Peter L. Higginson
Michael C. Shand

Development of Router Clusters to Provide Fast Failover in IP Networks

IP networks do not normally provide fast failover mechanisms when IP routers fail or when links between hosts and routers break. In response to a customer request, a DIGITAL engineering team developed new protocols and mechanisms, as well as improvements to the DECNIS implementation, to provide a fast failover feature. The project achieved loss-of-service times below five seconds in response to any single failure while still allowing traffic to be shared between routers when there are no failures.

A DIGITAL router engineering team has refined and extended routing protocols to guarantee a five-second maximum loss-of-service time during a single failure in an Internet Protocol (IP) network. We use the term *router cluster* to describe our improved implementation. A router cluster is defined as a group of routers on the same local area network (LAN), providing mutual backup. Router clusters have been in service since mid-1995.

Background

The Digital Equipment Corporation Network Integration Server (DECNIS) bridge/router is a midrange to high-end product designed and built by a DIGITAL Networks Product Business Group in Reading, U.K.[1] The DECNIS performs high-speed routing of IP, DECnet, and OSI (open system interconnection) protocols and can have the following network interfaces: Ethernet, FDDI (fiber distributed data interface), ATM (asynchronous transfer mode), HSSI (High-Speed Serial Interface), T1/E1 (digital transmission schemes), and lower-speed WAN (wide area network) interfaces. The DECNIS bridge/router is designed around a Futurebus backplane, with a number of semi-autonomous line cards, a hardware based address lookup engine, and a central control processor responsible for the control protocols and route calculation. Data packets are normally handled completely by the line cards and go to the central processor only in exception cases.

The DECNIS routers run a number of high-profile, high-availability, wide-area data networks for telephone service providers, stock exchanges, and chemical companies, as well as forming the backbone of DIGITAL's internal network.

Typically, the DECNIS routers are deployed in redundant groups with diverse interconnections, to provide very high availability. A common requirement is never to take the network down (i.e., during maintenance periods, connectivity is preserved but redundancy is reduced).

Overview

IP is the most widely used protocol for communication between hosts. Routers (or gateways) are used to link hosts that are not directly connected. When IP was originally designed, duplication of WAN links was common but duplication of gateways for hosts was rare, and no mechanisms for avoiding failed routers or broken links between hosts and routers were developed.

In 1994, we began a project to restrict loss-of-service times to below five seconds in response to any single failure; for example, failure of a router or its electrical supply, failure of a link between routers, or failure of the connection between the router and the LAN on which the host resides. In contrast, existing routing protocols have recovery times in the 30- to 45-second range, and bridging protocols are no better. Providing fast failover in IP networks required enhancements to many areas of the router's design to cover all the possible failure cases. It also required the invention of new protocols to support the host-router interaction under IP. This was achieved without requiring any changes to the host IP code.

In this paper, we start by discussing our targets and the behavior of existing routing or bridging protocols and follow this with a detailed analysis of the different failure cases. We then show how we have modified the behavior of the routing control protocols to achieve the desired failover times on links between routers or in response to the failure of intermediate routers. Finally, we describe the new IP Standby Protocol and the mechanisms we developed to achieve fast recovery from failures on the LANs local to the end hosts. This part of the problem is the most challenging because the hosts are of many types and have IP implementations that cannot realistically be changed. Thus all changes have to be made in the routers.

Our secondary aims were to allow the use of router clusters in any existing network configuration, not to constrain failover to simple pairs of routers, to be able to share traffic between available routers, and to continue to use the Internet Control Message Protocol (ICMP) redirect mechanism for optimum choice of router by hosts on a per destination basis. A common problem of hosts is that they do not time out redirects. This problem is avoided by the adoption mechanism within the router cluster. Having met these aims, as well as fast failover, we can justifiably call the result router clusters.

The Customer Challenge

A particular customer, a telecommunications service provider, has an Intelligent Services Network application by which voice calls can be transferred to another operator at a different location. The data network manages the transferral and passes information about the call. The application uses User Datagram Protocol (UDP) packets in IP with retransmission from the application itself.

Because this application requires a high level of data network availability, network designers planned a duplicate network with many paired links and some mesh connections. Particular problems arise when the human initiator becomes impatient if there are delays; however, the more critical requirement was one over which the network designers had no control. The source of the calls is another system that makes a single high-level retransmission after five seconds. If that retransmission does not receive a response, the whole system at the site is assumed to have failed. This leads to new calls being routed to other service sites or suppliers, and manual intervention is required.

To resolve this issue, the customer requested a networking system that would recover from a single failure in any link, interface, or router within a five-second period. The standard test (which both the customer and we use) is to start a once-per-second ping, and to expect to drop no more than four consecutive ping packets (or their responses) upon any event. The five-second maximum break also has to apply to any disruption when the failed component recovers.

To meet the customer challenge, the router group in Reading developed the router cluster implementation on the DECNIS. In the next two sections, we discuss the bridging and routing protocols in use at the start of our project and relate our analysis of the customer's network problems.

Bridging and Routing Default Recovery Times

In a large network, a routing control protocol is essential in order to dynamically determine the topology of the network and to detect failing links. Bridging control protocols may be used similarly in smaller networks or may be used in combination with routing.

Bridging and routing control protocols often have failure recovery times in the order of a minute or more. A typical recovery consists of a detect time during which adjacent routers learn about the failure; a distribution time during which the knowledge is shared, possibly throughout the whole network; and a route recalculation time during which a new set of routes is calculated and passed to the forwarding engine.

Detection times are in the order of tens of seconds; for example, 30 seconds is a common default. The two most popular link-state routing control protocols in large IP networks are Open Shortest Path First (OSPF)[2] and Integrated Intermediate System-to-Intermediate System (Integrated IS-IS).[3] These protocols have distribution "hold downs" (to limit the impact of route flaps) to prevent the generation of a

new control message within some interval (typically 5 or 30 seconds) of a previous one. The distribution of the new information is rapid (typically less than one second), depending primarily on link speeds and network diameter; however, the distribution may be adversely affected by transmission errors which require retransmission. The default retransmission times after packet loss vary between 2 and 10 seconds. The route recalculation typically takes less than one second. These values result in total recovery times after failures (for routing protocols with default settings) in the 45- to 90-second range.

Distance vector routing protocols, such as the Routing Information Protocol (RIP),[4] typically take even longer to recover, partly because the route computation process is inherently distributed and requires multiple protocol exchanges to reach convergence, and partly because their timer settings tend to be fixed at relatively long settings. Consequently, their use is not further considered in this paper.

Similarly, bridging protocols, as standard, use a 15-second timer; one of the worst-case recovery situations requires three timeouts, making 45 seconds in all. Another bridging recovery case requires an unsolicited data packet from a host and this results in an indeterminate time, although a timeout will cause flooding after a period.

In IP protocols, there is no simple way for a host to detect the failure of its gateway; nor is it simple for a router to detect the failure to communicate with a host. In the former case, several minutes may pass before an Address Resolution Protocol (ARP) entry times out and an alternative gateway is chosen; for some implementations, recovery may be impossible without manual intervention. Failure to communicate with a host may be the result of failure of the host itself, which is outside the scope of this project. Alternatively, it may be due to failure of the LAN, or the router's LAN interface. In this case, there exists an alternative route to the LAN through another router, but the routing protocols will not make use of it unless the subnet(s) on the LAN are declared unreachable. This requires either manual intervention or timely detection of the LAN failure by the router.

Analysis of the Failure Cases

The first task in meeting the customer's challenge was to analyze the various failure and recovery modes and determine which existing management parameters could be tuned to improve recovery times. After that, new protocols and mechanisms could be designed to fill the remaining shortcomings.

A typical network configuration is shown in Figure 1. The target network is similar but has more sites and many more hosts on each LAN. Many of the site routers are DECNIS 500 routers with one or two WAN links and two Ethernets. The second Ethernet is used as a management rail and as a redundant local path between routers one and two (R1–R2) and between routers three and four (R3–R4).

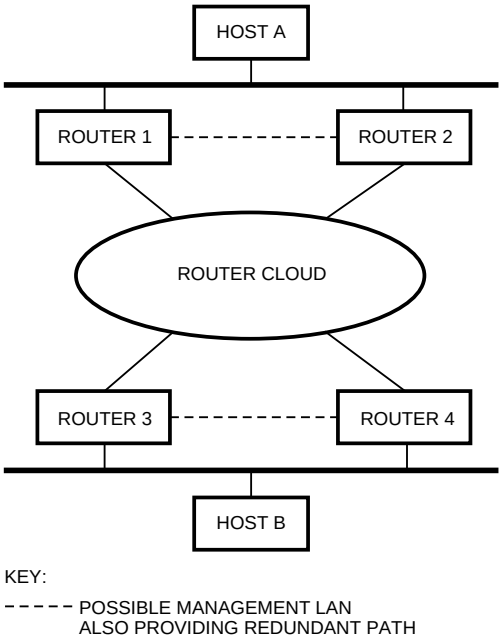

Figure 1
Typical Configuration for Router Cluster Use

In the original plans for the customer network, the router cloud consisted of groups of routers at two or three central sites and pairs of links to the host sites. In designing our solution, however, we tried to allow any number of routers on each LAN, interconnected by a general mesh network. For test purposes, both we and the customer used this set-up with direct R1–R3 and R2–R4 T1 links as the network cloud.

We have to consider what happens to packets traveling in each direction during a failure: there is little gain in delivering the data and losing the acknowledgments. Since the direction of data flow does not give rise to additional complications in the network cloud, there are just two failure cases:

1. Failure of a router in the network cloud
2. Failure of a link in the network cloud

We keep these cases distinct because the failure and recovery mechanisms are slightly different.

We also need to consider a failure local to one of the LANs on which the hosts are attached. A failure here has two consequences: (1) The packets originated by the host must be sent to a different router, and (2) The response packets from the other host through the network cloud must also be sent to a different router, so

that it can send them to the host. We break down this type of failure into the following three cases:

3. Packets from the host to a failed or disconnected router
4. Packets to the host when the router fails
5. Packets to the host when the router interface fails

Note that we are using the term *router interface failure* to include cases in which the connector falls out or some failure occurs in the LAN local to the router (such that the router can detect it). In practice, failure of an interface is rare. (Removing the plug is not particularly common in real networks but is easy to test.) Figure 2 shows these failure cases; this configuration was also used for some of the testing.

Recovery of a link that previously failed causes no problems because the routers will not attempt to use it until after it has been detected as being available. Prior to that, they have alternate paths available. Recovery of a failed router can cause problems because the router may receive traffic before it has acquired sufficient network topology to forward the traffic correctly. Recovery of a router is discussed more fully in the section on Interface Delay.

Can Existing Bridging or Routing Protocols Achieve 5-Second Failover in a Network Cloud?

In this section, we discuss the failure of a router and the failure of a link in the network cloud (cases 1 and 2). The customer requested enhanced routing, and the existing network was a large routed WAN, so enhancing bridging was never seriously considered. Our experience has shown that the 15-second bridge timers can be reduced only in small, tightly controlled networks and not in large WANs. Consequently, bridging is unsuitable for fast failover in large networks.

For link-state routing control protocols such as OSPF and Integrated IS-IS, once a failure has been detected recovery takes place in two overlapping phases: a flood phase in which information about the failure is distributed to all routers, and a route calculation phase in which each router works out the new routes. The protocols have been designed so that only local failures have to be detected and manageable parameters control the speed of detection.

Detection of failure is achieved by exchanging Hello messages on a regular basis with neighboring routers. Since the connections are usually LAN or Point-to-Point Protocol (PPP) (i.e., with no link-layer acknowledgments), a number of messages must be missed before the adjacency to the neighbor is lost. The messages used to maintain the adjacency are independent of other traffic (and in a design like the DECNIS may be the only traffic that the control processor sees). Typical default values are messages at three-second intervals and 10 lost for a failure, but it is possible to reduce these.

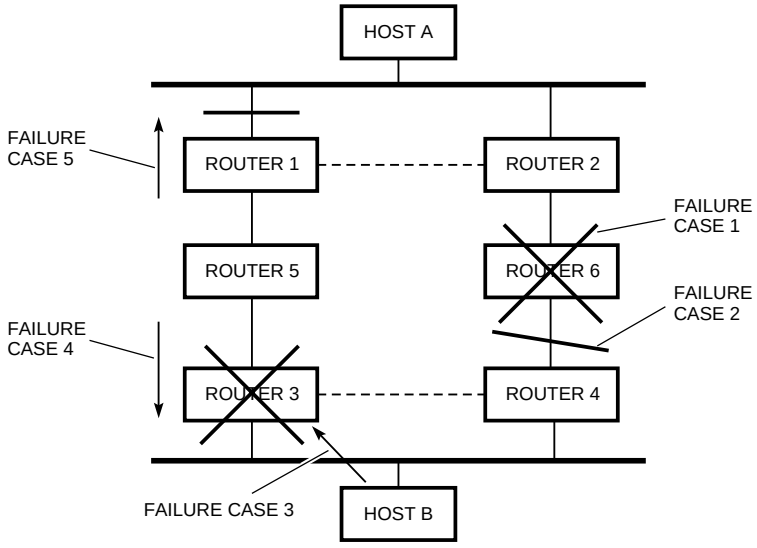

Figure 2
Diagram of Failure Cases Targeted for Recovery

Decreasing the Routing Timers

The default timer values are chosen to reduce overheads, to cover short outages, and to ensure that it is not possible for long packets to cause the adjacency to expire accidentally by blocking Hello transmission. (Note transmission of a 4,500-byte packet on a 64 kilobit-per-second link takes half a second, and queuing would normally require more than a packet time.) However, with high-quality T1 or higher link speeds in the target network and priority queuing of Hellos in the DECNIS, it is acceptable to send the Hellos at one-second intervals and count three missed as a failure. (Although we have successfully tested counts of two, we do not recommend that value for customers on WAN links because a single link error combined with a delay due to a long data packet would cause a spurious failure to be detected.) The settings of one second and three repeats were within the existing permitted ranges for the routing protocols.

When these shorter timers are used, it is important that any LANs in the network should not be overloaded to the extent that transmissions are delayed. The network managers should monitor WAN links and disable any links that have high error rates. Given the duplication of routes, it is better to disable and initiate repairs to a bad link than to continue a poor service. Many customers, with less controlled networks and less aggressive recovery targets, have adopted the router cluster system but kept to more conservative timers (such as 1 second and 10 repeats).

Implementation and Testing Issues

In some cases, a failed link may be detected at a lower level (e.g., modem signals or FDDI station management) well before the routing protocol realizes that it has stopped getting Hellos and declares the adjacency lost. (This can lead to good results during testing, but it is essential also to test link-failure modes that are not detected by lower levels.) In the worst case, however, both the detection of a failed router or the detection of a failed link rely on the adjacency loss and so have the same timings.

Loss of an adjacency causes a router to issue a revised (set of) link-state messages reflecting its new view of the local topology. These link-state messages are flooded throughout the network and cause every router in the network to recalculate its route tables. However, because the two or more routers will normally time out the adjacency at different times, one message arrives first and causes a premature recalculation of the tables. Therefore it may require a subsequent recalculation of the route tables before a new two-way path can be utilized. We had to tune the router implementation to make sure that subsequent recalculations were done in a speedy manner.

During initial testing of these parameters, we discovered that failure of certain routers represented a more serious case. However discussion of this is deferred to the later section The Designated Router Problem.

Our target five seconds is made up of three seconds for the failure to be detected, leaving two seconds for the information about the failure to be flooded to all routers and for them to recalculate their routes. Within the segment of the network where the recovery is required, this has been achieved (with some tuning of the software).

Recovery from Failures on the LANs Local to the End Hosts

The previous section shows that we can deal with router failure and link failure in the network cloud (cases 1 and 2). Here we consider cases 3, 4, and 5, those that deal with failures on the LANs local to the end hosts.

From the point of view of other routers, a failed router on a LAN (case 4) is identical to a failed router in the network cloud (case 1): a router has died, and the other routers need to route around it. Failure case 4 therefore is remedied by the timer adjustments described in the previous section. Note that these timer adjustments are an integral part of the LAN solution, because they allow the returning traffic to be re-routed. These timer adjustments cannot work properly if the LAN parts of router clusters are using an inappropriate routing control protocol such as RIP[4], which takes up to 90 seconds to recover from failures.

Detecting LAN Failure at the Router

A solution to case 5—packets to the host when the router interface fails—for IP requires that the router can detect a failure of its interface (for example, that the plug has been removed). If the LAN is an FDDI, this is trivial and virtually instantaneous because continuous signals on the ring indicate that it is working and the interface directly signals failure. For Ethernet, we faced a number of problems, partly due to our implementation and partly due to the nature of Ethernet itself. We formed a small team to work on this problem alone.

Because of the variety of Ethernet interfaces that might be attached, there is no direct indication of failure: only an indirect one by failure to successfully transmit a packet within a one-second interval. For maximum speed, the DECNIS implementation queues a ring of eight buffers on the transmit interface and does not check for errors until a ring slot is about to be reused. This means that an error is only detected some time after it has occurred, consuming much of our five-second budget.

The control software in the DECNIS management processor has no direct knowledge of data traffic because it passes directly between the line cards. Therefore it sends test packets at regular intervals to find out if the interface has failed. By sending large test packets occupying many buffers, it ensures that the ring circulates and errors are detected. Initially, we

reduced the timers and increased the frequency of test packets to be able to detect interface failure within three seconds. (The test packets have the sender as destination so that no one receives them and, as usual, more than one failure to transmit is required before the interface is declared unusable.)

This initial solution caused several problems when it was deployed to a wider customer group; we had more complaints than previously about the bandwidth consumed by the test messages and, more seriously, a number of instances of previously working networks being reported as unusable. These problem networks were either exceptionally busy or had some otherwise undetected hardware problem. Over time, the networks with hardware problems were fixed, and we modified the timers to avoid false triggering on very busy networks. Clearly, the three-second target required more thought.

Several enhancements have since been made. First, the timers are user configurable so that the network managers can trade off between aggressive recovery times, bandwidth used, and false detection. Second, the test packet generator takes into account other packets sent by the control processor such that they are only sent to the size and extent required for the total traffic to cause the ring to circulate. This is a significant improvement because the aggressive routing timers discussed previously cause Hello packets to be sent at one-second intervals, which is often sufficient not to require extra test packets. Third, the line card provides extra feedback to the control program about packets received and the transmission of packets not originated by the control processor. This feedback gives an indication of successful operation even if some transmits are failing.

Re-routing Host Traffic When a Router or Router Connection Fails

Case 3 was by far the most difficult problem to solve. IP does not provide a standard mechanism to re-route host traffic when a router fails, and the only method in common use (snooping RIP messages in the hosts) is both "deprecated" by the RFCs and has fixed 45-second timers that exceed our recovery target. Customers have a wide range of IP implementations on their hosts, and reliance on nonstandard features is difficult. The particular target application for this work ran on personal computer systems with a third-party IP stack, and we obtained a copy for testing. Such IP stacks sometimes do not have sophisticated recovery schemes and discussion with various experts led us to believe that we should not rely on any co-operation from the hosts.

Among other objectives, we wanted to be independent of the routing control protocol in use (if any), to permit both a mesh style of networking and more than two routers in a cluster, and to continue to route traffic by reasonably optimal routes. In addition, we wished to not confuse network management protocols about the true identity of the routers involved and, if possible, to share traffic over the WAN links where appropriate.

Electing a Primary Router

In our solution, the first requirement is for other routers on the LAN to detect that a router has failed or become disconnected, and to have a primary router elected to organize recovery. This is achieved by all routers broadcasting packets (called IP Standby Hellos) to other routers on the LAN every second. The highest priority (with the highest IP address breaking ties) router becomes the primary router, and failure to receive IP Standby Hellos from another router for n seconds (three is the default) causes it to be regarded as disconnected. This condition may cause the selection of a new primary router, which would then initiate recovery to take traffic on behalf of the disconnected router.

The IP Standby Hellos are sent as "all routers multicasts" and therefore do not add additional load to hosts. They are UDP datagrams[5] to a port we registered for this purpose (digital-vrc; see the Internet Assigned Numbers Authority [IANA] on-line list). The routers are manually configured with a list of all routers in the cluster. To make configuration easier and less error prone, the list on each router includes itself, and hence an identical set of configuration parameters can be used for all the routers in a cluster. Automatic configuration was rejected because of the problem of knowing which other routers should exist.

Function of the Primary Router in ARP Mode

Our first attempt (called ARP Mode) uses a fake IP address (one per subnet for a LAN with multiple subnets), which the current primary router adopts and the hosts have configured as their default router. The primary router returns its own media access control (MAC) address when the host broadcasts an ARP request (using the standard ARP protocol[6]) for the fake IP address and thus takes the traffic from the host. After a failure, a newly elected primary router broadcasts an ARP request containing the information that the fake IP address is now associated with the new primary router's MAC address. This causes the host to update its tables and to forward all traffic to the new primary router.

The sending of ICMP redirects[7] by the routers has to be disabled in ARP mode. Redirects sent by a router would cause hosts to send traffic to an IP address other than the fake IP address controlled by the cluster, and recovery from failure of that router would then be impossible. Disabling redirects causes an additional

problem. If the primary router's WAN link fails, all the packets have to be inefficiently forwarded back over the LAN to other routers. To avoid this problem, we introduced the concept of monitored circuits, whereby the priority of a router to become the primary depends on the state of the WAN link. Thus, the primary router changes when the WAN link fails (or all the links fail if there are several), and the hosts send the packets to the new primary (whose WAN link is still intact).

ARP mode has a number of disadvantages. It does not necessarily use an optimum route when the WAN links form a mesh rather than the simple pair case, because redirects have to be disabled. The monitored circuit concept works only on the first hop from the router; more distant failures cannot change the IP Standby priority and may result in inefficient routing. Most seriously, the rules for hosts acting on information in ARP requests have only a "suggested implementation" status in the RFCs, and we found several hosts that did not change when requested or were very slow in doing so. (Note that we did consider broadcasting an ARP response, but there is no allowance in the specifications for this message to be a broadcast packet, whereas an ARP request is normally a broadcast packet.)

MAC Mode IP Standby (to Re-route Host Traffic)

To solve these problems, we looked for a mechanism that did not rely on any host participation. The result was what we termed MAC mode. Here, each router uses its own IP address (or addresses for multiple subnets) but answers ARP requests with one of a group of special MAC addresses, configured for each router as part of the router cluster configuration. When a router fails or becomes disconnected, the primary (or the newly elected primary) router adopts the failed router. By adopt, we mean it responds to ARP requests for the failed router's IP address with the failed router's special MAC address, and it receives and forwards all packets sent to the failed router's special MAC address (in addition to traffic sent to the primary router's own special MAC address and those of any other failed routers it has adopted).

The immediate advantages of MAC mode are that ICMP redirects can continue to be used, and, providing the redirects are to routers in the cluster, the fast failover will continue to protect against further failures. The mechanism is completely transparent to the host. In a cluster with more than two routers, the primary router will use redirects to cause traffic (resulting from failure) to use other routers in the cluster if they have better routes to specific destinations. Thus multiple routers in a cluster and mesh networks are supported. This also solves the problem of hosts not timing out redirects (an omission common to many IP implementations derived from BSD), because the redirected address has been adopted.

In MAC mode, the hosts are configured with the IP address of any router in the cluster as the default gateway. (The concept that it does not matter which router is chosen is one of the hardest for users to accept.) Some load sharing can be achieved by setting different addresses in different hosts.

Since the DECNIS is a bridge router, it has the capability to receive all packets on Ethernet and many MAC addresses on FDDI; thus all packets on all the special MAC addresses are seen by all routers in the cluster, and its own and those of any adopted routers are forwarded. The special MAC addresses used are those associated with the unused DECnet area 0. They are ideal because they are part of the locally administered group and have implementation efficiencies in the DECNIS because the DECnet hi-ord (AA-00-04-00) is already decoded, and they are 16 addresses differing in one nibble only (i.e., AA-00-04-00-0x-00, where x is the hexadecimal index of the router in the cluster). Note that ARP requests sent by the router must also contain the special MAC address in the source hardware address field of the ARP packet, otherwise the hosts' ARP tables may be updated to contain the wrong MAC address.

MAC mode has minor disadvantages. Initially, it is easy to spread the load over a number of routers; however, this can be lost after redirects. In addition, a small chance of packet duplication exists during recovery because there may be a short period when both routers are receiving on the same special MAC address (which does not happen in ARP mode because the host changes the MAC address it is using). This is preferable to a period when no router is receiving on that address.

Interface Delay

Recently, we added an interface delay option to ameliorate a situation more likely to occur in large networks. In this situation, a router, rebooting after a power loss, a reboot, or a crash, reacquires its special MAC address before it has received all of the routing updates from neighboring routers and thus drops packets sent to it (and worse, returns "unreachable" to the host). Typically, the main LAN initialization would be delayed for 30 seconds while routing table updates were received over the WAN interfaces and any other LAN interfaces. The backup continues to operate during this 30 seconds. (Note that with Integrated IS-IS, we could have delayed IP on the whole router, but we did not do this because it would not have worked for OSPF, which requires IP to do the updates.) We use a fixed configurable time rather than attempting to detect the end of updating, because determining completion is difficult if the network is in a state of flux or the router's WAN links are down.

Redirects and Hosts That Ignore Them

When a router issues an ICMP redirect, the RFCs state that it must include its own IP address in the redirect packet. A host is required to ignore a redirect received from a router whose IP address is not the host's next hop address for the particular destination address. Therefore, it is necessary to ensure that the address of the failed router is correctly included when issuing a redirect on its behalf. In the DECNIS implementation, because the destination MAC address of a received packet is not available to the control processor, the primary router cannot tell whether a redirect has to be issued on behalf of itself or one of the adopted routers. The primary router therefore issues multiple redirects—one for each adopted router (in addition to its own). Since redirects are rare, this is not a problem, but they could be avoided by passing the MAC destination address of the original packet (or just five bits to flag a special MAC address and say which it is) to the control processor.

It is contrary to the basic IP rules for hosts to ignore redirects.[8] Despite the rules, some hosts do ignore redirects and continue sending traffic which has to be sent back over the same LAN. These cause problems in all networks because of the load, and, in the DECNIS implementation, because every time the line card recognizes a redirect opportunity, it signals the control processor to consider sending a redirect. This may happen at data packet rates and is a severe load on the control processor, which slows down processing of routing updates and might then cause our five-second recovery target to be exceeded.

To reduce the problems caused by hosts ignoring redirects, we improved the implementation to rate-limit the generation of redirect opportunity messages by the line cards. We also recommend in the documentation that, where it is known that hosts ignore redirects (or their generation is not desired), the routers be connected by a lower-cost LAN than the main service LAN (such as the management LANs shown in Figure 1). Normally, this would mean linking (just) the routers by a second Ethernet and setting its routing metric so that it is preferred to the main LAN for packets that would otherwise traverse back on the main LAN to the other router. This has two advantages. Such packets do not consume double bandwidth and cause congestion on the main LAN, and they pass only through the fast-path parts of the router, which are well able to handle full Ethernet bandwidth.

In MAC mode, it is also possible to define a router that does not actually exist (but has an IP address and a special MAC address) and is adopted by another router, depending on the state of monitored WAN circuits. Setting this as the default gateway is another way of coping with hosts that ignore redirects.

Special Considerations for Bridges

We do not recommend putting a bridge or layer 2 switch between members of a router cluster, because during failover, action would be required from the bridge in order for the primary router to receive packets that previously were not present on its side of the bridge. We cannot rely on this being the case, so we must have a way of allowing bridges to learn where the special MAC addresses currently are. More importantly, if bridges do not know where the special MAC addresses are, they often use much less efficient (flooding) mechanisms.

For greater traceability (and simpler implementation), we use the router's real MAC address as the source address in data packets that it sources or forwards. We use the special MAC address as the source address in the IP Standby Hellos. Since the Hello is sent out as an IP multicast, it is seen by all bridges or switches in the local bridged network and causes them to learn the location of the address (whereas data packets might not be seen by non-local bridges). Since we are sending the Hellos every one second anyway, there is no extra overhead.

When a primary router has adopted routers, it cycles the source MAC address used for sending its Hello between its own special MAC address and those of the adopted routers. We also send out an additional Hello immediately when we adopt a router to speed up recognition of the change.

Since the same set of special MAC addresses is used by all router clusters, we were concerned that a bridge that was set up to bridge a non-IP protocol (e.g., local area transport [LAT]) but not to bridge IP, might be confused to see the same special MAC address on more than one port. (This has been observed to happen accidentally, and the resultant meltdown has led us to avoid any risk, however slight, of this happening.) Hence we make 16 special MAC addresses available and recommend to users that they allocate them uniquely within a bridged domain, or at least use disjoint sets on either side of a bridge.

The Designated Router Problem

While testing router failures, we discovered additional delays during recovery due to the way in which link-state protocols operated on LANs. In these cases, the failure of routers not handling the data packets can also result in interruption of service due to the control mechanisms used.

For efficiency reasons in link-state routing protocols, when several routers are connected to a LAN, they elect a designated router and the routing protocols treat the LAN as having a single point-to-point connection between each real router and a pseudo router maintained by the designated router (rather

than connections between all the routers). The designated router issues link-state packets on behalf of the pseudo router, showing it as having connections to each real router on the local LAN, and each router issues a link-state packet showing connection to the pseudo router. This mechanism operates in a broadly similar way in both Integrated IS-IS and OSPF; the primary difference being that the OSPF election exhibits hysteresis, thus minimizing unnecessary designated router changes.

For routing table calculations, a transit path over the LAN is taken from a router to the pseudo router and then to another router on the LAN. Hence any change in pseudo router status disrupts calculation of the network map.

When a designated router fails, a slew of updates occurs; each router on the LAN loses the adjacency to the old designated router and issues a new link-state packet. Next, the new designated router is elected (or in the case of OSPF, the backup designated router takes over), and each router issues a link-state packet showing a link to it. In parallel, the new designated router issues a set of link-state packets showing its connections. This is a new router on the network as far as the other routers are concerned; the old designated router stays, disconnected, in the tables for as long as 20 minutes to an hour. This happens at level 1 and at level 2 in Integrated IS-IS, resulting in twice as many updates. The interactions are complex; in general, they result in the sending of multiple, new link-state messages.

Apart from the pure distribution and processing problem of these updates and new link-state packets, there are deliberate delays added. A minor one is that updates in Integrated IS-IS are rate-limited on LANs (to minimize the possibility of message loss). A major one is that a particular link-state packet cannot be updated within a holding time from a previous update (to limit the number of messages actually generated). The default holding time is 30 seconds in Integrated IS-IS; it can be reduced to 1 second in the event we found that the best solution was to allow as many as 10 updates in a 10-second period. The reason for this is that the first update usually contains information about the disconnection and it is highly desirable to get the update with the connection out as fast as possible. In addition, in the wider network, an update can overtake and replace a previous one.

With OSPF, the protocol defines a minimum holding time of five seconds, which limits the recovery time when the designated router fails. The target customer's network was using Integrated IS-IS, and so we were able to achieve the five-second recovery even when the designated router failed. (Note that with two routers, one must be the designated router so it is not a rare case.) We have not, so far, felt that it is worthwhile to break the rules by allowing a shorter holding time for OSPF.

Conclusions

We successfully designed and implemented router clusters for the DECNIS router with shared workload and interruptions after failures of less than five seconds in both LAN and WAN environments. This capability has been deployed in the product since the middle of 1995. An Internet Engineering Task Force (IETF) group is currently attempting to produce a standard protocol to meet this need.[9]

Acknowledgments

Various members of the router engineering team in Reading, U.K. assisted with ideas for this work. In particular, we must mention Dave Forster who implemented the high-level IP changes, Chris Szmidt who implemented the line card forwarding, and John Rigby who implemented the bit in-between and the Ethernet cable-out detection.

References

1. D. Brash and S. Bryant, "The DECNIS 500/600 Multiprotocol Bridge Router and Gateway," *Digital Technical Journal,* vol. 5, no. 1 (Winter 1993): 84–98.
2. J. Moy, "OSPF Version 2," Internet Engineering Task Force, RFC 1583 (March 1994).
3. R. Callon, "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments," Internet Engineering Task Force, RFC 1195 (December 1990).
4. C. Hedrick, "Routing Information Protocol," Internet Engineering Task Force, RFC 1058 (June 1988).
5. J. Postel, "User Datagram Protocol," SRI Network Information Center, Menlo Park, Calif., RFC 768 (August 1980).
6. D. Plummer, "Ethernet Address Resolution Protocol," Internet Engineering Task Force, RFC 826 (November 1982).
7. J. Postel, "Internet Control Message Protocol," Internet Engineering Task Force, RFC 792 (September 1981).
8. R. Braden, "Requirements for Internet Hosts—Communication Layers," Network Information Center, RFC 1122 (October 1989).
9. R. Hinden, S. Knight, D. Weaver, D. Whipple, D. Mitzel, P. Hunt, P. Higginson, and M. Shand, "Virtual Router Redundancy Protocol," Internet Drafts <draft-ietf-vrrp-spec-03.txt> (October 1997).

Biographies

Peter L. Higginson
Peter Higginson manages the advanced development work on router products for DIGITAL's Internetworking Products Engineering Group in Reading U.K. (IPEG-Europe). His responsibilities include improving communications on customers' large networks. Most recently, he contributed to the corporate Web gateway strategy and future router products. Peter was issued one patent on efficient ATM cell synchronization and has applied for several other patents related to networks. He has published many papers, including one on a PDP-9 for DECUS in 1971. Before joining DIGITAL in 1990, Peter was the software director for UltraNet Ltd. (now part of the Anite Group), a maker of X.25 equipment. For 12 years before that, he was a lecturer in the Department of Computer Science, University College London. He received an M. Sc. in computer science from University of London in 1970 and a B. Sc. (honours) in mathematics from University College London in 1969. Peter connected the first non-U.S. host to the Arpanet in 1973.

Michael C. Shand
Mike Shand is a consulting software engineer with DIGITAL's Network Products Business in Reading, U.K. He is currently involved in the design of IP routing algorithms and the system-level design of networking products. Formerly, Mike was a member of the NAC (Networks and Communications) Architecture Group where he designed DECnet OSI, Phase V routing architecture. Before joining DIGITAL in 1985, Mike was the assistant director (systems) of the Computing Centre at Kingston University. He earned an M.A. in the natural sciences from the University of Cambridge in 1971 and a Ph. D. in surface chemistry from Kingston University in 1974. He was awarded six patents (and has filed another) in various aspects of networking.