Internet Draft
Expiration date:January 1998

Yakov Rekhter
cisco Systems
Bruce Davie
cisco Systems
Dave Katz
Juniper Networks Inc.
Eric Rosen
cisco Systems
George Swallow
cisco Systems
Dino Farinacci
cisco Systems
July 1997

Tag Switching Architecture - Overview

draft-rekhter-tagswitch-arch-01.txt

1. Status of this Memo

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress."

Please check the 1id-abstracts.txt listing contained in the internet-drafts Shadow Directories on nic.ddn.mil, nnsc.nsf.net, nic.nordu.net, ftp.nisc.sri.com, or munnari.oz.au to learn the current status of any Internet Draft.

2. Abstract

This document provides an overview of tag switching. Tag switching is a way to combine the label-swapping forwarding paradigm with network layer routing. This has several advantages. Tags can have a wide spectrum of forwarding granularities, so at one end of the spectrum a tag could be associated with a group of destinations, while at the other a tag could be associated with a single application flow. At the same time forwarding based on tag switching, due to its simplicity, is well suited to high performance forwarding. These factors facilitate the development of a routing system which is both functionally rich and scalable. Finally, tag switching simplifies integration of routers and ATM switches by employing common addressing, routing, and management procedures.

3. Introduction

Continuous growth of the Internet demands higher bandwidth within the Internet Service Providers (ISPs). However, growth of the Internet is not the only driving factor for higher bandwidth - demand for higher bandwidth also comes from emerging multimedia applications. Demand for higher bandwidth, in turn, requires higher forwarding performance for both multicast and unicast traffic.

The growth of the Internet also demands improved scaling properties of the Internet routing system. The ability to contain the volume of routing information maintained by individual routers and the ability to build a hierarchy of routing knowledge are essential to support a high quality, scalable routing system.

While the destination-based forwarding paradigm is adequate in many situations, we already see examples where it is no longer adequate. The ability to overcome the rigidity of destination-based forwarding and to have more flexible control over how traffic is routed is likely to become more and more important.

We see the need to improve forwarding performance while at the same time adding routing functionality to support multicast, allowing more flexible control over how traffic is routed, and providing the ability to build a hierarchy of routing knowledge. Moreover, it becomes more and more crucial to have a routing system that can

support graceful evolution to accommodate new and emerging requirements.

Tag switching is a technology that provides an efficient solution to these challenges. Tag switching blends the flexibility and rich

functionality provided by Network Layer routing with the simplicity provided by the label swapping forwarding paradigm. The simplicity of the tag switching forwarding paradigm (label swapping) enables improved forwarding performance, while maintaining competitive price/performance. By associating a wide range of forwarding granularities with a tag, the same forwarding paradigm can be used to support a wide variety of routing functions, such as destination-based routing, multicast, hierarchy of routing knowledge, and flexible routing control. Finally, a combination of simple forwarding, a wide range of forwarding granularities, and the ability to evolve routing functionality while preserving the same forwarding paradigm enables a routing system that can gracefully evolve to accommodate new and emerging requirements.

4. Tag Switching components

Tag switching consists of two components: forwarding and control. The forwarding component uses the tag information (tags) carried by packets and the tag forwarding information maintained by a tag switch to perform packet forwarding. The control component is responsible for maintaining correct tag forwarding information among a group of inter- connected tag switches.

Segregating control and forwarding into separate components promotes modularity, which in turn enables to build a system that can gracefully evolve to accommodate new and emerging requirements.

5. Forwarding component

The fundamental forwarding paradigm employed by tag switching is based on the notion of label swapping. When a packet with a tag is received by a tag switch, the switch uses the tag as an index in its Tag Information Base (TIB). Each entry in the TIB consists of an

incoming tag, and one or more sub-entries of the form <outgoing tag, outgoing interface, outgoing link level information>. If the switch finds an entry with the incoming tag equal to the tag carried in the packet, then for each <outgoing tag, outgoing interface, outgoing link level information> in the entry the switch replaces the tag in the packet with the outgoing tag, replaces the link level information (e.g MAC address) in the packet with the outgoing link level information, and forwards the packet over the outgoing interface.

From the above description of the forwarding component we can make several observations. First, the forwarding decision is based on the exact match algorithm using a fixed length, fairly short tag as an

---

index. This enables a simplified forwarding procedure, relative to longest match forwarding traditionally used at the network layer. This in turn enables higher forwarding performance (higher packets per second). The forwarding procedure is simple enough to allow a straightforward hardware implementation.

A second observation is that the forwarding decision is independent of the tag's forwarding granularity. For example, the same forwarding algorithm applies to both unicast and multicast - a unicast entry would just have a single (outgoing tag, outgoing interface, outgoing link level information) sub-entry, while a multicast entry may have one or more (outgoing tag, outgoing interface, outgoing link level information) sub-entries. (For multi-access links, the outgoing link level information in this case would include a multicast MAC address.) This illustrates how with tag switching the same forwarding paradigm can be used to support different routing functions (e.g., unicast, multicast, etc...)

The simple forwarding procedure is thus essentially decoupled from the control component of tag switching. New routing (control) functions can readily be deployed without disturbing the forwarding paradigm. This means that it is not necessary to re-optimize forwarding performance (by modifying either hardware or software) as new routing functionality is added.

In the tag switching architecture, various implementation options are acceptable. For example, support for network layer forwarding by a tag switch (i.e., forwarding based on the network layer header as opposed to a tag) is optional. Moreover, use of network layer

forwarding may be constrained to handling network layer control traffic only. (Note, however, that a tag switch must be able to source and sink network layer packets, e.g. to participate in network layer routing protocols)

For the purpose of handling network layer hop count (time-to-live) the architecture allows two alternatives: network layer hops may correspond directly to hops formed by tag switches, or one network layer hop may correspond to several tag switched hops.

When a switch receives a packet with a tag, and the TIB maintained by the switch has no entry with the incoming tag equal to the tag carried by the packet, or the entry exists, the outgoing tag entry is entry, and the entry does not indicate local delivery to the switch, the switch may either (a) discard the packet, or (b) strip the tag information, and submit the packet for network layer processing. Support for the latter is optional (as support for network layer forwarding is optional). Note that it may not always be possible to successfully forward a packet after stripping a tag even if a tag

switch supports network layer forwarding.

The architecture allows a tag switch to maintain either a single TIB per tag switch, or a TIB per interface. Moreover, a tag switch could mix both of these options - some tags could be maintained in a single TIB, while other tags could be maintained in a TIB associated with individual interfaces.

5.1. Tag encapsulation

Tag switching clearly requires a tag to be carried in each packet. The tag information can be carried in a variety of ways:

- as a small "shim" tag header inserted between the layer 2 and the Network Layer headers;

- as part of the layer 2 header, if the layer 2 header provides adequate semantics (e.g., Frame Relay, or ATM);

- as part of the Network Layer header (e.g., using the Flow Label

field in IPv6 with appropriately modified semantics).

It is therefore possible to implement tag switching over virtually any media type including point-to-point links, multi-access links, and ATM. At the same time the forwarding component allows specific optimizations for particular media (e.g., ATM).

Observe also that the tag forwarding component is Network Layer independent. Use of control component(s) specific to a particular Network Layer protocol enables the use of tag switching with different Network Layer protocols.

6. Control component

Essential to tag switching is the notion of binding between a tag and Network Layer routing (routes). The control component is responsible for creating tag bindings, and then distributing the tag binding information among tag switches. Creating a tag binding involves allocating a tag, and then binding a tag to a route. The distribution of tag binding information among tag switches could be accomplished via several options:

- piggybacking on existing routing protocols

- using a separate Tag Distribution Protocol (TDP)

While the architecture supports distribution of tag binding information that is independent of the underlying routing protocols, the architecture acknowledges that considerable optimizations can be achieved in some cases by small enhancements of existing protocols to enable piggybacking tag binding information on these protocols.

One important characteristic of the tag switching architecture is that creation of tag bindings is driven primarily by control traffic rather than by data traffic. Control traffic driven creation of tag bindings has several advantages, as compared to data traffic driven creation of tag bindings. For one thing, it minimizes the amount of additional control traffic needed to distribute tag binding information, as tag binding information is distributed only in

response to control traffic, independent of data traffic. It also makes the overall scheme independent of and insensitive to the data traffic profile/pattern. Control traffic driven creation of tag binding improves forwarding performance, as tags are precomputed (prebound) before data traffic arrives, rather than being created as data traffic arrives. It also simplifies the overall system behavior, as the control plane is controlled solely by control traffic, rather than by a mix of control and data traffic.

Another important characteristic of the tag switching architecture is that distribution and maintenance of tag binding information is consistent with distribution and maintenance of the associated routing information. For example, distribution of tag binding information for tags associated with unicast routing is based on the technique of incremental updates with explicit acknowledgment. This is very similar to the way unicast routing information gets distributed by such protocols as OSPF and BGP. In contrast, distribution of tag binding information for tags associated with multicast routing is based on period updates/ refreshes, without any explicit acknowledgments. This is consistent with the way multicast routing information is distributed by such protocols as PIM.

To provide good scaling characteristics, while also accommodating diverse routing functionality, tag switching supports a wide range of forwarding granularities. At one extreme a tag could be associated (bound) to a group of routes (more specifically to the Network Layer Reachability Information of the routes in the group). At the other extreme a tag could be bound to an individual application flow (e.g., an RSVP flow). A tag could also be bound to a multicast tree. In addition, a tag may be bound to a path that has been selected for a certain set of packets based on some policy (e.g. an explicit route).

The control component is organized as a collection of modules, each

---

designed to support a particular routing function. To support new routing functions, new modules can be added. The architecture does not mandate a prescribed set of modules that have to be supported by every tag switch.

The following describes some of the modules.

6.1. Destination-based routing

In this section we describe how tag switching can support destination-based routing. Recall that with destination-based routing a router makes a forwarding decision based on the destination address carried in a packet and the information stored in the Forwarding Information Base (FIB) maintained by the router. A router constructs its FIB by using the information it receives from routing protocols (e.g., OSPF, BGP).

To support destination-based routing with tag switching, a tag switch, just like a router, participates in routing protocols (e.g., OSPF, BGP), and constructs its FIB using the information it receives from these protocols.

There are three permitted methods for tag allocation and Tag Information Base (TIB) management: (a) downstream tag allocation, (b) downstream tag allocation on demand, and (c) upstream tag allocation. In all cases, a switch allocates tags and binds them to address prefixes in its FIB. In downstream allocation, the tag that is carried in a packet is generated and bound to a prefix by the switch at the downstream end of the link (with respect to the direction of data flow). On demand allocation means that tags will only be allocated and distributed by the downstream switch when it is requested to do so by the upstream switch. Method (b) is most useful in ATM networks (see Section 8). In upstream allocation, tags are allocated and bound at the upstream end of the link. Note that in downstream allocation, a switch is responsible for creating tag bindings that apply to incoming data packets, and receives tag bindings for outgoing packets from its neighbors. In upstream allocation, a switch is responsible for creating tag bindings for outgoing tags, i.e. tags that are applied to data packets leaving the switch, and receives bindings for incoming tags from its neighbors.

The downstream tag allocation scheme operates as follows: for each route in its FIB the switch allocates a tag, creates an entry in its Tag Information Base (TIB) with the incoming tag set to the allocated tag, and then advertises the binding between the (incoming) tag and the route to other adjacent tag switches. The advertisement could be accomplished by either piggybacking the binding on top of the

existing routing protocols, or by using a separate Tag Distribution

Protocol (TDP). When a tag switch receives tag binding information for a route, and that information was originated by the next hop for that route, the switch places the tag (carried as part of the binding information) into the outgoing tag of the TIB entry associated with the route. This creates the binding between the outgoing tag and the route.

With the downstream on demand tag allocation scheme, operation is as follows. For each route in its FIB, the switch identifies the next hop for that route. It then issues a request (via TDP) to the next hop for a tag binding for that route. When the next hop receives the request, it allocates a tag, creates an entry in its TIB with the incoming tag set to the allocated tag, and then returns the binding between the (incoming) tag and the route to the switch that sent the original request. When the switch receives the binding information, the switch creates an entry in its TIB, and sets the outgoing tag in the entry to the value received from the next hop. Handling of data packets is as for downstream allocation. The main application for this mode of operation is with ATM switches, as described in Section 8.

The upstream tag allocation scheme is used as follows. If a tag switch has one or more point-to-point interfaces, then for each route in its FIB whose next hop is reachable via one of these interfaces, the switch allocates a tag, creates an entry in its TIB with the outgoing tag set to the allocated tag, and then advertises to the next hop (via TDP) the binding between the (outgoing) tag and the route. When a tag switch that is the next hop receives the tag binding information, the switch places the tag (carried as part of the binding information) into the incoming tag of the TIB entry associated with the route.

Note that, while we have described upstream allocation for the sake of completeness, we have found the two downstream allocation methods adequate for all practical purposes so far.

Independent of which tag allocation method is used, once a TIB entry is populated with both incoming and outgoing tags, the tag switch can forward packets for routes bound to the tags by using the tag switching forwarding algorithm (as described in Section 5).

When a tag switch creates a binding between an outgoing tag and a route, the switch, in addition to populating its TIB, also updates its FIB with the binding information. This enables the switch to add tags to previously untagged packets.

So far we have described how a tag could be bound to a single route,

creating a one-to-one mapping between routes and tags. However, under certain conditions it is possible to bind a tag not just to a single route, but to a group of routes, creating a many-to-one mapping between routes and tags. Consider a tag switch that is connected to a router. It is quite possible that the switch uses the router as the next hop not just for one route, but for a group of routes. Under these conditions the switch does not have to allocate distinct tags to each of these routes - one tag would suffice. The distribution of tag binding information is unaffected by whether there is a one-to-one or one-to-many mapping between tags and routes. Now consider a tag switch that receives from one of its neighbors (tag switching peers) tag binding information for a set of routes, such that the set is bound to a single tag. If the switch decides to use some or all of the routes in the set, then for these routes the switch does not need to allocate individual tags - one tag would suffice. Such an approach may be valuable when tags are a precious resource. Note that the ability to support many-to-one mapping makes no assumptions about the routing protocols being used.

When a tag switch adds a tag to a previously untagged packet the tag could be either associated with the route to the destination address carried in the packet, or with the route to some other tag switch along the path to the destination (in some cases the address of that other tag switch could be gleaned from network layer routing protocols). The latter option provides yet another way of mapping multiple routes into a single tag. However, this option is either dependent on particular routing protocols, or would require a separate mechanism for discovering tag switches along a path.

To understand the scaling properties of tag switching in conjunction with destination-based routing, observe that the total number of tags that a tag switch has to maintain can not be greater than the number of routes in the switch's FIB. Moreover, as we have just seen, the number of tags can be much less than the number of routes. Thus, much less state is required than would be the case if tags were allocated to individual flows.

In general, a tag switch will try to populate its TIB with incoming and outgoing tags for all routes to which it has reachability, so that all packets can be forwarded by simple label swapping. Tag allocation is thus driven by topology (routing), not data traffic - it is the existence of a FIB entry that causes tag allocations, not the arrival of data packets.

Use of tags associated with routes, rather than flows, also means that there is no need to perform flow classification procedures for

all the flows to determine whether to assign a tag to a flow. That, in turn, simplifies the overall scheme, and makes it more robust and

stable in the presence of changing traffic patterns.

Note that when tag switching is used to support destination-based routing, tag switching does not completely eliminate the need to perform normal Network Layer forwarding at some network elements. First of all, to add a tag to a previously untagged packet requires normal Network Layer forwarding. This function could be performed by the first hop router, or by the first router on the path that is able to participate in tag switching. In addition, whenever a tag switch aggregates a set of routes (e.g., by using the technique of hierarchical routing), into a single route, and the routes do not share a common next hop, the switch needs to perform Network Layer forwarding for packets carrying the tag associated with the aggregated route. However, one could observe that the number of places where routes get aggregated is smaller than the total number of places where forwarding decisions have to be made. Moreover, quite often aggregation is applied to only a subset of the routes maintained by a tag switch. As a result, on average a packet can be forwarded most of the time using the tag switching algorithm. Note that many tag switches may not need to perform any network layer forwarding.

6.2. Hierarchy of routing knowledge

The IP routing architecture models a network as a collection of routing domains. Within a domain, routing is provided via interior routing (e.g., OSPF), while routing across domains is provided via exterior routing (e.g., BGP). However, all routers within domains that carry transit traffic (e.g., domains formed by Internet Service Providers) have to maintain information provided by not just interior routing, but exterior routing as well, even if only some of these routers participate in exterior routing. That creates certain problems. First of all, the amount of this information is not insignificant. Thus it places additional demand on the resources required by the routers. Moreover, increase in the volume of routing information quite often increases routing convergence time. This, in turn, degrades the overall performance of the system.

Tag switching allows complete decoupling of interior and exterior routing. With tag switching only tag switches at the border of a domain would be required to maintain routing information provided by exterior routing - all other switches within the domain would just maintain routing information provided by the domains interior routing (which is usually significantly smaller than the exterior routing information), with no "leaking" of exterior routing information into interior routing. This, in turn, reduces the routing load on non-border switches, and shortens routing convergence time.

To support this functionality, tag switching allows a packet to carry not one but a set of tags, organized as a stack. A tag switch could either swap the tag at the top of the stack, or pop the stack, or swap the tag and push one or more tags into the stack.

Consider a tag switch that is at the border of a routing domain. This switch maintains both exterior and interior routes. The interior routes provide routing information and tags to all the other tag switches within the domain. For each exterior route that the switch receives from some other border tag switch that is in the same domain as the local switch, the switch maintains not just a tag associated with the route, but also a tag associated with the route to that other border tag switch. Moreover, for inter-domain routing protocols that are capable of passing the "third-party" next hop information the switch would maintain a tag associated with the route to the next hop, rather than with the route to the border tag switch from whom the local switch received the exterior route.

When a packet is forwarded between two (border) tag switches in different domains, the tag stack in the packet contains just one tag (associated with an exterior route). However, when a packet is forwarded within a domain, the tag stack in the packet contains not one, but two tags (the second tag is pushed by the domain's ingress border tag switch). The tag at the top of the stack provides packet forwarding to an appropriate egress border tag switch (or the "third-party" next hop), while the next tag in the stack provides correct packet forwarding at the egress switch (or at the "third-party" next hop). The stack is popped by either the egress switch (or the "third-party" next hop) or by the penultimate (with respect to the egress switch/"third-party" next hop) switch.

One could observe that when tag switching is confined to a single

routing domain, the above still could be used to decouple interior from exterior routing, similar to what was described above. However, in this case a border tag switch wouldn't maintain tags associated with each exterior route, and forwarding between domains would be performed at the network layer.

The control component used in this scenario is fairly similar to the one used with destination-based routing. In fact, the only essential difference is that in this scenario the tag binding information is distributed both among physically adjacent tag switches, and among border tag switches within a single domain. One could also observe that the latter (distribution among border switches) could be trivially accommodated by very minor extensions to BGP.

The notion of supporting hierarchy of routing knowledge with tag switching is not limited to the case of exterior/interior routing,

but could be applicable to other cases where the hierarchy of routing knowledge is possible. Moreover, while the above describes only a two-level hierarchy of routing knowledge, the tag switching architecture does not impose limits on the depth of the hierarchy.

In the presence of hierarchy of routing knowledge a tag switched path at the level N in the hierarchy has to have its endpoints at tag switches that are at border between the level N and (N-1) in the hierarchy (level 0 in the hierarchy corresponds to an untagged path).

6.3. Multicast

Essential to multicast routing is the notion of spanning trees. Multicast routing procedures (e.g., PIM) are responsible for constructing such trees (with receivers as leafs), while multicast forwarding is responsible for forwarding multicast packets along such trees. Thus, to support a multicast forwarding function with tag switching we need to be able to associate a tag with a multicast tree. The following describes the procedures for allocation and distribution of tags for multicast.

When tag switching is used for multicast, it is important that tag switching be able to utilize multicast capabilities provided by the Data Link layer (e.g., multicast capabilities provided by Ethernet).

To be able to do this, an (upstream) tag switch connected to a given Data Link subnetwork should use the same tag when forwarding a multicast packet to all of the (downstream) switches on that subnetwork. This way the packet will be multicasted at the Data Link layer over the subnetwork. To support this, all tag switches that are part of a given multicast tree and are on a common subnetwork must agree on a common tag that would be used for forwarding multicast packets along the tree over the subnetwork. Moreover, since multicast forwarding is based on Reverse Path Forwarding (RPF), it is crucial that, when a tag switch receives a multicast packet, a tag carried in a packet must enable the switch to identify both (a) a particular multicast group, as well as (b) the previous hop (upstream) tag switch that sent the packet.

To support the requirements outlined in the previous paragraph, the tag switching architecture assumes that (a) multicast tags are associated with interfaces on a tag switch (rather than with a tag switch as a whole), (b) the tag space that a tag switch could use for allocating tags for multicast is partitioned into non-overlapping regions among all the tag switches connected to a common Data Link subnetwork, and (c) there are procedures by which tag switches that belong to a common multicast tree and are on a common Data Link subnetwork agree on the tag switch that is responsible for allocating

---

a tag for the tree.

One possible way of partitioning tag space into non-overlapping regions among tag switches connected to a common subnetwork is for each tag switch to claim a region of the space and announce this region to its neighbors. Conflicts are resolved based on the IP address of the contending switches (the higher address wins, the lower retries). Once the tag space is partitioned among tag switches, the switches may create bindings between tags and multicast trees (routes).

At least in principle there are two possible ways to create bindings between tags and multicast trees (routes). With the first alternative for a set of tag switches that share a common Data Link subnetwork, the tag switch that is upstream with respect to a particular multicast tree allocates a tag (out of its own region that does not overlap with the regions of other switches on the subnetwork), binds the tag to a multicast route, and then advertises the binding to all

the (downstream) switches on the subnetwork. With the second alternative, one of the tag switches that is downstream with respect to a particular multicast tree allocates a tag (out of its own region that does not overlap with the regions of other switches on the subnetwork), binds the tag to a multicast route, and then advertises the binding to all the switches (both downstream and upstream) on the subnetwork. Usually the first tag switch to join the group is the one that performs the allocation.

Each of the above alternatives has its own trade-offs. The first alternative is fairly simple - one upstream router does the tag binding and multicasts the binding downstream. However, the first alternative may create uneven distribution of allocated tags, as some tag switches on a common subnetwork may have more upstream multicast sources than the others. Also, changes in topology could result in upstream neighbor changes, which in turn would require tag re-binding. Finally, one could observe that distributing tag binding from upstream towards downstream is inconsistent with the direction of multicast routing information distribution (from downstream towards upstream).

The second alternative, even if more complex that the first one, has its own advantages. For one thing, it makes distribution of multicast tag binding consistent with the distribution of unicast tag binding. It also makes distribution of multicast tag binding consistent with the distribution of multicast routing information. This, in turn, allows the piggybacking of tag binding information on existing multicast routing protocols (PIM). This alternative also avoids the need for tag re-binding when there are changes in upstream neighbor. Finally it is more likely to provide more even distribution of

allocated tags, as compared to the first alternative. Note that this approach does require a mechanism to choose the tag allocator from among the downstream tag switches on the subnetwork.

6.4. Quality of service

Two mechanisms are needed for providing a range of qualities of service to packets passing through a router or a tag switch. First, we need to classify packets into different classes. Second, we need to ensure that the handling of packets is such that the appropriate

QOS characteristics (bandwidth, loss, etc.) are provided to each class.

Tag switching provides an easy way to mark packets as belonging to a particular class after they have been classified the first time. Initial classification could be done using configuration information (e.g., all traffic from a certain interface) or using information carried in the network layer or higher layer headers (e.g., all packets between a certain pair of hosts). A tag corresponding to the resultant class would then be applied to the packet. Tagged packets can then be efficiently handled by the tag switching routers in their path without needing to be reclassified. The actual scheduling and queueing of packets is largely orthogonal - the key point here is that tag switching enables simple logic to be used to find the state that identifies how the packet should be scheduled.

Tag switching can, for example, be used to support a small number of classes of service in a service provider network (e.g. premium and standard). On frame-based media, the class can be encoded by a field in the tag header. On ATM tag switches, additional tags can be allocated to differentiate the different classes. For example, rather than having one tag for each destination prefix in the FIB, an ATM tag switch could have two tags per prefix, one to be used by premium traffic and one by standard. Thus a tag binding in this case is a triple consisting of <prefix, QOS class, tag>. Such a tag would be used both to make a forwarding decision and to make a scheduling decision, e.g., by selecting the appropriate queue in a weighted fair queueing (WFQ) scheduler.

To provide a finer granularity of QOS, tag switching can be used with RSVP. We propose a simple extension to RSVP in which a tag object is defined. Such an object can be carried in an RSVP reservation message and thus associated with a session. Each tag capable router assigns a tag to the session and passes it upstream with the reservation message. Thus the association of tags with RSVP sessions works very much like the binding of tags to routes with downstream allocation. Note, however, that binding is accomplished using RSVP rather than

---

TDP. (It would be possible to use TDP, but it is simpler to extend RSVP to carry tags and this ensures that tags and reservation information are communicated in a similar manner.)

When data packets are transmitted, the first router in the path that is tag-capable applies the tag that it received from its downstream neighbor. This tag can be used at the next hop to find the corresponding reservation state, to forward and schedule the packet appropriately, and to find the suitable outgoing tag value provided by the next hop. Note that tag imposition could also be performed at the sending host.

6.5. Flexible routing (explicit routes)

One of the fundamental properties of destination-based routing is that the only information from a packet that is used to forward the packet is the destination address. While this property enables highly scalable routing, it also limits the ability to influence the actual paths taken by packets. This, in turn, limits the ability to evenly distribute traffic among multiple links, taking the load off highly utilized links, and shifting it towards less utilized links. For Internet Service Providers (ISPs) who support different classes of service, destination-based routing also limits their ability to segregate different classes with respect to the links used by these classes. Some of the ISPs today use Frame Relay or ATM to overcome the limitations imposed by destination-based routing. Tag switching, because of the flexible granularity of tags, is able to overcome these limitations without using either Frame Relay or ATM.

Another application where destination-based routing is no longer adequate is routing with resource reservations (QOS routing). Increasing the number of ways by which a particular reservation could traverse a network may improve the success of the reservation. Increasing the number of ways, in turn, requires the ability to explore paths that are not constrained to the ones constructed solely based on destination.

To provide forwarding along paths that are different from the paths determined by destination-based routing, the control component of tag switching allows installation of tag bindings in tag switches that do not correspond to the destination-based routing paths.

One possible alternative for supporting explicit routes is to allow TDP to carry information about an explicit route, where such a route could be expressed as a sequence of tag switches. Another alternative is to use tag-capable RSVP (see Section 6.4) as a mechanism to distribute tag bindings, and to augment RSVP with the ability to

steer the PATH message along a particular (explicit) route. Finally, it is also possible in principle to use some form of source route (e.g., SDRP, GRE) to steer RSVP PATH messages carrying tag bindings along a particular path. Note, however, that this would require a change to the way in which RSVP handles PATH messages, as it would be necessary to store the source route as part of the PATH state.

7. Tag Forwarding Granularities and Forwarding Equivalence Classes

A conventional router has some sort of structure or set of structures which may be called a "forwarding table", which has a finite number of entries. Whenever a packet is received, the router applies a classification algorithm which maps the packet to one of the forwarding table entries. This entry specifies how to forward the packet.

We can think of this classification algorithm as a means of partitioning the universe of possible packets into a finite set of "Forwarding Equivalence Classes" (FECs).

Each router along a path must have some way of determining the next hop for that FEC. For a given FEC, the corresponding entry in the forwarding table may be created dynamically, by operation of the routing protocols (unicast or multicast), or it might be created by configuration, or it might be created by some combination of configuration and protocol.

In tag switching, if a pair of tag switches are adjacent along a tag switched path, they must agree on an assignment of tags to FECs. Once this agreement is made, all tag switches on the tag switched path other than the first are spared the work of actually executing the classification algorithm. In fact, subsequent tag switches need not even have the code which would be necessary to do this.

There are a large number of different ways in which one may choose to partition a set of packets into FECs. Some examples:

1. Consider two packets to be in the same FEC if there is a single address prefix in the routing table which is the longest match for the destination address of each packet;

2. Consider two packets to be in the same FEC if these packets have to traverse through a common router/tag switch;

3. Consider two packets to be in the same FEC if they have the same source address and the same destination address;

4. Consider two packets to be in the same FEC if they have the same source address, the same destination address, the same transport protocol, the same source port, and the same destination port.

5. Consider two packets to be in the same FEC if they are alike in some arbitrary manner determined by policy. Note that the assignment of a packet to a FEC by policy need not be done solely by examining the network layer header. One might want, for example, all packets arriving over a certain interface to be classified into a single FEC, so that those packets all get tunnelled through the network to a particular exit point.

Other examples can easily be thought of.

In case 1, the FEC can be identified by an address prefix (as described in Section 6.1). In case 2, the FEC can be identified by the address of a tag switch (as described in Section 6.1). Both 1 and 2 are useful for binding tags to unicast routes - tags are bound to FECs, and an address prefix, or an address identifies a particular FEC. Case 3 is useful for binding tags to multicast trees that are constructed by protocols such as PIM (as described in Section 6.3). Case 4 is useful for binding tags to individual flows, using, say, RSVP (as described in Section 6.4). Case 5 is useful as a way of connecting two pieces of a private network across a public backbone (without even assuming that the private network is an IP network) (as described in Section 6.5).

Any number of different kinds of FEC can co-exist in a single tag switch, as long as the result is to partition the universe of packets seen by that tag switch. Likewise, the procedures which different tag switches use to classify (hitherto untagged) packets into FECs need not be identical.

Networks could be organized around a hierarchy of FECs. For example, (non-adjacent) tag switches TSa and TSb may classify packets into some set of FECs FEC1,...,FECn. However from the point of view of the intermediate tag switches between TSa and TSb, all of these FECs may be treated indistinguishably. That is, as far as the intermediate

tag switches are concerned, the union of the FEC1,...,FECn is a single FEC. Each intermediate tag switch may then prefer to use a single tag for this union (rather than maintaining individual tags for each member of this union). Tag switching accommodates this by providing a hierarchy of tags, organized in a stack.

Much of the power of tag switching arises from the facts that:

- there are so many different ways to partition the packets into FECs,

- different tag switches can partition the hitherto untagged packets in different ways,

- the route to be used for a particular FEC can be chosen in different ways,

- a hierarchy of tags, organized as a stack, can be used to represent the network's hierarchy of FECs.

Note that tag switching does not specify, as an element of any particular protocol, a general notion of "FEC identifier". Even if it were possible to have such a thing, there is no need for it, since there is no "one size fits all" setup protocol which works for any arbitrary combination of packet classifier and routing protocol. That's why tag distribution is sometimes done with TDP, sometimes with BGP, sometimes with PIM, sometimes with RSVP.

8. Tag switching with ATM

Since the tag switching forwarding paradigm is based on label swapping, and since ATM forwarding is also based on label swapping, tag switching technology can readily be applied to ATM switches by implementing the control component of tag switching.

The tag information needed for tag switching can be carried in the VCI field. If two levels of tagging are needed, then the VPI field could be used as well, although the size of the VPI field limits the size of networks in which this would be practical. However, for most applications of one level of tagging the VCI field is adequate.

To obtain the necessary control information, the switch should be able to support the tag switching control component. Moreover, if the switch has to perform routing information aggregation, then to support destination-based unicast routing the switch should be able to perform Network Layer forwarding for some fraction of the traffic as well.

Supporting the destination-based routing function with tag switching on an ATM switch may require the switch to maintain not one, but several tags associated with a route (or a group of routes with the same next hop). This is necessary to avoid the interleaving of packets which arrive from different upstream tag switches, but are sent concurrently to the same next hop.

If an ATM switch has built-in mechanism(s) to suppress cell interleave, then the switch could implement the destination-based routing function precisely the way it was described in Section 6.1. This would eliminate the need to maintain several tags per route. Note, however, that suppressing cell interleave is not part of the ATM User Plane, as defined by the ATM Forum.

Yet another alternative that eliminates the need to maintain several tags per route is to carry the tag information in the VPI field, and use the VCI field for identifying cells that were sent by different tag switches. Note, however, that the scalability of this alternative is constrained by the size of the VPI space (4096 tags total). Moreover, this alternative assumes that for a set of ATM tag switches that form a contiguous segment of a network topology there exists a mechanism to assign to each ATM tag switch around the edge of the segment a set of unique VCIs that would be used by this switch alone.

The downstream tag allocation on demand scheme is likely to be a preferred scheme for the tag allocation and TIB maintenance procedures with ATM switches, as this scheme allows efficient use of entries in the cross-connect tables maintained by ATM switches.

Implementing tag switching on an ATM switch simplifies integration of ATM switches and routers. From a routing peering point of view an ATM switch capable of tag switching would appear as a router to an adjacent router; this reduces the number of routing peers a router

would have to maintain (relative to the common arrangement where a large number of routers are fully meshed over an ATM cloud). Tag switching enables better routing, as it exposes the underlying physical topology to the Network Layer routing. Finally tag switching simplifies overall operations by employing common addressing, routing, and management procedures among both routers and ATM switches. That could provide a viable, more scalable alternative to the overlay model. Because creation of tag binding is driven by control traffic, rather than data traffic, application of this approach to ATM switches does not produce high call setup rates, nor does it depend on the longevity of flows.

Implementing tag switching on an ATM switch does not preclude the ability to support a traditional ATM control plane (e.g., PNNI) on the same switch. The two components, tag switching and the ATM control plane, would operate in a Ships In the Night mode (with VPI/VCI space and other resources partitioned so that the components do not interact).

---

9. Tag switching migration strategies

Since tag switching is performed between a pair of adjacent tag switches, and since the tag binding information can be distributed on a pairwise basis, tag switching could be introduced in a fairly simple, incremental fashion. For example, once a pair of adjacent routers are converted into tag switches, each of the switches would tag packets destined to the other, thus enabling the other switch to use tag switching. Since tag switches use the same routing protocols as routers, the introduction of tag switches has no impact on routers. In fact, a tag switch connected to a router acts just as a router from the router's perspective.

As more and more routers are upgraded to enable tag switching, the scope of functionality provided by tag switching widens. For example, once all the routers within a domain are upgraded to support tag switching, in becomes possible to start using the hierarchy of routing knowledge function.

10. Summary

In this paper we described the tag switching technology. Tag switching is not constrained to a particular Network Layer protocol - it is a multiprotocol solution. The forwarding component of tag switching is simple enough to facilitate high performance forwarding, and may be implemented on high performance forwarding hardware such as ATM switches. The control component is flexible enough to support a wide variety of routing functions, such as destination-based routing, multicast routing, hierarchy of routing knowledge, and explicitly defined routes. By allowing a wide range of forwarding granularities that could be associated with a tag, we provide both scalable and functionally rich routing. A combination of a wide range of forwarding granularities and the ability to evolve the control component fairly independently from the forwarding component results in a solution that enables graceful introduction of new routing functionality to meet the demands of a rapidly evolving computer networking environment.

---

11. Security Considerations

Security considerations are not addressed in this document.

12. Intellectual Property Considerations

Cisco Systems may seek patent or other intellectual property protection for some or all of the technologies disclosed in this document. If any standards arising from this document are or become protected by one or more patents assigned to Cisco Systems, Cisco intends to disclose those patents and license them under openly

specified and non-discriminatory terms, for no fee.

13. Acknowledgments

Significant contributions to this work have been made by Anthony Alles, Fred Baker, Paul Doolan, Guy Fedorkow, Jeremy Lawrence, Arthur Lin, Morgan Littlewood, Keith McCloghrie, and Dan Tappan.

14. References

15. Authors' Addresses

Yakov Rekhter
Cisco Systems, Inc.
170 Tasman Drive
San Jose, CA, 95134
E-mail: yakov@cisco.com

Bruce Davie
Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824
E-mail: bsd@cisco.com

Dave Katz
Juniper Networks
3260 Jay Street
Santa Clara, CA 95051
E-mail: dkatz@jnx.com

Eric Rosen

Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824
E-mail: erosen@cisco.com

George Swallow
Cisco Systems, Inc.
250 Apollo Drive
Chelmsford, MA, 01824
E-mail: swallow@cisco.com

Dino Farinacci
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134
E-mail: dino@cisco.com