IP Switching and Gigabit Routers*

Peter Newman, Greg Minshall, Tom Lyon, and Larry Huston†

Ipsilon Networks Inc.‡

To cope with the growth in the Internet and corporate IP networks we require IP routers capable of much higher performance than is possible with existing architectures. This paper examines two approaches to the design of a high-performance router, the gigabit router and the IP switch, and then provides some detail on the implementation of an IP switch and the protocols associated with IP switching.

The Internet is growing rapidly. The number of hosts on the Internet has doubled approximately every 56 weeks since 1989 [14] and the number of web servers has doubled at least every 23 weeks for the last three years [15]. As such growth persists, and as common access line speeds increase, we require IP routing capacity of many gigabits/s (Gbps) of aggregate traffic. Existing bus and central processor based architectures can handle a maximum load in the region of 1 Gbps and a few hundred thousand packets per second (kpps) but to get much beyond this requires alternative architectures. This paper examines two such approaches, the gigabit router and the IP switch, and then provides some detail on the implementation of an IP switch and the protocols associated with IP switching.

Gigabit Routers

A number of projects to provide high speed routing are under way, of which information is available for: the Multigigabit Router [1], IP/ATM [2], the Cell Switch Router (CSR) [3], and IP switching [4]. Also, the NetStar GigaRouter is a commercial implementation of a gigabit router [5]. These will serve to illustrate the basic approaches to designing for high speed routing. (For a more general discussion of routing and bridging see [16].)

All of the designs use the same functional components illustrated in fig. 1. The line card contains the physical layer components necessary to interface the external data link to the switch fabric. The switch fabric is used to interconnect the various components of the gigabit router. The forwarding engine inspects packet headers, determines to which outgoing line card they should be sent, and rewrites the header. The network processor runs the routing protocols and computes the routing tables that are copied into each of the forwarding engines. It handles network management and housekeeping functions and may also process unusual packets that require special handling.

A switch fabric is used for interconnection as it offers a much higher aggregate capacity than is available from the more conventional backplane bus. The Multigigabit Router will use a 15 port crossbar switch with each port operating at 3.3 Gbps. The NetStar GigaRouter also uses a crossbar switch fabric with 16 ports each operating at 1 Gbps. The IP/ATM, CSR, and IP switch solutions use asynchronous transfer mode (ATM) for the switch fabric. In the case of the IP switch a complete ATM switch, not just a fabric, may be used. This allows use of more highly integrated switch solutions, that, for example, integrate line card and switch fabric functionality. The advantage of an ATM switch is that the hardware is standardized and is available in many different sizes from different vendors with different cost/functionality tradeoffs. Additionally, advanced features such as hardware Quality of Service (QoS) support and hardware multicast are typically available in ATM switches. The disadvantage of an ATM switch is that it is cell, not packet, oriented and is

†`{pn,minshall,pugs,huston}@ipsilon.com`

‡ `<http://www.ipsilon.com>`

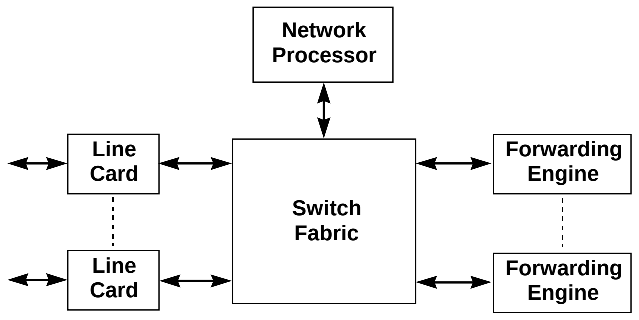

Figure 1: General structure of a high-speed router

connection oriented unlike the connectionless network protocols that are the subject of high-speed routing.

The forwarding engine may be a physically separate component or may be integrated with either the line card or the network processor. If the forwarding engine is a separate component the packet forwarding rate may be varied independently from the aggregate capacity by adjusting the ratio of forwarding engines to line cards. This is the approach taken in the Multigigabit Router and is an option in the IP/ATM solution. However, separating the line card and the forwarding engine creates additional overhead across the switch fabric. The NetStar GigaRouter integrates a forwarding engine with each line card. In the current realization of an IP switch the forwarding engine is combined with the network processor although combination with the line card or a separate implementation is not prohibited by the architecture.

A key difference between the router approach and the IP switching architecture is that IP switching allows most data between ATM ports to traverse the switch without being handled at all by a forwarding engine, whereas a router approach always requires use of at least one forwarding engine.

Measurements from the Internet indicate that the average packet size is now about 2000 bits [4]. This has increased from an average of 1000 bits just over five years ago because of the increase in large transfers due to web usage which now represents almost 50% of the Internet's traffic. Thus at present we need a forwarding rate of about 500 kpps for each 1 Gbps of traffic, though this may change as the traffic profile changes. Two approaches have been proposed to achieve packet forwarding rates of this magnitude: the silicon forwarding engine; and a high-speed general purpose processor with destination address caching using an on-chip cache.

Design of the Forwarding Engine

To build a forwarding engine in silicon we need a tree-structured routing table in memory and a tree walking ASIC [6]. Each IPv4 route in the table requires a minimum of about 16 bytes so for a large table, of say 250,000 routes, we require about 4 Mbytes of memory. This is within the realms of possibility for current SRAM. The number of memory accesses per route lookup is about $1 + \log N$, where N is the total number of routes in the table. So if we assume 10 ns SRAM, one full route lookup every 200 ns is possible. This gives us a forwarding engine capable of forwarding 5 million packets per second (Mpps), enough for an average of about 10 Gbps of traffic. Worst case it will handle 1.6 Gbps of 40 byte packets at wire speed. In addition, for large routing tables, techniques exist that can significantly reduce the number of memory references required.

The same hardware can be extended to handle multicast forwarding and more complex policy-based forwarding if some flexibility is provided in the fields from the packet header used as the key for the lookup.

The forwarding engine of the Multigigabit Router uses a 415 MHz general purpose processor with destination address caching using an internal (on-chip) cache. The internal cache is a least recently used cache of 9000 IPv4 destination addresses. An external memory of 8 Mbytes holds a complete routing table of several hundred thousand routes. This forwarding engine is capable of forwarding about 11 Mpps if all of the requested destinations are available in the cache. Multicast packets are handled by the full routing table rather than the cache as they require additional processing because the forwarding decision is based upon the source address as well as the destination (multicast) address. Additional processing is also required to offer firewall filtering, or other policy-based forwarding decisions. Packets with unusual options are sent to the network processor.

The design of the Multigigabit Router requires a performance from the forwarding engine of 6.5 Mpps at full speed for average traffic. It is estimated that the forwarding engine can perform at full speed 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.

There is an ongoing debate in the research community regarding the use of caching in a forwarding engine designed for a gigabit router. The question concerns whether there is sufficient locality in a stream of packets in the Internet for caching, with a moderate sized cache, to be useful. The silicon forwarding engine can maintain its maximum forwarding rate regardless of the past history of destination addresses in the traffic stream. For the caching solution in the Multigigabit Router to perform at full rate there must be at least an 60% chance that any packet destination has already been seen in the recent past and that the entry is still in the cache. A study of a recent traffic trace taken from the Internet gave a 95% cache hit rate with a 6,000 entry cache [7]. However, this study was based on a packet trace from a 37 Mbps traffic stream. It is debatable whether the same amount of locality would be observed in traffic streams in excess of 1 Gbps. Also the traffic profile of the Internet changes over time so the debate continues.

The NetStar GigaRouter includes a forwarding engine on each line card with a 1 Gbps connection to the switch fabric. The forwarding engine is based on a SPARC microprocessor with hardware assisted route lookups so that no route caching is required. It supports a routing table of up to 150,000 routes. With a full routing table the route lookup takes 3 µs but the highest packet forwarding performance of currently available line cards is 136 kpps. This is well below the 500 kpps target rate needed for a 1 Gbps port.

The argument against a silicon based forwarding engine is that a hardware solution is fixed. Applications within the Internet, and thus the traffic profile, change over time and a fixed forwarding engine may not be able to track these changes. For example, multicast traffic may become much more important from multimedia applications and a move to IP version 6 may occur sooner than expected, both of which could invalidate a fixed implementation.

A forwarding engine designed to perform a high-speed destination address to outbound interface lookup is sufficient to offer a simple, best-effort packet forwarding service. But additional functionality will be required of the next generation of routers. This includes: multicast, quality of service differentiation, firewall filtering and more complex policy-based routing. To offer such functionality one needs to base the routing decision on more fields in the packet header than just the destination address.

IP Switching

IP switching is an alternative to the Gigabit Router. An IP switch maps the forwarding functions onto a hardware switch such as an ATM switch. A similar idea occurred independently, at about the same time, to three groups. The devices based upon this idea are called: IP/ATM [2], the IP switch [4], and the Cell Switch Router (CSR) [3,8]. Another mechanism for binding forwarding functions to an ATM VCI is also discussed in [9]. In addition, the Cisco Tag Switching proposal also appears to be similar to these earlier works [10].

Unlike some approaches, IP Switching may be used with any higher level IP functionality; it is not restricted to particular IP routing protocols or routing domains, and may be used, e.g., between an Internet Service Provider (ISP) and its customers or between ISPs.

Each approach uses the concept of a *flow*. A flow is defined as a sequence of packets that are treated identically by the possibly complex routing function. An example of a flow is a sequence of packets sent from a particular source to a particular destination (unicast or multicast) that are forwarded through particular ports with a particular QoS. The forwarding and handling of each flow is determined by the first packets in the flow. Once the flow is classified, these decisions may be cached and further packets on the flow may be processed according to the cache entry, without requiring the full flow classification.

Each of the above three solutions uses an ATM switch as the switch fabric for a high-speed router. Incoming flows are mapped onto ATM virtual channels (VCs) established across the ATM switch. Only one or a few packets from each flow need be inspected to perform the mapping and establish an ATM virtual channel. Once the virtual channel is established for a flow, all further traffic on that flow can be switched directly through the ATM switch, greatly reducing the load on the forwarding engine(s).

The IP/ATM solution uses a pool of pre-established permanent virtual channels (PVCs) that are taken by active incoming flows. Packets on a new flow are not forwarded until a PVC has been activated. The IP switch uses a protocol, IFMP (RFC1953), to propagate the mapping between flow and VCI upstream and forwards packets using the forwarding engine until the cut-through connection is established across the ATM switch. The Cell Switch Router attempts to be more general than the IP switch in that it will permit entire Classical IP over ATM (RFC1577) subnets between CSRs. It proposes using the RSVP protocol to propagate the mapping between flows and VCIs. To examine this class of device we will discuss the Ipsilon IP Switch in detail.

Flow Classification

An important function of the flow classification operation is to select those flows that are to be switched in the ATM switch and those that should be forwarded packet-by-packet in the forwarding engine. Clearly one wishes to select long duration flows with a lot of traffic for switching. Multimedia traffic: voice, image, video conferencing, etc., offers an example of long duration flows where there is a good probability of a high traffic volume. Many multimedia applications also require multicast which makes it very suitable for switching across an ATM switch making use of ATM’s hardware multicast capability. Short duration flows consisting of a small number of packets should be handled directly by the forwarding engine. Nameserver queries and brief client-server transactions are examples of traffic that are

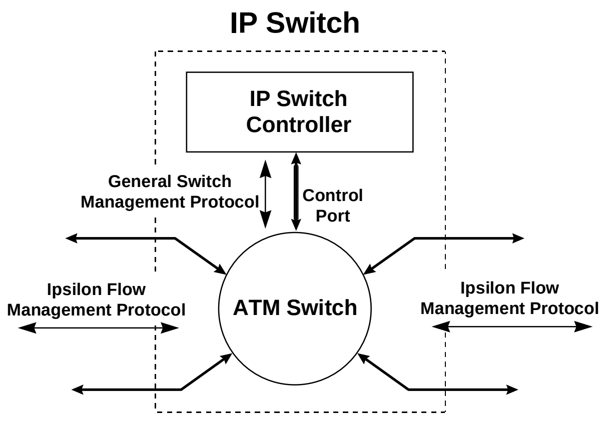
IP Switch
IP Switch Controller
General Switch Management Protocol
Control Port
Ipsilon Flow Management Protocol
ATM Switch
Ipsilon Flow Management Protocol

Figure 2: Structure of an IP switch

probably not worth the effort of establishing a switched connection.

For the flows selected for switching, a virtual channel must be established across the ATM switch. ATM switching requires that all arriving traffic be labelled with a virtual channel identifier (VCI) to indicate the virtual channel to which it belongs. So IP switching requires a protocol to distribute the association of flow and VCI label upstream across each incoming link.

Every packet on a flow that is switched through a network of IP switches must be labelled with a VCI. But the task of cache lookup and packet labelling is propagated upstream to the edge of the network. The task of labelling each packet typically involves more effort than simple forwarding because it must examine more fields than the destination address. However, once a virtual channel is established, this flow labelling need only be performed at a single location within the IP switch network; a traditional router network would need to perform the route lookup at every hop. Another advantage is that the rate of packet arrival is typically much lower at the edge of the network than in the center. Thus, for switched flows, per-packet work is offloaded from the forwarding engines in the center of the network at the cost of slightly increased per-flow work at the edge of the IP switch network. If the device on the edge of the network is a directly connected host, the classification and labelling operations can be trivially integrated into the host protocol software.

The set of virtual channels across the switch (or equivalently the VCI table on each link of the switch) may be regarded as a cache of flow forwarding decisions. In this sense it uses caching similar to the Multigigabit Router. In the Multigigabit Router the routing decision for every incoming packet is cached. However, in IP switches only selected flows are cached in the ATM hardware — the cache is explicitly managed.

The debate regarding whether caching is a good idea is equally applicable to IP switching, in fact, even more so since it takes more work to establish a switched flow. But since the forwarding of switched flows is implemented within the ATM switching hardware the forwarding engine of an IP switch only has to deal with the classification and forwarding of new flows and the forwarding of packets belonging to flows that are not switched. This allows some flexibility in the dimensioning of the forwarding engine depending upon the anticipated ratio of flows to be switched and packets to be forwarded and the traffic characteristics of the switched flows.

In summary, the IP switch provides high-speed routing by low-level switching of flows (equivalent to cached routing decisions). It defines a protocol to indicate these flows, and to associate a link layer label with each flow, to the upstream network node. This enables the switching. All flows are classified, and the forwarding engine is optimized for flow classification and for forwarding packets on those flows that are decided should not be cached in the switch fabric.

IP Switch Implementation

We now turn our attention to the Ipsilon IP Switch implementation and the two protocols required to support IP switching. The IP Switch is constructed from two components, an ATM switch and the IP Switch Controller, fig. 2. The IP Switch Controller is a high-end Pentium Pro machine running an operating system that continues to bear some resemblance to UNIX. One of the ports on the ATM switch is connected to an ATM interface on the IP Switch Controller and is used for both control and data transfer. The control protocol used between the switch and the controller is the General Switch Management Protocol, GSMP (RFC1987) [11], which has been designed to give the IP Switch Controller full control of an ATM switch. The Ipsilon Flow Management Protocol, IFMP (RFC1953) [12], is the flow-forwarding-cache distribution protocol. It runs between the IP Switch Controller and its peers across each external link. In comparison with fig. 1, the line cards are part of the ATM switch and the forwarding engine is implemented in software within the IP Switch Controller.

The IP switch is implemented in two components to allow a separation between hardware and software. Thus any ATM switch that supports RFC1987 may be used for the switching component. Different ATM switches are designed with different size, cost, and functionality tradeoffs so it makes sense to support a choice. This choice goes both ways. GSMP can also support a standard ATM Forum control protocol stack instead of the IP Switch Controller software. So a choice of network control software is possible for the same hardware.

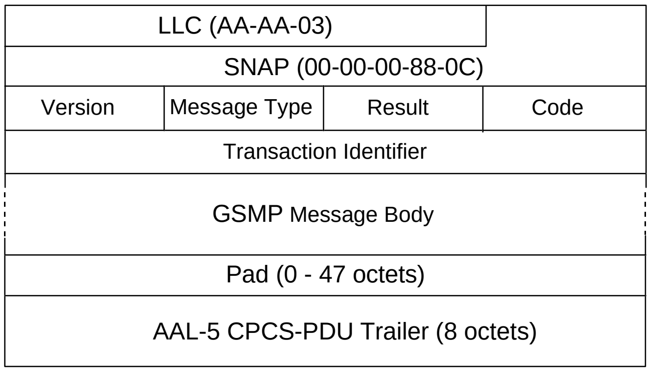

Figure 3: GSMP Message Format

General Switch Management Protocol (GSMP)

The design goal for the GSMP interface is to be as close to the actual switch hardware as possible and yet capable of controlling all (reasonable) ATM switch designs. These are conflicting requirements. GSMP is a simple master-slave, request-response protocol. The master (switch controller) sends requests and the switch issues a positive or negative response when the operation is complete. Virtual paths and virtual channels are assumed to be unidirectional (a requirement of RFC1953). Unreliable message transport is assumed between controller and switch for speed and simplicity. (The link between switch and controller will either be very reliable, or broken, in which case the overhead of adding error detection and retransmission through a protocol like SSCOP is unnecessary. All GSMP messages are acknowledged and the implementation handles its own retransmission.)

GSMP runs on a single, well-known virtual channel (VPI 0, VCI 15). All messages use an AAL-5 LLC/SNAP encapsulation but the most frequent messages (connection management) are designed to be small enough to be single cell AAL-5 packets, fig. 3. The LLC/SNAP encapsulation was chosen to allow other protocols beside GSMP to be multiplexed onto the link by using a different "Ethertype" in the SNAP header. For example, while GSMP offers some simple network management features, the Simple Network Management Protocol (SNMP) will be required between the controller and switch to offer full-service network management. (While SNMP can be used to establish connections in an ATM switch it was considered far too heavyweight a protocol to satisfy the design goals of GSMP.)

An adjacency protocol is used to synchronize state across the control link, to discover the identity of the entity at the far end of the link, and to detect when it changes. No GSMP messages other than the adjacency protocol may be sent across a link until adjacency has been established. Once established, five types of message may be sent: configuration, connection management, port management, statistics, and events.

The configuration messages are used by the controller to discover the capabilities of the ATM switch. Beyond name, rank, and serial number, each ATM switch port can report: the incoming VPI and VCI ranges it can support, its interface type and cell rate, its administrative and line status, and the number of priority levels it supports in its output queue. The current version of GSMP (RFC1987) assumes simple strict priority output queues of which any number of priority queues per port may be specified. (Queue structures other than output queueing may be mapped into this model.) The protocol will need to be extended to support the next generation of ATM queueing and scheduling hardware currently in development. Traffic policing (usage parameter control) is also not supported in this version. (It is unlikely to be required in IP switching until RSVP signalling is more widely deployed and it will be rendered unnecessary by implementations with per-VC queueing and scheduling.)

Once the configuration of the switch has been discovered, the controller can begin issuing connection management messages. These are the most common messages. They enable the controller to establish and remove connections across the switch. No distinction is made between unicast and multicast connections — the "Add Branch" and "Delete Branch" messages are used for both. The first Add Branch message on a new incoming VCI defines a new unicast connection. The second Add Branch message on an existing incoming VCI converts the connection to a point-to-multipoint connection with two branches, etc. This was intentional as no distinction is made in IFMP. However, in hindsight, it would be better to give a hint if a multicast connection is being established as many switches use completely different data structures to implement unicast and multicast connections. A Delete Tree message is available to delete an entire multicast connection. A Move Branch message allows a single output branch of a multicast connection to be moved from one output port and VCI to another. The Move Branch message is used in the cut-through operation where an IP flow is moved from connectionless forwarding to direct switching.

Of the remaining GSMP messages, the Port Management message is used to reset, bring up, take down and loopback switch ports. The statistics message permits various per-VC and per-port performance counts to be requested. The event messages allow a switch to asynchronously alert the controller to significant events such as: loss (or detection) of carrier on a port, loss (or detection) of port interfaces (so that hot-swap hardware may be supported), and arrival of cells with invalid

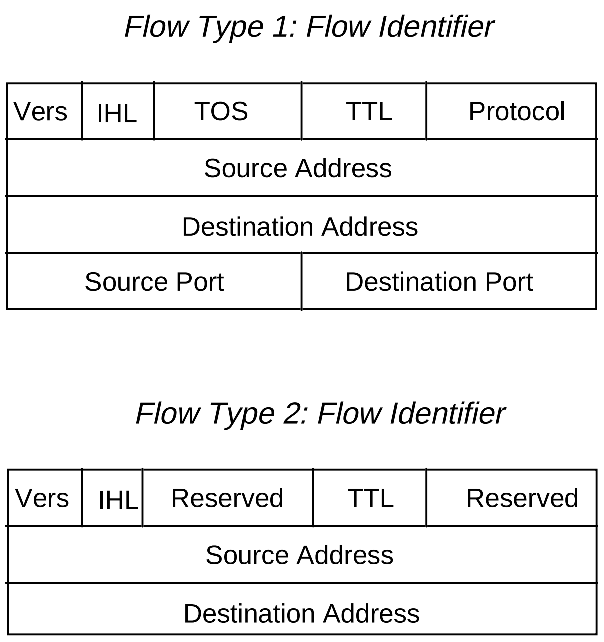

Figure 4: IFMP Flow Identifiers

VPI/VCIs. A simple flow control protocol is applied to the event messages to prevent the controller being flooded.

At the time of writing, RFC1987 has been implemented on at least eight different ATM switches. The code size for the GSMP slave is about 2,000 lines. A reference implementation is available and it typically takes one or two weeks to get GSMP up and running on a new switch design. The measured performance of the GSMP slave on Ipsilon's IP switch is currently just under 1000 connection setups per second. This could be improved considerably if an ATM segmentation and reassembly (SAR) device were added to the switch to offload many of the packet handling and AAL processing functions currently performed in software by the embedded processor on the ATM switch.

Ipsilon Flow Management Protocol (IFMP)

IFMP runs independently across each link in a network of IP switches that connects IFMP peers — IP switches, directly attached hosts, or IFMP capable edge routers. On ATM links it uses the default virtual channel (VPI 0, VCI 15) [13]. The purpose of IFMP is to inform the transmitting end of a link of the VCI that should be associated with a particular IP flow. The VCI is selected by the receiving end of the link.

All packets belonging to flows that have not yet been switched are forwarded hop-by-hop between IP Switch Controllers using the default virtual channel on each link. When a new flow arrives at an IP switch it is classified. One of the results of flow classification is a decision as to

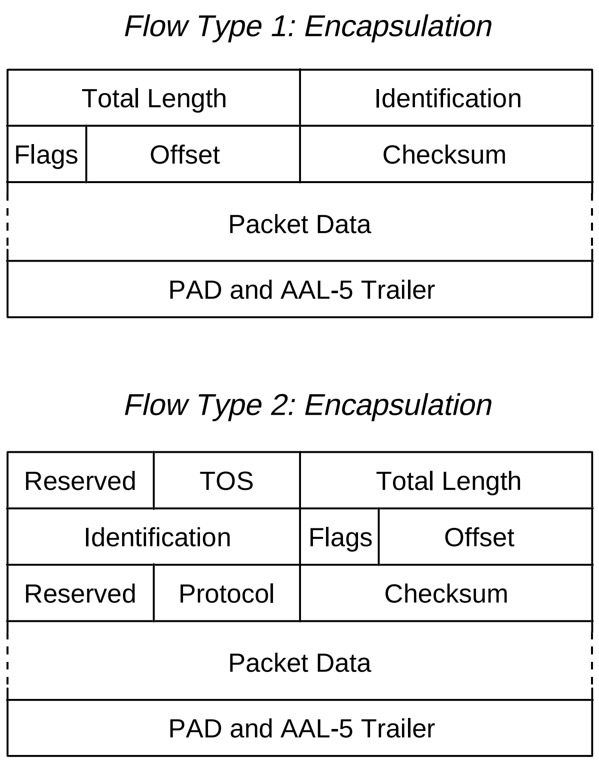

Figure 5: IFMP Packet Encapsulation

if or when the flow should be switched and the granularity, or flow type, at which it should be switched. Currently we have defined two flow types: a host-pair flow type (flow type 2) and a port-pair flow type (flow type 1). The host-pair flow type is for traffic flowing between the same source and destination IP addresses. 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 port-pair flow type allows quality of service differentiation among flows between the same pair of hosts and also supports simple flow-based firewall security features.

Before a flow can be switched it must first be labelled. A free VCI is first selected by the receiver on the incoming link. An IFMP redirect message is then sent upstream to inform the transmitter at the other end of the link of the association between flow and VCI. The flow is identified by a flow identifier, fig. 4. The flow identifier gives the values of the set of fields from the packet header that a packet must match to belong to this flow. The redirect message also contains a lifetime field that specifies the length of time for which this association of flow and VCI is valid. The flow redirection must be refreshed by another IFMP redirect message before the lifetime expires else the association of flow and VCI is deleted.

This flow labelling process occurs independently and concurrently on each link in an IP switching network. However, we may assume that the flow classification policy is consistent within an administrative domain. So if one node decides to label a flow, its neighbors within the same domain will very likely make the same flow classification and switching decision. When an IP switch controller sends an IFMP redirect message it checks to see if the flow is labelled yet on the downstream link. Also, when it receives a redirect request it checks to see if the flow is yet labelled on the upstream link. When upstream and downstream links are both labelled for a given flow, that flow is switched directly through the ATM switch.

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 channel does not use an LLC/SNAP header and removes all of the IP header fields specified by the flow identifier from the header of each packet, fig. 5. The IP packet with the resulting compressed header is then encapsulated in AAL-5 and transmitted on the specified virtual channel. The IP switch issuing the redirection keeps a copy of the removed fields and associates them with the specified ATM VCI. The switch my reconstruct the complete header using the stored 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.

The Time to Live (TTL) field from the IP packet header is included in the flow identifier for both flow types 1 and 2. This ensures that only packets with the correct TTL may be included in a switched flow. Thus at the end of a switched flow, the TTL of packets on that flow must be correct as the TTL field is not transmitted in the packet but is recovered from information stored at the destination. 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. The value of the TTL field is added to the header checksum at the end of a switched flow 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 over time, if, for example, more upstream IP switches decide to switch a particular flow.

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. The upstream IP switch controller continues to send packets on the redirected VCI until a timeout interval expires during which it has not received any redirects. Once the flow has timed out, the upstream controller removes the associated state. The same is true for the controller that issued the redirect.

Alternatively, the downstream IP Switch controller may reclaim the VCI by issuing an IFMP Reclaim message. The downstream flow state is deleted after the IFMP Reclaim Ack message is received. Normally the flows are allowed to timeout, but in some cases they need to be explicitly deleted. Some examples are routing changes and lack of receive VCI resources.

Conclusion

The IP switch is an alternative architecture to the gigabit router for providing high speed routing. It uses low-level switching of flows (equivalent to cached routing decisions) and includes a cooperative protocol to allow explicit use and management of this cached information, on a link-by-link basis, throughout an IP switching network. We have presented an overview of the protocols developed to support Ipsilon's IP switch implementation. In an IP switch, all flows are classified. The flow classification process dynamically selects flows to be forwarded in the switch fabric while the remaining flows are forwarded hop-by-hop. This flow classification allows an IP switch to intrinsically support multicast, quality of service differentiation, simple firewall filtering, and complex policy-based routing decisions for each switched flow. These features can be difficult to support in a gigabit router with the fast forwarding path optimized for only destination address lookup. The forwarding engine in an IP switch is optimized for flow classification and for forwarding packets on those flows that are decided should not be cut-through the switch fabric.

Acknowledgment

We would like to thank Bob Hinden, Fong-Ching Liaw, and Eric Hoffman, for their contribution to the work presented here and also Dennis Ferguson for his helpful comments on reviewing this paper.

References

[1] C. Partridge, "A fifty gigabit per second IP router," Paper in preparation.

[2] G. Parulkar, D. C. Schmidt, J. S. Turner, "IP/ATM: A strategy for integrating IP with ATM," In SIGCOMM

Symp. on Commun. Architectures and Protocols, Cambridge MA, Sep. 1995, page 9.

[3] H. Esaki, and K.-I. Nagami, M. Ohta, “High speed datagram delivery over Internet using ATM technology,” Networld+Interop, Las Vegas, Mar. 1995, E12-1.

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

[5] D. Kachelmeyer, “A new router architecture for tomorrow’s Internet,” NetStar, Inc. http://www.netstar.com.

[6] T-B. Pei, C. Zukowski, “Putting routing tables in silicon,” IEEE Network Mag., Jan. 1992, 42–50.

[7] C. Partridge, “Locality and route caches,” NSF Workshop on Internet Statistics Measurement and Analysis, San Diego CA, Feb. 1996.

[8] Y. Katsube, K.-I. Nagami, and H. Esaki, “Router architecture extensions for ATM: Overview,” IETF Internet Draft, draft-katsube-router-atm-overview-02.txt, Mar. 1996.

[9] Y. Goto, “Session Identity Notification Protocol (SINP),” IETF Internet Draft, draft-goto-sinp-02.txt, Jan. 1996.

[10] Y. Rekhter et al., “Tag switching architecture overview,” IETF Internet Draft, draft-rfced-info-rekhter-00.txt, Sep. 1996.

[11] P. Newman et al., “Ipsilon's General Switch Management Protocol Specification Version 1.1,” IETF RFC 1987, Aug. 1996.

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

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

[14] M. Lottor, Network Wizards, http://www.nw.com.

[15] M. Gray, Massachusetts Institute of Technology, http://www.mit.edu/people/mkgray.

[16] R. Pearlman, "Interconnections: bridges and routers," Addison-Wesley, 1992.