Thus, a suitable flow must exist, or be established, for every packet. In this case, a mean of 422 flows/s was observed with an average of 42,000 entries in the flow table. This clearly demonstrates the advantage of connectionless forwarding for short-lived flows.

The Sep. 1995 trace contains an average of 16,700 incoming packet/s. In the first experiment, about 14,100

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

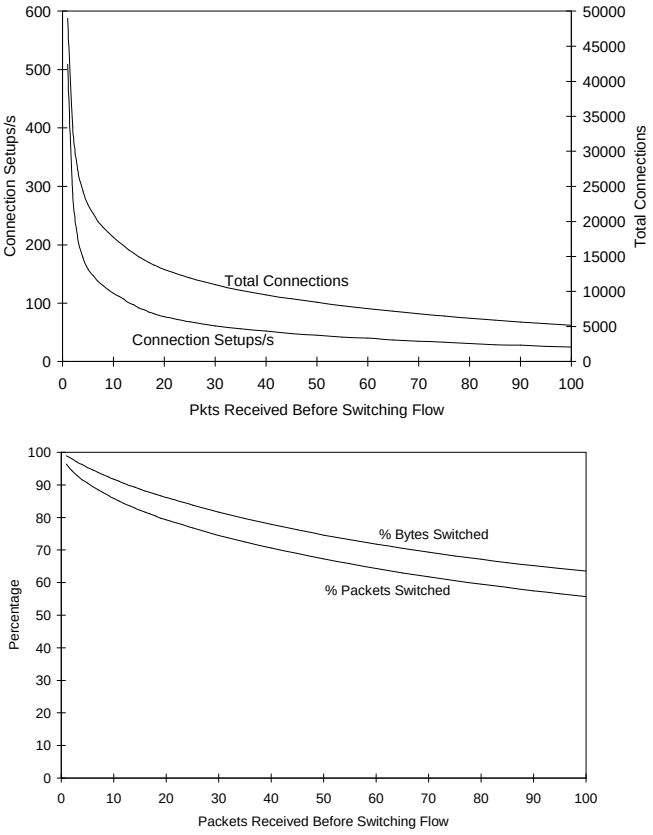

Figure 3: Variation in Switching Performance vs. Flow Classification Threshold

packets/s are recognized as suitable for switching, and the remaining 2,600 must be forwarded by the IP switch controller. From implementation experience, we find that it takes four packets to establish a flow and none to tear it down (if it is left to time out). Also, it takes two packets per flow for each refresh period to maintain a flow (due to the soft-state refresh messages). We currently use a refresh period of 20 seconds to allow some loss of redirect packets without causing the flow to be timed out.

Thus, to establish 92 flows/s and maintain 15,500 flows requires, on average, 1918 packets/s of control messages to be processed. Without IP switching we would not require this control traffic, and could forward approximately the same number of data packets instead. For IP switching, a total of 16,700 pps (packets per second) may be handled with an amount of work approximately equivalent to that required to forward 4,518 packets per second. By adding the ATM switch we are able to handle approximately 3.7 times more traffic.

Next, we investigated an alternative flow classification algorithm. In this method the number of packets on each flow is counted, and when it reaches a threshold, within a given time interval, the flow is switched. In these simulations the flow classification time interval was set to the same value as the flow timeout: 60s. Performance with respect to both threshold and time interval is investigated in [Lin97].

Figs. 3a and 3b present the results for the Sep. 1995 trace. The x-axis gives the flow classification threshold, i.e. the number of packets observed on a flow before switching the flow. Fig. 3a gives the number of connection setups/s observed and the total connection table size. Fig. 3b gives the percentage of packets switched and bytes switched against the flow classification threshold. We see that the number of connection setups/s and the total table size fall rapidly as the flow classification threshold is increased beyond 1, but the percentage of packets and bytes switched decreases much more slowly. A flow classification threshold of 13 yields the same number of connection setups/s as the previous method, selection by protocol type. It gives slightly better performance for total connections and packets switched. The flow classification 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. Also, it is much less sensitive to assumptions about the traffic mix, or the traffic characteristics of different applications, compared to selection by a static protocol table.

The above experiment was repeated with the two Feb. 1996 traces. The results are presented in figs. 4a and 4b, and are compared to the Sep. 1995 trace results. For comparison, the curves for the total number of connections and the number of connections/s have been normalized to the average amount of traffic (Mbps) in each trace. The difference in the total number of connections between the Sep. 1995 and the Feb. 1996 traces is, at least in part, due to the difference in the length of these traces, 5 minutes and 10 minutes. A trace of 5 minutes duration was not quite long enough for the total number of connections in the connection table to reach stability, whereas the 10-minute traces could be observed to reach stability after about 7 minutes.

The difference in the percentage of packets switched in fig. 4b, between the Sep. 1995 and the Feb. 1996 traces, is mainly due to a change in the traffic profile. The proportion of TCP packets has dropped by 20%, and the proportion of UDP packets has increased by 60% for the PM trace and 84% for the AM trace. Within TCP, the amount of http packets has grown from 40% to 48% of the entire trace while there has been a reduction in the older protocols of ftp-data and telnet. Within the range of interest of the flow classification threshold parameter, the

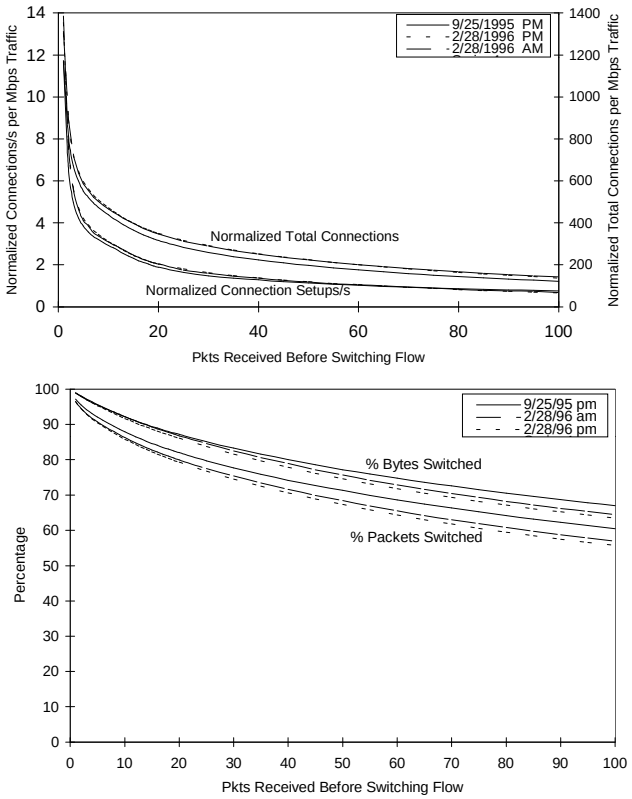

Figure 4: Variation in Switching Performance vs. Flow Classification Threshold for Multiple Traces

difference between the percentage of packets switched for the 1995 and 1996 traces is about 2.5%.

To further investigate the effect of the flow classification threshold parameter we looked at the switching gain. We define the switching gain to be the ratio of the average number of data packets transferred by the system to the average number of packets/s processed by the IP switch controller. The packets processed by the switch controller include both control messages and forwarded data packets. (As before, we assume four control messages to establish a flow, none to tear it down, and two control messages every 20 s to maintain a flow.)

Fig. 5 shows the switching gain for the three traces. With small values of the threshold parameter, the number of data packets forwarded is small, but the number of control messages is large due to the large number of flows established. With large values of the threshold parameter, the number of control messages is small, but there is a large number of data packets to forward. There is a maximum in the switching gain at a threshold parameter of between 5 and 7. The switching gain only varies by about 8% for a flow classification threshold within the range 3–18.

The difference in switching gain, between the 1995 and 1996 traces, is due to a 20% increase in the average

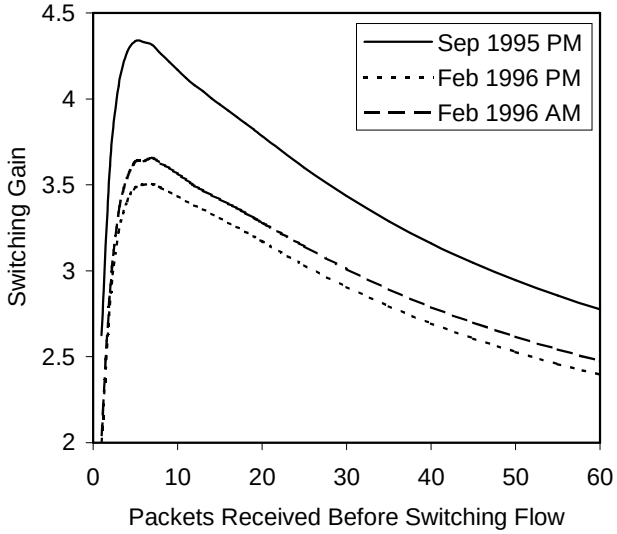

Figure 5: Switching Gain vs. Flow Classification Threshold

number of established flows, and a 15% increase in the number of flow setups/s. The increase in the number of established flows is in part due to the short sample size of the 1995 trace. This causes the average number of established flows to be underestimated. However, the large increase in UDP traffic between the 1995 and 1996 traces also contributes to the number of packets forwarded, or to the number of flows established, depending upon the value of the threshold parameter. The increase in UDP traffic is largely due to an increase in nameserver (DNS) traffic.

To investigate the performance of IP switching for campus and enterprise backbone applications, we obtained a packet trace from the corporate backbone of a large US-based manufacturer. The trace is of approximately 1.5 hours duration and contains an average of 528 packets per second at an average rate of 1.3 Mbps. The most active second in terms of packets saw 2,475 packets, giving a peak rate of 8.6 Mbps. The most active second in terms of bytes saw a peak rate of 8.9 Mbps in 2,374 packets.

Figs. 6a and 6b give the percentage of packets and bytes switched over time for the corporate trace. The flow classification threshold parameter is 10 packets, the flow timeout is 60s, and all switched flows are host-pair flows. The percentage of packets switched varies with time but lies mostly between 70 and 80 percent. The total number of connections in the table stabilizes at about 2,500 connections, and the number of connection setups/s after the connection table has reached stability is about 5 connection setups/s.

It is interesting to note that if we normalize the number of connection setups/s to the number of packets per second in the trace, we see 1 connection setup/s for every 105 packets per second in the corporate trace. In the Feb.

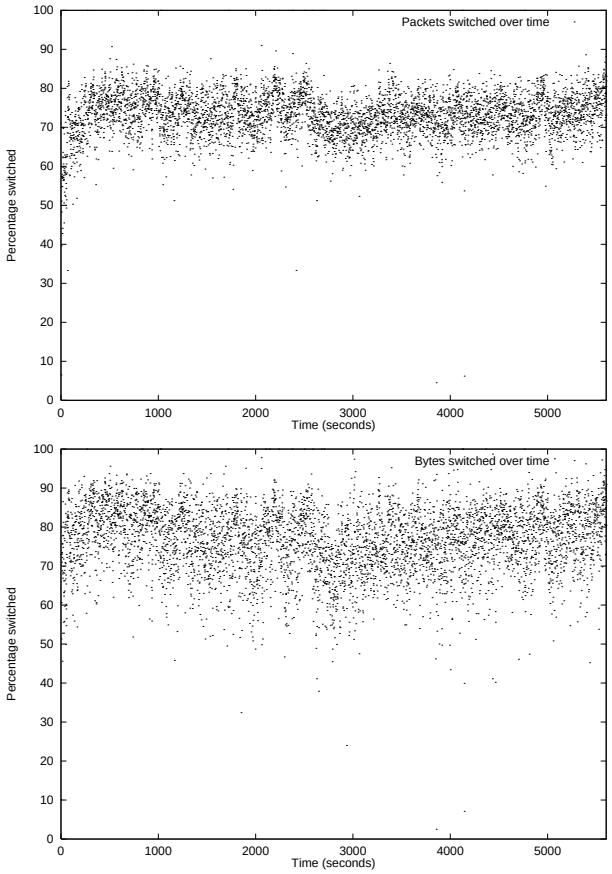

Figure 6: Packets and Bytes Switched over Time for Corporate Backbone Trace

1996 traces we see 1 connection setup/s for about every 130 packets per second. These connection setup rates are similar and reflect the average number of packets per connection in each trace. It is likely that the number of connection setups per second will grow in proportion to the traffic. To address the total number of connections, and the number of connection setups/s required, coarser granularity flows may be employed for traffic without specific QoS requirements. (Coarser granularity flows could not be investigated with the above traces as the actual IP addresses had been modified to protect privacy, and no information regarding the structure of the original addresses was available.)

7. Related Work

7.1 Gigabit Routers

An alternative approach to achieve IP packet forwarding at gigabit/s speeds is to implement a router with an internal switch fabric and the ability to perform layer-3 packet header processing at high speed. The MGR is an example of this approach [Partridge97]. The MGR uses a 15-port crossbar switch with each port operating at 3.3 Gb/s. The packet header processing is implemented in one or more forwarding engines. This gives the flexibility of allowing the packet-forwarding rate to be varied with respect to the aggregate switch capacity by changing the number of forwarding engines. However, it requires an overhead of about 25% of the switch fabric bandwidth to transfer packet headers between the interfaces and the forwarding engines.

Each forwarding engine of the MGR uses a 433 MHz general-purpose processor. The processor's internal cache is employed as a least recently used cache of about 8000 IPv4 destination addresses. An external memory of 8 Mbytes holds a complete routing table of several hundred thousand routes. The forwarding engine can forward 6.5 Mpps. It is estimated that the forwarding engine can maintain full performance with a minimum cache hit rate of about 60%. Under worst case conditions, where every packet receives a cache miss, the forwarding performance for average traffic degrades to about 50% of best case performance.

Other designs of forwarding engine have concentrated on IP packet header processing in hardware, to remove the dependence upon caching, and to avoid the cost of a high-speed processor. Designs based upon content-addressable memory have been investigated [Mcauley93], but such memory is far too expensive to be applied to a large routing table. Hardware tree walking is also under active investigation in both research and commercial designs [Zitterbart97, Pei92]. The argument for a software-based implementation stresses flexibility. Hardware implementations can generally achieve a higher performance at lower cost but are less flexible [Newman97].

Recently, new algorithms have been proposed for software-based implementation. Using general-purpose processors, they claim to achieve lookup speeds comparable to existing hardware-based approaches, yet without relying upon caching. One proposal reduces the forwarding table to a very efficient representation of a binary tree such that the majority of the table can reside in the level-2 processor cache [Deger97]. This method implements the forwarding table with a fixed 16 Kbytes plus a further 5 bytes per route prefix. Simulation results for a 200 MHz Pentium Pro processor, using a 40,000 entry routing table from the Internet, indicate a forwarding performance in the region of 2 Mpps.

An alternative approach attempts to reduce the number of memory references rather than compact the forwarding table [Waldvogel97]. This method uses an array of hash tables, one for each prefix length, with a binary search of the hash tables. Hints are included in the data structure to reduce the search time. About 36 bytes per prefix are

required. Simulation results for a 200 MHz Pentium Pro processor, using a 40,000 entry routing table from the Internet, indicate a forwarding performance in the region of 5–10 Mpps.

Layer-2 switching is less complex than packet forwarding at layer-3. This observation has led to a number of proposals that combine layer-2 switching for performance with IP network control for functionality and flexibility. These proposals include IP switching, IP/ATM [Parulkar95], the Cell Switch Router (CSR) [Esaki95, rfc2098, rfc2129], Ipsofacto [Acharya97], Tag Switching [rfc2105], and ARIS [Woundy96]. The Multiprotocol Label Switching (MPLS) working group in the IETF is working to combine these proposals into an IETF standard [MPLS].

7.2 Data-Driven Label Switching

IP switching, IP/ATM, the CSR, and Ipsofacto are similar proposals. All use data-driven label assignment where the binding of a label to a flow is initiated by the arrival of traffic on that flow. IP/ATM proposes the capability to support a distributed approach, with the equivalent of an IP switch controller on every line card of the switch. All inactive VCIs are mapped to the IP processing element on the line card. Packets may arrive unannounced on any inactive VCI. When a packet arrives on an inactive VCI: the VCI is marked active, the packet is buffered, IP processing determines the required output port, and a message is sent to that output port requesting an outgoing VCI. When a response is received from the output port, a path across the ATM switch is established, and all buffered packets and subsequent packets on the flow are forwarded across the switched path.

Transmitter selected, unannounced VCI selection works for IP/ATM because IP processing is available on every line card, and because the project is implementing its own segmentation and reassembly (SAR) device. This ensures that all inactive VCIs are mapped to an IP processing function with sufficient processing capacity. For a proposal with centralized IP processing and unannounced VCI selection, such as Ipsofacto, all inactive VCIs would need to be mapped to the single ATM switch port leading to the IP processor (IP switch controller). Such a design is likely to suffer from an inadequate number of VCIs available on the IP processor port, or in the SAR function of the IP processor, unless the switch is capable of VC merging. (VC merging allows multiple incoming VCs to be merged into a single outgoing VC without interleaving cells from different AAL-5 frames.) IP switching and the CSR proposal both use a protocol to bind a flow to a VCI, and therefore only those switched flows in the process of being established need a unique VCI to the switch controller.

IP/ATM buffers the packets on a new flow while establishing a switched path. IP switching and the CSR proposal forward packets on a new flow hop-by-hop until a switched path is established. IP/ATM is able to buffer the packets because it does not require an external binding protocol, it has distributed IP processing, and therefore it can assume a low delay in establishing the switched path. Another design of switch with routing capability on the line cards also proposes to delay all cells in a packet until processing is complete [Hoymany96]. However, this design simply forwards all packets hop-by-hop between switches capable of routing. It has no capability to dynamically establish switched paths on behalf of individual flows. Buffering the packets on a new flow avoids possible packet mis-ordering but may require special hardware support.

IP switching requires a point-to-point ATM link between IP switches. This link may be a Permanent Virtual Path (PVP) across an ATM network, but not a permanent or switched virtual channel (PVC or SVC). This is because IP switching expects to manipulate the VCI itself. The CSR proposal permits the use of PVCs or SVCs across an ATM network between CSRs. This allows CSRs to be interconnected across an ATM network, e.g. a classical IP-over-ATM subnet (RFC 1577) or a LAN emulation subnet. It requires the CSR to implement ATM signalling. Also, for an ATM virtual circuit between CSRs that traverses one or more ATM switches, the transmitter does not know the value of the VCI at the receiver. The CSR flow binding protocol (FANP) defines an additional message exchange for the transmitter to be informed of the VCI at the receiver. It is possible for switched virtual circuits to be established on demand between CSRs using ATM signalling. However, the typically high delay of ATM connection establishment implies that a pool of ATM connections is likely to be pre-established between CSRs and activated on traffic arrival.

7.3 Topology-Driven Label Switching

Tag Switching and ARIS propose topology-driven label assignment. Labels are allocated based upon information available from the routing protocols, and virtual connections are established in advance of traffic being received. This has the advantage of not requiring packets to be forwarded by the processor while establishing a virtual connection. Thus, there is no delay while establishing the link layer connection, and no possibility for packet mis-ordering when traffic is cut through to the new connection. Even very short flows can benefit from layer-2 switching, and flow classification is not required to distinguish between short- and long-lived flows. However, since labels are allocated and distributed without reference to traffic, labels will be allocated for

routes on which there is no traffic flowing. In a large network without the capability of VC merging, topology-driven label assignment will result in the establishment of a full mesh and will consume a large number of labels. For the same reason (and also because hosts typically do not participate in the routing protocol), topology-driven label assignment tends not to extend the switched path all the way to the hosts on the network.

Topology-driven label assignment is coupled to the operation of the routing protocol. Thus the granularity of the labels may be equivalent to the granularity of the routes advertised by the routing protocol. This allows the labels to cover highly aggregated routes. However, for switches without VC-merge capability, coupling the label assignment to the routing protocol can cause unnecessary tag information to be exchanged should the routing table change frequently (as in a route flap). There is no fundamental reason preventing data-driven label assignment from using labels with the same granularity as the appropriate entry in the forwarding table. However, for simplicity, data-driven label assignment has so far used a small set of fixed granularities.

The ARIS proposal differs from Tag Switching in that it builds a sink tree rooted at the egress node back throughout the network. This requires VC-merge capability in the switches or the use of both VPI and VCI for label assignment. (However, the most recent proposals for Tag Switching over ATM also suggest the use of a sink tree for switches with VC-merge capability.) The ARIS protocol also guarantees that VC loops are prevented even in the presence of transient IP routing loops. Offering such a strong guarantee of loop prevention requires a more complex label assignment protocol. It also implies that the label assignment process will take longer to converge.

7.4 Performance Investigation

A simulation study of IP switching using 9 different packet traces is reported in [Lin97]. The parameters investigated include the number of VCIs available, the flow creation delay, the flow timeout, the flow classifier, and the percentage of datagrams misordered. The results indicate that with an adequate number of VCIs, the amount of packets switched exceeds 80%. A flow timeout of 30s is recommended to reduce the number of VCIs required. The flow classifier parameters are investigated, and a value of 2 packets received within 60s is recommended if sufficient VCIs are available. The number of packets misordered as a result of establishing a switched path is reported to be less than 0.1%. This is about the same proportion of datagram misordering as seen in current datagram networks.

An analytical study of the flow classifier and flow timeout parameters is reported in [Che97]. This work investigates an adaptive flow classifier that adjusts the values of the flow classification threshold and the flow timeout in response to the arriving traffic. It attempts to balance the utilization of the system resources: the software packet forwarding, the number of VCIs available, and the number of flow setups/s. Adaptation of the flow classifier parameters in response to a changing traffic load is shown to offer better performance than static parameters. However, the cost of the adaptation algorithm, in terms of the additional load on the processor, is not reported.

8. 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 layer-2 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.

IP switching is 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 traffic traces from the Internet indicate that for these traces, with a flow classification threshold of 10 packets, 86% of the packets and 92% of the bytes may be switched in the ATM hardware using host-pair flows. An average of about 118 connection setups per second was required with an average of about 18,000 established connections. A brief investigation of a traffic trace from a corporate backbone also indicates that for host-pair flows, between 70% and 80% of the traffic may be switched. The threshold based flow classification algorithm offers a simple control to vary the ratio of packets forwarded by the IP switch controller to those switched by the switching hardware. This determines the number of connection setups/s and the connection table size required.

The results indicate that for IP switching, with these traffic traces, the addition of the ATM switch increases

the traffic capacity of the routing software by about 3.5 times. However, using host-pair flows results in a relatively large number of connections. To reduce the number of connections required, the definition of a flow type that offers higher aggregation than a host-pair flow must be investigated for the best-effort traffic.

Acknowledgement

We would like to thank Bob Hinden, Fong-Ching Liaw, Eric Hoffman, and Larry Huston for their contribution to the architecture and implementation presented here. Also, we would like to thank Nick McKeown, Steve Lin, and the anonymous reviewers for their detailed review and helpful comments on an earlier version of this paper.

References

[Acharya97] A. Acharya, R. Dighe, F. Ansari, "IPSOFACTO: IP switching over fast ATM cell transport," IETF Internet Draft, draft-acharya-ipsw-fast-cell-00.txt, Jly. 1997.

[Alles95] A. Alles, "Routing and Internetworking in ATM Networks," Proc. Engineer Conf., Networld+Interop, Las Vegas, Mar. 1995.

[Che97] Hao Che, S. Q. Li, and Arthur Lin, "Adaptive resource management for IP/ATM hybrid switching systems," submitted to IEEE Infocom '98 (available at www.ece.utexas.edu).

[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.

[Deger97] M. Degermark et al., "Small forwarding tables for fast lookups," Proc. ACM SIGCOMM, Cannes, Sep. 1997.

[Esaki95] H. Esaki, K. Nagami, and M. Ohta, "High speed datagram delivery over Internet using ATM technology," Proc. Engineer Conf., Networld+Interop, Las Vegas, E 12, Mar. 1995.

[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.

[Hoymany96] F. Hoymany and D. Mossé, "Switch-borne router for high performance packet forwarding of connectionless traffic in ATM networks," Proc. ICCCN, Washington, DC, Oct. 1996.

[LANE] "LAN emulation over ATM — LUNI Specification," The ATM Forum, Version 2.0, Jul. 1997.

[Lin97] S. Lin and N. McKeown, "A Simulation Study of IP Switching," Proc. ACM SIGCOMM, Cannes, France, Sep. 1997.

[Mcauley93] A. J. McAuley and P. Francis, "Fast routing table lookup using CAMs," Proc. IEEE Infocom, Vol. 3, San Francisco, CA, Mar. 1993, p. 1382–1391.

[MPLS] R. Callon et al., "A framework for multiprotocol label switching," IETF Internet Draft, draft-ietf-mpls-framework-00.txt, May 1997.

[MPOA] "Multiprotocol over ATM Specification," The ATM Forum, version 1.0, Jul. 1997.

[Newman96] P. Newman, T. Lyon, G. Minshall, "Flow labelled IP: A connectionless approach to ATM," Proc. IEEE Infocom, San Francisco, Mar. 1996, 1251–1260.

[Newman97] P. Newman, G. Minshall, T. Lyon, and L. Huston, "IP switching and gigabit routers," IEEE Commun. Mag., Jan. 1997.

[Niehaus97] D. Niehaus et al., "Performance benchmarking of signaling in ATM networks," IEEE Commun. Mag., Aug. 1997, 134–143.

[NHRP] J. V. Luciani, D. Katz, D. Piscitello, and B. Cole, "NBMA next hop resolution protocol (NHRP)," IETF Internet Draft, draft-ietf-rolc-nhrp-12.txt, Sep. 1997.

[Partridge97] C. Partridge et al., "A fifty gigabit per second IP router," submitted to IEEE/ACM Trans. Networking.

[Parulkar95] G. Parulkar, D. C. Schmidt, J. S. Turner, "IP/ATM: A strategy for integrating IP with ATM," In Proc. ACM SIGCOMM, Cambridge MA, Sep. 1995, page 49.

[Pei92] T. B. Pei, C. Zukowski, "Putting routing tables in silicon," IEEE Network Mag., Jan. 1992, 42–50.

[Perkins97] C. Perkins and J. Crowcroft, "Real-time audio and video transmission of IEEE Globecom '96 over the Internet," IEEE Commun. Mag., Apr. 1997, 30–33.

[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.

[rfc1735] J. Heinanen and R. Govindan, "NBMA address resolution protocol (NARP)," IETF RFC 1735, Dec. 1994.

[rfc1883] S. Deering, R. Hinden, "Internet protocol, version 6 (IPv6) specification," IETF RFC 1883, Dec 1995.

[rfc1932] R. G. Cole, D. H. Shur, and C. Villamizar, "IP over ATM: A framework document," IETF RFC 1932, Apr. 1996.

[rfc1953] P. Newman et al., "Ipsilon Flow Management Protocol Specification for IPv4," IETF RFC 1953, May 1996.

[rfc1954] P. Newman et al., "Transmission of Flow Labelled IPv4 on ATM Data Links," IETF RFC 1954, May 1996.

[rfc1987] P. Newman et al., "Ipsilon's General Switch Management Protocol Specification," IETF RFC 1987, Aug. 1996.

[rfc2098] Y. Katsube, K Nagami, and H Esaki, "Toshiba's router architecture extensions for ATM: Overview," IETF RFC 2098, Feb. 1997.

[rfc2105] Y. Rekhter et al., “Cisco Sytems’ tag switching architecture overview,” IETF RFC 2105, Feb. 1997.

[rfc2129] K Nagami et al., “Toshiba’s flow attribute notification protocol (FANP) specification,” IETF RFC 2129, Apr. 1997.

[RSVP] R. Braden et al., “Resource ReSerVation Protocol — Version 1 functional specification,” IETF Internet Draft, draft-ietf-rsvp-spec-16.ps, Jun. 1997.

[Turletti96] T. Turletti and C. Huitema, “Videoconferencing on the Internet,” IEEE/ACM Trans. Networking, Jun. 1996, 340–351.

[Waldvogel97] M. Waldvogel et al., “Scalable high speed IP routing lookups,” Proc. ACM SIGCOMM, Cannes, Sep. 1997.

[White97] P. P. White, “RSVP and integrated services in the Internet: A tutorial,” IEEE Commun. Mag., May 1997, 100–106.

[Woundy96] R. Woundy et al., “ARIS: Aggregate route-based IP switching,” IETF Internet Draft, draft-woundy-aris-ipswitching-00.txt, Nov. 1996.

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

[Zhang95] H. Zhang, “Service disciplines for guaranteed performance service in packet-switching networks,” Proc. IEEE, 83 (10), Oct 1995.

[Zitterbart97] M. Zitterbart et al., “HeaRT: High performance routing table lookup,” 4th IEEE Workshop on the Architecture & Implementation of High Performance Communications Subsystems, Thessaloniki, Greece, Jun. 1997.