# Fault Detection in Routing Protocols *

Daniel Massey
Computer Science Department
UCLA
Los Angeles, CA 90095
masseyd@cs.ucla.edu

Bill Fenner
AT&T Labs Research
75 Willow Rd.
Menlo Park, CA 94025
fenner@research.att.com

## Abstract

*Routing protocol faults cause problems ranging from an inability to communicate to excessive routing overhead. This paper proposes a system for detecting a wide range of routing protocol faults. Our system deploys virtual routers called RouteMonitors to monitor a routing protocol. We deployed RouteMonitors in the MBone's DVMRP infrastructure and uncovered a number of faults. We were also able to identify the cause of these errors and deploy fixes. The result was a substantial improvement in DVMRP route stability. This paper reviews the RouteMonitor approach and details the results achieved in the MBone.*

## 1 Introduction

Detecting routing protocol faults remains an open challenge for protocol designers as well as network administrators. Some routing protocol faults result in obvious problems. For example, end systems will be unable to communicate with each other if a fault created an invalid route between them. Other faults may result in more subtle problems. Packets might be routed along suboptimal or frequently changing routes. Even if packets arrive correctly, faults may cause extra communication and computation by the routing protocol. As networks grow in size and complexity, these subtle problems can develop into widespread breakdowns. An ideal network would have a mechanism to automatically detect routing protocol faults.

Routing protocol faults are caused by problems ranging from simple hardware failures to intentional attacks. Most routing protocols are designed to cope with only simple **fail-stop** behavior. A fail-stop fault occurs when a link or router simply stops operating. A fail-stop fault may result in some short period of instability, but nearly all routing protocols detect these device failures and eventually recover to a correct state. Most currently deployed routing protocols are not designed to cope with more complex faults such as hardware malfunctions, incorrect router configurations, incompatible protocol implementations, or intentional attacks. These problems present a more complex fault model where a device malfunctions but does not stop operating. Additional mechanisms are needed to monitor the routing protocols and detect these type of faults.

This paper presents the *RouteMonitor* approach for monitoring a routing protocol and describes the results achieved by deploying this system in the MBone's DVMRP routing infrastructure. Beginning in summer 1997, our *RouteMonitor* was used to collect data from the MBone's DVMRP infrastructure. Our observation showed that only 67% of DVMRP routes remained stable during an average hour and some routes changed over 150 times every hour. Using RouteMonitor, we were able to identify faults which were responsible for this high level of instability. In the intervening time, we worked to encourage the deployment of patches which corrected these faults. By fall 1998, most of the instability had been eliminated and we observed that an average of 95% of DVMRP routes remained stable during each hour. This paper presents our RouteMonitor approach, a detailed picture of the instability discovered in the MBone, and the specific faults responsible for this instability.

### 1.1 Previous Work

Most of the systems and tools for detecting routing protocol faults can be classified as either active monitoring tools or passive monitoring tools. Active monitoring tools add something to the system and watch for an expected result. Passive tools observe the system and report on unusual events. We briefly outline some the most widely used systems and tools below.

Some of the best known active fault detection tools are

*This material is based upon work partially supported by the National Science Foundation under Grant No. 9729498. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

route tracing tools. The *traceroute* tool sends packets of varying lifetimes to learn the route to a destination. An advanced version of traceroute, *pathchar*, sends more packets and generates more statistics such as estimated throughput on each link. The *mtrace* tool traces the multicast route from a destination to a source using special IGMP protocol messages.

The Real-Time Transport Protocol (RTP) specifies a mechanism for the participants in a multicast session to report their observed loss rates from each traffic source in that session. These loss rates can then be passively monitored by any site capable of receiving the multicasted session. Systems such as rtpmon[17] and MHealth[14] combine these statistics with knowledge of the network topology to help track down faults. Since these statistics can only be gathered with active senders and receivers, MRM[20] specifies a protocol to cause test stations (routers or hosts) in the middle of the network to source or sink RTP streams.

As networks grow in scale, wide scale active monitoring requires more traffic generation or more reliance on generating representative forms of traffic. One must generate enough traffic to find the problem, but generating traffic may be costly or create problems of its own. Network measurement projects can lead to a better understanding of how the network operates and lead to the discovery of underlying network faults. Projects such as NIMI [1] and MINC [4] concentrate on gathering performance statistics on the network. These works tend to focus on higher layer issues such as congestion, measuring flows, and so forth.

Passive fault detection is most commonly done using the well known management protocol SNMP[6]. SNMP allows one to query a fixed a set of variables maintained in a management information base (MIB). An SNMP agent sends a request asking for a particular MIB variable and the device maintaining the MIB returns the result. SNMP also defines methods for authorizing access to data so that only those with proper permissions could query the device. A network operations center can monitor the status of network devices and potentially detect problems by polling the device or using traps.

SNMP provides access to a wide range of low level information. This low level access has tremendous advantages for some types of network management, but also creates limitations for routing protocol fault detection. The type of routing protocol data available is fixed by the MIB. Limitations are placed on MIB variables to reduce the complexity of implementation. This restricts the type and amount of data available for fault detection. In addition, one is relying on the potentially faulty device to correctly maintain and report its state. Finally, even if the device does report correctly, frequently sending SNMP queries to a production router adds additional duties and loads to those crucial devices.

Recently a number of experiments have used passive monitoring approaches that rely more heavily on observing packets. In [13] Labovitz, Jahanian, and Malan observed BGP route updates exchanged at major interchange points and noticed BGP faults such as duplicate withdraws for routes which were never advertised. A more general packet filter/monitoring tool by Malan and Jahanian called *windmill* is described in [15]. In [18], Thompson, Miller, and Wilder designed an OC-3 packet capture system and used it to analyze traffic and routing in MCI's backbone. The MOAT [16] and CoralReef [5] projects follow on with passive and some active measurement and analysis.

## 2 The RouteMonitor Approach

The fault detection approach taken here is to deploy a collection of monitoring devices called RouteMonitors. RouteMonitors observe the protocol messages (e.g. route updates) exchanged between routers. Based on the updates observed, RouteMonitor imitates the actions of router. Upon receiving an update, the RouteMonitor executes the routing protocol and constructs a forwarding table. In this sense, RouteMonitor acts as a router executing the protocol in real time. However, RouteMonitors do not advertise any routes or change the route computation in a network. Since no route advertisements are sent by a RouteMonitor, no data packets are forwarded to the RouteMonitor. Instead of performing the normal packet forwarding operations of a router, the RouteMonitors devote resources to tracking routing changes and offer points at which to observe how the routing protocol is performing.

Beginning in summer 1997, we used RouteMonitors to analyze the MBone's DVMRP infrastructure. Our results showed that during an average hour, only 67% of the DVMRP routes remained stable. Some of the worst routes changed an average of over 150 times per hour. This route instability created both communication problems for a number of sources and also generated extra load for all routers in the DVMRP infrastructure.

We were able to identify the main causes for this instability and implement fixes. By late 1998, the fixes for these problems had been deployed at most problem sites and 95% of DVMRP routes remained stable during an average hour. No routes averaged over 100 changes per hour. The complete results are detailed below.

### 2.1 The MBone and the DVMRP RouteMonitor

IP multicast allows a sender to address a packet to a group rather than to a specific receiver. The sender need not belong to the group and need not know any of the group members. The sender simply addresses the packet to a multicast group address. The routing protocol is responsible for

determining who belongs to the group and delivering a copy of the packet to each group member. The collection of multicast capable hosts, routes, and links forms the Multicast Backbone or MBone[8].

In the MBone, the hosts and routers use the IGMP[7] protocol to learn group membership information. The MBone uses routing protocols such as DVMRP[19], PIM[9], CBT[2], and MBGP[3] to compute a routing tree and forward packets from the sender to the receivers. While the details of these protocols are not essential for the remainder of this discussion, it should be clear that some underlying topology information is necessary to compute the forwarding trees. In particular, the routing protocols rely on being able to compute the shortest path to a source or, in the case of PIM-SM or CBT, a "Rendez-vous Point" or "Core". If this shortest path information is invalid, the resulting tree will also be invalid. We used RouteMonitor to look for faults in one of the routing protocols used to compute this topology information.

In summer 1997, nearly all MBone routers used the DVMRP[1] topology learning algorithm. DVMRP is a distance vector algorithm [10, 12]. A distance vector algorithm finds the neighboring router which is closest to the source and its distance to the source (in DVMRP, this is called the "previous hop"). A router periodically sends to its neighbors updates listing its distance to each source. Upon receiving an update, a router compares the distance learned from the update with the router's own distance to the source. If the distance learned from the update is shorter, then the sender of the update becomes the previous hop to the source and the distance to the source becomes the distance learned from the update.

The DVMRP topology database maintained at a router consists of one entry for each known source network. The entry for a source contains the source network address, the netmask for the address, the previous hop router along the path to the address, and the distance to the network. The maximum distance allowed by DVMRP is 31. A distance of 32 (called "infinity") indicates that a source is unreachable The DVMRP algorithm used in the MBone requires a router to send at least one update for a source every sixty seconds. If 140 seconds elapse without hearing an update from the previous hop to a source, the route to the source times out. If the distance to a source changes, a router may trigger an update for the source rather than wait for the next sixty second update interval. Routers multicast updates onto each of their attached subnets or links.

A DVMRP RouteMonitor subscribes to the multicast group on which route updates are transmitted, 224.0.0.2. This allows RouteMonitor to receive route updates from all multicast routers attached to the subnet. The route updates can be filtered based on the sender's IP address so that only route updates from a particular router are accepted. The updates from the selected router or routers are processed according to the DVMRP protocol. The RouteMonitor maintains a standard routing table listing the best distance to each source address appearing in a route update. If a later update reports a new distance, the RouteMonitor changes its distance to the source accordingly.

In addition to the standard routing table, RouteMonitor keeps track of additional statistics for each source. A **route flap** occurs anytime the distance or previous hop to a source changes. RouteMonitor keeps a weekly, hourly, and monthly count of the number of route flaps for each source. RouteMonitor also stores time stamps indicating when a source first appeared in an update and last appeared in update. Using this information, it tracks sources which vanish from and reappear in the routing table. The frequency of metric changes is an extremely simplistic measure and would be expected to only capture the most pathological behavior. However, in our initially deployed RouteMonitor, a huge amount of the network was behaving in pathological ways. We decided to analyze this behavior before progressing to analysis of more subtle measures.

The data is collected in real time, and the RouteMonitor maintains web pages which show the current routing table and the statistics associated with each source. A long term statistics file is also maintained and can be searched from the web page. Search parameters available include number of route changes, number of updates, up time, first advertisement, last advertisement, and so forth. For example, one can use the web page to report all sources matching a CIDR address, changing some range of values during the current hour (week/month), and first reported on or before some date. RouteMonitor can also output a real time view of each update received, but this feature is used sparingly.

## 3 A View from a DVMRP RouteMonitor

Our main RouteMonitor was deployed at the UCLA Computer Science Department. UCLA had a single connection to the MBone via ISI. The local MBone topology is shown in Figure 1. Since UCLA has only one connection to the general MBone, the view from the subnet attached to the UCLA to ISI router accurately reflects UCLA's view of the MBone topology.

It is important to note that the data gathered by a RouteMonitor is strongly dependent on its location. If the UCLA to ISI link fails, the UCLA RouteMonitor will detect that all non-local routes in the MBone have become unreachable. But other portions of the MBone do not depend heavily on the UCLA to ISI link and they would see a far less dramatic

[1]DVMRP specifies both a topology learning protocol and a tree construction protocol. Although less than half of the routers in the MBone at the time used the DVMRP tree construction protocol, nearly all of them used DVMRP for topology learning. Unless specifically stated otherwise, the term DVMRP will refer only to the topology construction protocol.

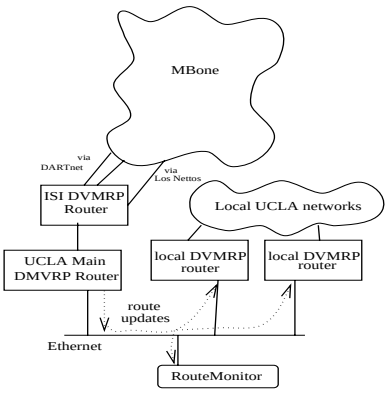


**Figure 1. Data Observation Point**

change. To another site, only the UCLA sources become unreachable and all other routes in the MBone would remain unchanged. We do not claim the data collected by the UCLA RouteMonitor represents the definitive view of the DVMRP infrastructure; instead we will show that by starting from the UCLA view we were able to detect faults that impacted virtually every portion of the infrastructure.

To understand the following data, it is important to note that the DVMRP topology learning algorithm does not depend on any dynamic factor such as network congestion or which hosts are actively involved in a multicast session. If no faults were occurring, the topology database would remain constant. Allowing for the occasional stop-failure of links or routers, one would expect that topology of the MBone changes very little, if at all. The DVMRP topology database observed by RouteMonitors, however, changed frequently.

### 3.1 DVMRP Hourly Data

The results displayed here were compiled from the RouteMonitor database and are intended to illustrate the scope and type of problems occurring when our project began.

Figure and 2 shows the number of total routes (solid line) and the number of stable routes (dotted line) observed each hour at the UCLA RouteMonitor during August of 1997. The number of routes is computed by counting how many distinct sources appear in route updates during an hour. A route to a source is said to be stable during the hour if the distance reported to that source did not change. On average routes for 5600 sources are seen during each hour and 3700 routes remain stable. Only 67 percent of the routes remain stable during an hour.

The large drops in figure 2 corresponds to fail-stop behavior by links or routers critical to UCLA. All but 1200 routes rely on an particular ISI link. (see figure 1). The

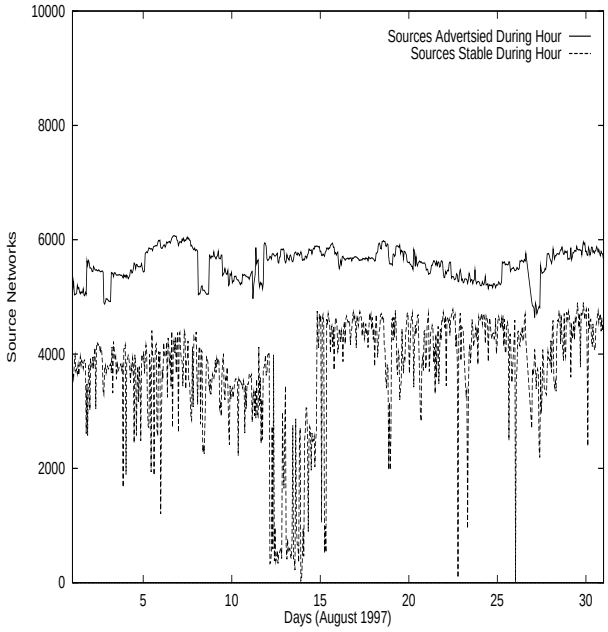


**Figure 2. August 1997**

large drops in the number of stable routes very likely correspond to failures of this link. These large drops only indicate that critical links near UCLA failed during that hour and do not reflect faults in the routing protocol itself. Fail-stop behavior by links and routers can only be eliminated by better maintenance of the underlying network hardware and software.

Even after considering the failures of critical links, the overall number of stable routes is considerably lower than anticipated. During any hour, some links and routers may fail. The failure of a few critical links or several less critical links can create a large number of route changes. But the level of instability suggested that other problems may be occurring. In section 3.2, we provide data that suggests many route changes are not due to the failure of links or routers.

### 3.2 Average Number of Flaps Per Route

As the previous section indicates, failure of a single critical link may cause many changes. The distance to a source may change a few times as news of the link failure spreads; but it should stabilize and remain constant (at least until another link fails). Figure 3 shows the number of changes that occurred per route during a representative one hour period. Link failures could explain the routes that change a few times during the hour, but link failures are unlikely to account for over 60 changes during a single hour. Figure 4 shows the average number of changes per hour, measured over a several month period. Note that a large number of

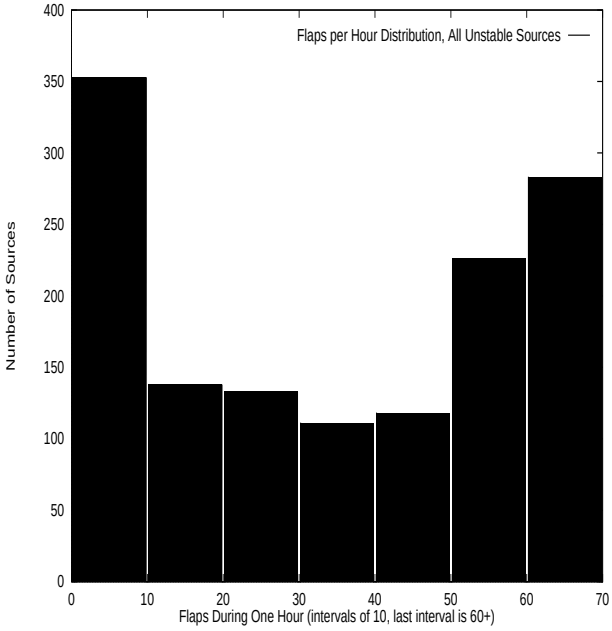


**Figure 3. Flaps during one hour (in August 1997, stable routes not shown)**

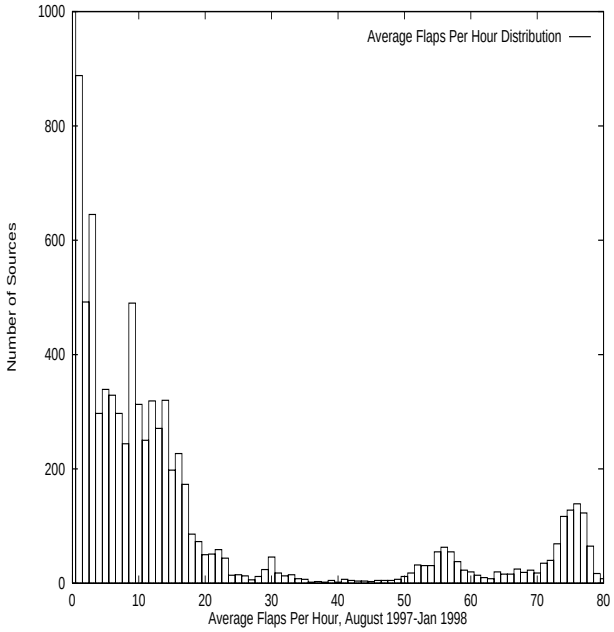


**Figure 4. Average Flaps per Hour (y-axis chopped at 1000)**

routes consistently change over 60 times an hour.

### 3.3 Number of DVMRP Routes

From the graph in Figure 2, one might infer there were approximately 6000 distinct sources in the DVMRP routing table. But RouteMonitor discovered this was not the case. While the number of distinct sources updated each hour consistently ranges between 5000 and 6000, a different set of sources was updated each hour. Updates for some sources ceased to appear while updates for previously unknown sources appeared. In fact, nearly 10,000 distinct sources appeared in at least one update during August 1997. Every hour, our picture of the MBone consisted of a *different* set of roughly 5600 sources and the true MBone could consist of nearly 10,000 sources. In January 1998, our route database had entries for over 18,000 different sources.

The graph in Figure 5 shows the number of sources advertised each hour and the number of sources advertised in August, 1997. The RouteMonitor detected that each hour some sources **appear** while other sources **vanish** and thus UCLA sees a different routing table each hour. A source **appeared** if at least one update for the source was received during the current hour and no updates for the source were received during the previous hour. Similarly a source **vanished** if no updates for the source were received during the current hour, but at least one update for the source was received during the previous hour.

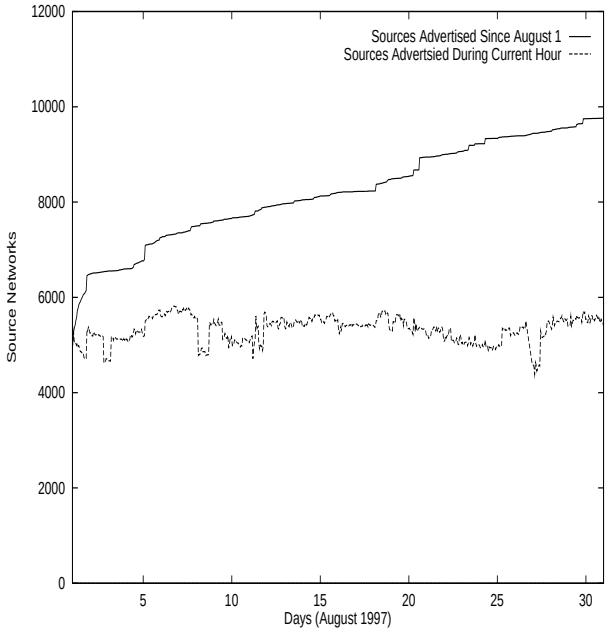


**Figure 5. Total sources seen in August 1997**

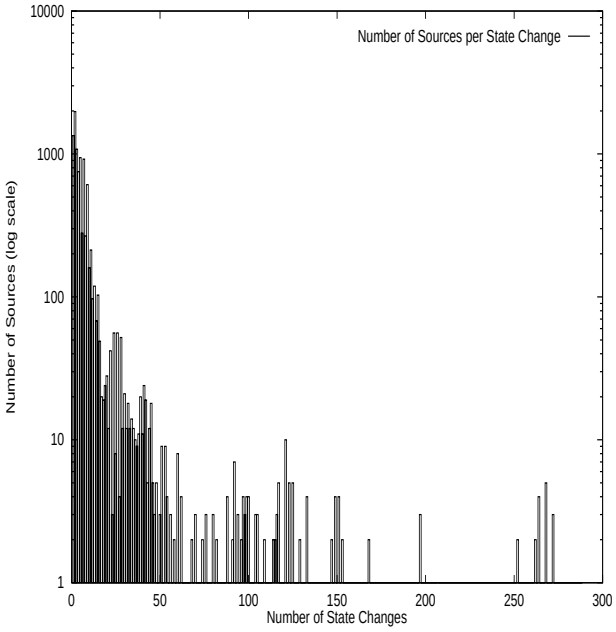


**Figure 6. State Changes per Route**

Figure 5 shows that new, previously unknown routes continuously appear. In fact, the collection of reachable sources in the MBone was a highly variable set; some sources appeared and other sources vanished in each data collection interval.

A route which vanishes or appears does not necessarily indicate a fault in the routing protocol. During the span of a month, some valid routes may temporarily vanish due to administrative work on their local router. If a critical link is down for an entire hour, this could also cause a valid route to vanish. This does not indicate a fault in the routing protocol. It simply represents the protocol correctly reacting to the fail-stop actions of a link or router. But again the large number of changes seemed to indicate some other factor was involved.

RouteMonitor tracked the frequency with which routes appeared and vanished and discovered a large number of routes vanish and reappear periodically. We say a route **changes state** every time it appears or vanishes. By definition, each route changed state at least once, the first time it appeared. Figure 6 shows the distribution of state changes observed during August 1997. The distribution shows that only 1346 sources appeared and then continued to appear in at least one update for each subsequent hour during the month.

### 3.4 Summary of the UCLA View

The view from the UCLA RouteMonitor provided one viewpoint of the DVMRP routing table. We were able to determine that the UCLA routers were behaving correctly, but frequent route changes were being learned from neighboring DVMRP sites. Based on the frequency and overall amount of route changes, it seemed unlikely that these changes were due to fail-stop behavior by links or routers. However, the UCLA view alone was not sufficient to identify the cause of the problems.

Despite the limitations of a single viewpoint, we were able to detect and correct some MBone configuration faults. For example, in late July the number of DVMRP routes suddenly rose by nearly 2000 routes. The RouteMonitor data alerted us to this change and allowed us to easily determine which routes suddenly appeared. The problem was then easily traced to a configuration error at a remote site. The site administrator was contacted and corrected the error. The number of DVMRP routes seen by the RouteMonitor quickly soared to over 7600 and then returned to 5600 a few hours later after the administrator corrected the error.

Note that this data is trivial to collect using RouteMonitor, but not easily obtained by other methods. Active tools can not easily capture a picture of the global routing table. By passively observing updates, RouteMonitor collected this data without adding any additional load to the network and without sending any SNMP style query messages to any router. The data did not rely on a polling system which could miss some changes. Every route change reported by the protocol was captured by RouteMonitor and the data is complete from the perspective of the DVMRP protocol. Both the global picture and the specifics of any individual source are easily obtained.

RouteMonitor had allowed us to collect large amount of data without adding any load to local routers. We were able to observe the actions of local routers and provide a real time, easy to view picture of the MBone. Based on this picture, we identified problem behaviors that had not been widely recognized. But based on our single point of observation, we could only determine that the problems were not local.

## 4 Identifying DVMRP Network Layer Faults

To further analyze the problem, we needed to obtain data from other portions of the MBone. While a single RouteMonitor is useful, our approach is based on a collection of RouteMonitors running at different points in the network. The UCLA RouteMonitor identified several sources which change frequently. We needed to see what was happening at these sites or along the path these sites. Since there is no central authority “in charge” of the MBone, there was no single contact to provide the necessary access. Instead, we had to try to determine who was responsible for the sites in question, contact them individually, and try to convince them to give us access to information from the site.

The UCLA RouteMonitor provided two important assets in our attempt to gather data from other sites. The first asset was that we could easily demonstrate the problem by simply pointing to the UCLA RouteMonitor web page. This allowed anyone to see the scope of the problem and helped motivate the interest of other sites. The second important asset was that we did not have to request access to routers at other sites. Instead, we were able to provide the RouteMonitor software for collecting data. A site could deploy the software on a host, filter the data based on a particular address, and simply send us a text file showing the updates exchanged. This allowed us access to the required information without requiring other sites to give us access to their systems.

We focused on the most frequently changing source addresses and used *mtrace*, the multicast traceroute program, to try and identify interesting points to deploy RouteMonitors. Most sites were receptive to running some RouteMonitor data collection. The data viewed from other sites confirmed the results seen at UCLA. These sites observed pathologies similar to those at UCLA and the sources seen as unstable at UCLA tended to be unstable as seen from other sites as well.

Using a system of multiple RouteMonitors allowed us to isolate problems to particular routers or links. RouteMonitor provides both high level protocol level information and a packet sniffer capability to view individual route updates exchanged over a link. Once the RouteMonitor data pinned down the location of a fault and provided us with both high level data and low level individual updates, we were able to identify the specific causes of most DVMRP instability.

### 4.1 Sending Bursts of Updates

When using DVMRP, a router must send an update for each source once a minute. At least one implementation sends the updates for all sources at once. On average, 5600 sources need to be updated. For each source the router sends the source address, netmask for the address, and the distance metric. The updates are sent in approximately 70 consecutive packets, each packet containing updates for about 80 sources. We call a router exhibiting this behavior a "bursty router".

Neighboring routers receive the burst of updates faster than they can be processed. This results in packet loss as queues overflow. RouteMonitor would also suffer this buffer overflow, but RouteMonitor counts the number of updates received each hour. Looking at the data, we were able to determine that the expected number of updates was not being received and the problem was traced to the buffer overflow. These bursts have been observed to overwhelm several different implementations, including the one that sends the bursts.

Route instability occurs when a router $R$'s best path to a source is via a bursty router, $B$. Router $R$ hears an update for the source from $B$. The route via $B$ is the shortest route so $R$ declares $B$ to be its previous hop toward the source. The bursty router continues to send updates for the source each minute, but many of these updates are dropped because of buffer overflow at $R$. If three consecutive updates from $B$ are dropped, $R$ believes the route has timed out and removes it from its routing table or begins using an alternate path. $B$ continues to send updates for the source and eventually some update is not dropped. $R$ then relearns the route via $B$ and the process repeats.

In one particularly bad case, a bursty router advertised routes to 256 local sources. At our UCLA RouteMonitor, each of these 256 routes changed an average of 68 times per hour. During one particularly bad hour, these routes changed 173 times. The burst problem alone could account for at most 60 changes per hour; the remainder are due to other behaviors that added to the initial instability. At no point during our study were any of these routes stable for an entire hour.

This problem is essentially a protocol implementation error. An implementation needs to send updates at a reasonable rate. The solution to this problem is simply to space out updates rather than send a large burst of updates. Most DVMRP routers distribute the updates evenly over the 1 minute period and do not suffer from the burst problem. We informed the router vendor of the problem and a configuration parameter to reduce the problem was added in more recent software releases.

### 4.2 Interactions with Unicast Routing

Some routers are used for both unicast and multicast routing. When using the DVMRP topology discovery protocol, these routes go into a separate table, and the router uses both tables to determine routes for multicast tree construction. However, the router can be configured to advertise sources from the unicast table to neighboring multicast routers. In certain topologies, the router becomes confused about how to reach these sources.

Consider the topology shown in Figure 7. The router $C$ learns of local network $S$ from its unicast routing table and is configured to advertise network $S$ into DVMRP. $C$ sends a multicast route update to $A$ and $B$, listing $C$'s distance to $S$ as 1. $A$ will select $C$ as its previous hop to $S$ and set its distance to $S$ as 2. When $B$ hears an update from $A$, it will prefer the route it heard from $A$ (cost 3) over the route it heard from $C$ over the backup link (cost 5). $B$ will then send an update to $C$ listing $B$'s new distance to $S$ as 3. $C$ should ignore this new route to $S$ since $C$ has a route of cost 1.

However, $C$'s route to $S$ came from its unicast table. $C$

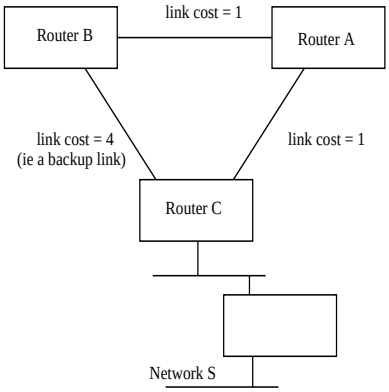


**Figure 7. Sample Topology for Section 4.2**

doesn't have a route for $S$ in its DVMRP routing table, so it accepts the cost 7 route to $S$ from $B$. This forms a loop where $B$ believes (correctly) its previous hop to $S$ is $A$, $A$ believes (correctly) its previous hop toward $S$ is $C$, but $C$ believes its previous hop toward $S$ is $B$. $C$ sends an update to $A$ informing its distance to $S$ is now 7 (via $B$). $A$ then informs $B$ that $A$'s distance to $S$ is now 8. $B$ informs $C$ the distance to $S$ is now 9. Gradually the distance to $S$ increases until the maximum allowable distance is reached, 31 in DVMRP. This behavior where routing updates flow around a loop and count up to the maximum metric is called "counting to infinity." Once the distance to $S$ exceeds 31, $C$ prefers its original unicast cost 1 route to $S$ and the process begins again.

This problem will only occur if $C$ is part of a topology where it may hear an update for a route it introduced and if $C$ is configured to prefer the multicast routes to its own unicast routes. In this particular implementation, this is the default configuration, because in simpler topologies this is the correct behavior.

In complex systems, where multiple routing protocols can interact, it is extremely difficult to make sure that all cases work correctly. The current state of the art is to make the configuration as flexible as possible, and expect that administrators will configure their systems properly. Since these systems are so complex, it's hard to even know what the interdependencies are, let alone document them. Unfortunately, this leads to situations like this one where an implicit configuration requirement causes problems for the network.

### 4.3 False Aggregation of Routes

Overlapping routes can sometimes cause confusion in another implementation. Suppose both a specific route 131.179.96/24 and more general route 131.179/16 are both being advertised. This implementation will accept the updates for both routes but advertise only the more general route, 131.179/16, to its downstream neighbors. The correct behavior would be to advertise both the 131.179/16 and the 131.179.96/24 routes to the downstream neighbors.

Instability can occur because such a buggy router applies the metric for each route to the more general route in the routing table. In most configurations, the more specific route will have a higher metric than the general route (e.g. the more specific route is injected by a site's internal router and the more general is injected by the border router). However, if the more specific route has a lower metric (e.g. because of "hole-punching" in a CIDR[11] aggregate), this behavior causes the apparent metric of the general route to vary. The router will accept the specific route with the low metric, update the general route's metric with it, and trigger an update of this new information to its peers. Then, in the next routing update, the router will accept the general route with the higher metric and update the general route's metric back to where it was. This information is triggered also. Then comes the more specific route in the update, and the process starts over again. This causes instability for the general route downstream of this router, and can cause abbreviated counting to infinity in cases where the metric difference is high.

### 4.4 RouteMonitor's Contribution

The RouteMonitor system allowed us to detect problems in the DVMRP infrastructure that had not been detected by other means. Figure 2 indicates that there was high level of overall instability and this instability had been occurring for at least several months. Figures 3 and 4 indicate some sources were suffering from severe routing problems. Prior to RouteMonitor, the general problems were not widely recognized and the specific faults had not been detected.

Higher layer protocols are designed to be robust. Legitimate route changes do occur and higher layer protocols should work despite these changes. Higher layer protocols ranging from TCP (unicast only) to audio tools have proven extremely robust in hiding underlying problems. But hiding instability has performance costs for the end user as well as the network. Routing protocol fault detection tools need to be as robust in finding errors as higher layers are at hiding them.

Without a problem report for a specific source, it is not surprising that active monitoring tools such as *mtrace* did not discover the problem. But it may seem surprising that passive monitoring tools such as SNMP did not find the problem. The difficulty lies in noticing the difference between legitimate change and problem behavior. Using traps or polling, SNMP can detect route changes or route timeouts. But routes can change or time out for a number of valid reasons such as the failure of a critical link. When a link does fail, the route may change several times before

converging on the new correct route. Thus route changes, even several route changes, would not necessarily indicate a problem. To detect the problem, one would have to observe that a route continuously changes over a sufficiently long time such as days or weeks.

This kind of long term observation costs the router resources. Considering that there are over 4000 active routes, long term observation of all routes may be tedious and unrealistic. Long term monitoring of some routes has a probability of catching the problem, but picking the proper subset to monitor is difficult, as is deciding what is normal and what is not.

Asking a faulty device whether or not it is faulty may not reveal any problems. For example, the bursty router from section 4.1 would simply report stable routes. Even if one suspected a problem with lost updates, the bursty router believes it is sending the correct number of updates and one would eventually be forced to observe the packets directly.

RouteMonitor provided a simple method to detect problems and added no cost to the routing protocol or the routers. RouteMonitor is not involved in forwarding packets. It runs on a host and its primary job is to collect and analyze data. For RouteMonitor, a natural question to ask is which routes change most frequently. In fact, RouteMonitor automatically posts a list of the current week's worst routes, the *frequent flappers*. The frequent flapper list is sorted by the average number changes per hour during that week. Based on this data, some suspect routes are immediately obvious. Closer analysis of the data, such as figures 3 and 4, clearly indicates problems are occurring.

### 4.5 Results Achieved with the DVMRP RouteMonitor

Most of the frequently changing routes could be traced to one of three problems described above. We contacted the vendors of each implementation mentioned and they produced fixes for the bugs. The fixes for sending bursts of updates (section 4.1) and false aggregation (section 4.3) have been introduced in newer releases of these implementations. The unicast/multicast route confusion (section 4.2) can be solved via the router's configuration file and details on how to do this were posted on the RouteMonitor web page.

RouteMonitor also sent weekly email message to the mailing lists for MBone router operators and its European equivalent. At the end of each week RouteMonitor determined which routes changed the most during the week. Only one route per autonomous system (AS) was included in the list. The email was sent to the mailing lists in order to inform sites of potential problems and help speed deployment of patches which could correct much of the instability in the network.

This approach was successful and the sites responsible for the worst problems deployed the patches. From 1997 to 1998, the average number of stable routes improved from 67% to 95% and no routes averaged over 60 changes per hour. Because of the declining role of DVMRP in the MBone, RouteMonitor data is no longer sent to the mailing lists. The RouteMonitor web site continues to provide hourly reports at http://ganef.cs.ucla.edu/~masseyd/Route.

## 5 Summary

Reliable systems for monitoring routing protocols are required to ensure reasonable performance and to guard against faults. Current systems are able to cope with only some potential faults. We have presented an effective approach to network monitoring and fault detection. Our system adds a collection of virtual routers called RouteMonitors into the network, which allowed us to monitor the network without adding load to the routing protocols or routers.

In the MBone, we discovered that a relatively small number of faults had been creating widespread problems. Some sites were unable to send data due to frequent route changes. All sites suffered from excessive DVMRP overhead at the routers. Existing tools and trouble reports by end users had not detected these problems. The DVMRP RouteMonitor system identified that these faults existed and pointed manual debugging efforts in the right direction.

The DVMRP RouteMonitor is of decreasing utility as the MBone moves towards using MBGP for topology discovery, but can still be useful in enterprise networks or intranets in which the use of DVMRP is still appropriate. The RouteMonitor software is freely available from http://ganef.cs.ucla.edu/~masseyd/Route and has been deployed by a number of sites.

### 5.1 Future Work

This work primarily discussed a DVMRP RouteMonitor, but the same principle applies to protocols other than DVMRP. An MBGP RouteMonitor is now available to study the behavior of the MBGP protocol, and an MSDP RouteMonitor is in the works. Protocols such as MBGP and BGP offer a greater potential for automatic fault detection. Route updates for these protocols include a path to the source instead of simply the distance reported by DVMRP. This provides RouteMonitor with more information to use in constructing views of the network and detecting routing protocol faults.

The results presented here combined automated RouteMonitor work with manual analysis of the data. More faulty behaviors could be automatically detected by having RouteMonitor check certain axioms or comparing the views from different sites. For example, the number of updates for a

source must be greater than the number of minutes over which the source has been observed, or a route should be present at all monitoring locations. We are investigating enhancements that allow for a higher degree of automated detection.

## References


[1] A. Adams, J. Mahdavi, M. Mathis, and V. Paxson. Creating a Scalable Architecture for Internet Measurement. In *Proceedings of INET 98*, July 1998.

[2] A. Ballardie. Core Based Trees (CBT version 2) Multicast Routing. RFC 2189, SRI Network Information Center, Sept. 1997.

[3] T. Bates, R. Chandra, D. Katz, and Y. Rekhter. Multiprotocol Extensions for BGP-4. RFC 2283, SRI Network Information Center, Feb. 1998.

[4] R. Caceres, N. Duffield, F. LoPresti, J. Horowitz, J. Kurose, D. Towsley, and V. Paxson. MINC: Multicast-based Inference of Network-internal Characteristics. http://gaia.cs.umass.edu/minc/.

[5] CAIDA. The coralreef software suite. http://www.caida.org/Tools/CoralReef/.

[6] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP). RFC 1157, SRI Network Information Center, May 1990.

[7] S. Deering. Host Extensions for IP Multicasting. RFC 1112, SRI Network Information Center, Aug. 1989.

[8] H. Eriksson. MBONE: the Multicast Backbone. *Communications of the ACM*, 37(8):54–60, Aug. 1994.

[9] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol specification. RFC 2362, SRI Network Information Center, June 1998.

[10] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[11] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. RFC 1519, SRI Network Information Center, Sept. 1993.

[12] C. Huitema. *Routing in the Internet*. Prentice-Hall, 1995.

[13] C. Labovitz, G. Malan, and F. Jahanian. Internet Routing Instability. In *Proceedings of ACM Sigcomm*, Sept. 1997.

[14] D. Makofske and K. Almeroth. The MHealth project. http://imj.ucsb.edu/mhealth/.

[15] G. Malan and F. Jahanian. An Extensible Probe Architecture for Network Protocol Performance Measurement. In *Proceedings of ACM Sigcomm*, Sept. 1998.

[16] NLANR. Measurement and opeartions analysis team. http://moat.nlanr.net/.

[17] A. Swan and D. Bacher. A tool for viewing RTCP feedback packets from an RTP session. ftp://mm-ftp.cs.berkeley.edu/pub/rtpmon/, 1997.

[18] K. Thompson, G. Miller, and R. Wilder. Wide-Area Internet Traffic Patterns and Characteristics. *IEEE Network*, Nov. 1997.

[19] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol. RFC 1075, SRI Network Information Center, Nov. 1988.

[20] L. Wei. MRM. IETF MBoneD mailing list; ftp://ns.uoregon.edu/mailing-lists/mboned.archive, 1997.