SLAC-PUB-5895
August 1992
(M)

Optimal Routing of IP Packets to Multi-Homed Servers[†]

Karl L. Swartz
Stanford Linear Accelerator Center
Stanford University, Stanford, California 94309

ABSTRACT

Multi-homing, or direct attachment to multiple networks, offers both performance and availability benefits for important servers on busy networks. Exploiting these benefits to their fullest requires a modicum of routing knowledge in the clients. Careful policy control must also be reflected in the routing used within the network to make best use of specialized and often scarce resources. While relatively straightforward in theory, this problem becomes much more difficult to solve in a real network containing often intractable implementations from a variety of vendors.

This paper presents an analysis of the problem and proposes a useful solution for a typical campus network. Application of this solution at the Stanford Linear Accelerator Center is studied and the problems and pitfalls encountered are discussed, as are the workarounds used to make the system work in the real world.

Introduction

In a simple network composed of a collection of machines connected to a single multi-access network, such as an Ethernet, routing of IP packets is simple since each host can talk to every other host on the network. Introduction of a second network, connected via a router, adds only slightly to complexity — any non-local hosts must be reached via the router. This can be trivially implemented by adding a *default* route on each host pointing to the router's adjacent port.

Introduction of the second network and the router does serve to segregate traffic and thus reduce network loads, but it also creates some new problems if hosts on both networks share one or more large servers. If the router goes down, one network loses access to the server. If the network to which the server is attached goes down, perhaps for maintenance, both networks lose access to the server. If the two networks serve two different user communities with very different work habits, this may make it difficult to find a time when maintenance can be done on the critical network, and perhaps costly as well if overtime is required.

Problems with NFS and routers

If the server is an NFS fileserver, another more subtle problem may appear. NFS uses 8 kilobytes as its default packet size, a value which parallels the largest possible block size in a BSD filesystem. Ethernet, on the other hand, has a Maximum Transfer Unit (MTU) of 1500 bytes. In order to pass an 8 KB packet over an Ethernet, IP must fragment this large packet into a number of smaller packets, which are re-assembled when they reach the destination. If the

[†]This work supported by the United States Department of Energy under contract number DE-AC03-76SF00515.

Presented at the USENIX Association's Sixth Large Installation System Adminstration Conference (LISA VI), Long Beach, California, October 19-23, 1992.

network is congested the sender may delay between each packet. Ideally the destination is eagerly awaiting the packets and will have no trouble accepting all of the fragments. An intermediate device such as the router, however, may not be able to accommodate all of the incoming traffic and may simply drop some of the fragments. The destination will eventually decide that some fragments are never going to arrive and will purge the entire packet. The NFS client software, with no knowledge of this fragmentation, simply sees a slow or unresponsive server and will reissue the request, at some point also issuing the infamous `NFS server not responding` diagnostic.[1]

One can certainly reduce the NFS packet size by adjusting the **rsize** and **wsize** parameters of the mount, but this could have a substantial performance impact of its own, and requires knowledge of the network topology by each client to decide whether or not the smaller values should be used. Recent versions of AMD [2] address the latter issue through the **wire** selector, though the other problems still remain.

Multi-homed servers

Instead of trying to work around the problems introduced by addition of the router, a better approach is to avoid them wherever possible by connecting the critical server to both networks. One could eliminate the router altogether and use the server as a router, and some server vendors suggest this. This is probably going a bit too far unless there is little traffic between the two networks. Even relatively inexpensive routers are better at this job than most fileservers, and fileservers burdened with many incidental tasks tend to become poor fileservers. Using a router in parallel with a multi-homed server, as in figure 1, provides the best service both for traffic between the networks and for access to the server.

Now the challenge becomes one of getting the hosts to talk to the server directly rather than via the router. If the host in figure 1 refers to the server using the IP address on network *B* (or for brevity of notation simply S_B), all is well. If, however, the host uses S_A, it will see that the address is not on its local network and will use the default route to choose the router as the gateway to the final destination, as illustrated in figure 2. Besides placing unnecessary traffic on network *A*, this affects availability by introducing dependencies on both router *R* and network *A*. It also raises the specter of packet fragmentation problems.

[1]A discussion of this problem is contained in [1]. "Appendix C: NFS Problem Diagnosis" of that document details how to determine whether or not dropped packets are a problem for a particular system.

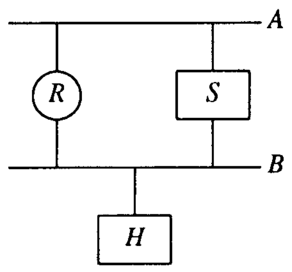


Figure 1. Simple network with multi-homed server

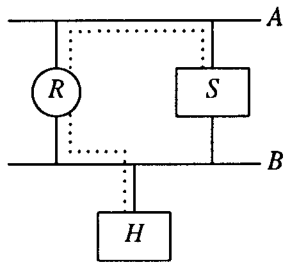


Figure 2. Path using default route to S_A

One could simply use different names for S_A and S_B, and this is what some vendors recommend [3]. Users probably won't remember the difference, however, even if they understand the network topology sufficiently. NFS configuration also becomes more complicated, which may be a substantial consideration for a site with many machines and servers.

Another approach is to use the same name but to somehow resolve it to the closest address for each machine. This host-specific name resolution can easily be accomplished via `/etc/hosts`, though this doesn't scale well to a large network. Sun's Network Information Service (NIS, formerly known as Yellow Pages) [4] is commonly used, for better or worse, to assist with maintenance of the hosts database, but it provides for only one address per host name across the entire network. The Domain Name Service (DNS) [5] provides for multiple addresses but no preference for one over another, though some implementations do provide a facility for sorting multiple addresses [6].

Host routes

With resolution of a single server name to the best address being eliminated as a viable option in many cases, the focus turns to what can be done if a host chooses the "wrong" address for the server. A host's routing table can certainly be more sophisticated than a simple default route for all non-adjacent addresses. In particular, a *host route* may specify a different route to one particular destination host than would be used for other hosts on the same network. Figure 3 illustrates the host's view of this.

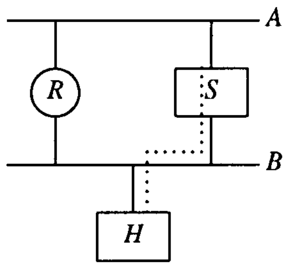


Figure 4. Path using host route to S_A

Here, S_B is the gateway to S_A, while R_B is used as the gateway to other hosts on network *A*. Thus packets sent by the host to the server's "wrong" address will still go directly to the server, as in figure 4. Some additional processing is required on the server but for any modern server this should be insignificant.

Dynamic routing

For the network presented thus far, static routing is adequate (if a bit tedious) to implement everywhere. For a real network with many servers and perhaps multiple routers, maintenance of the routing tables on a multitude of hosts using static routes would be prohibitive. Fortunately, dynamic routing protocols exist which permit network devices to discover routes on their own and which enable them to adapt to changes in network topology, planned or otherwise.

Three different types of participants in the routing process have been encountered in this discussion, each with different needs and capabilities. Starting with the simplest, they are

- A workstation or other host *H* with only a single interface. It silently listens to routing information on the network and updates its own routing tables as necessary. Having only one interface, it can perform no useful gatewaying and thus has no need to advertise any routing information to others.
- A server *S* which has multiple interfaces, i.e., it is multi-homed. Like a workstation, it listens to routing information on the network and updates its own routing tables. (It should also be able to choose the best interface for outgoing traffic when confronted with a choice.) In addition, it acts as a gateway for packets addressed to one of its interfaces that are received via a different interface, and it must advertise routing information (host routes) so other hosts know of this service. As a matter of policy it does not forward packets between two networks and thus should not advertise any network routes.
- A router *R* which gateways packets between networks on behalf of other hosts. Logically, it is nothing more than a host with two or more interfaces, but without the policy restriction against passing packets between networks. In practice it will probably be a specialized device and may thus be less flexible in how policies may be implemented. Since it is willing to pass packets beyond itself it must also advertise network routes, including ones learned from other gateways and not just the host routes which a server advertises.

Choosing a routing protocol

As with editors, shells, and mailers on UNIX, an abundance of routing protocols exist for IP. Fortunately, most can be eliminated as being irrelevant to the task of *interior routing* — routing within a single network. When selecting from the available interior routing protocols, the requirement that hosts be able to listen to the protocol is the most stringent. Nearly every TCP/IP package includes support for RIP, the Routing Information Protocol [7], often via the BSD **routed** utility. Relatively few support any other protocol.

Newer and perhaps better protocols do exist, and portable software is available to implement many of them. Cornell University's **gated** [8] is a notable example of such software. A better choice of protocol might be cisco's Gateway Discovery Protocol (GDP) [9] or the ICMP Router Discovery Protocol [10], for which several UNIX implementations are freely available in source form. At best, this approach suggests a lot of porting effort for a typical installation with a multi-vendor environment. More likely, there are at

```
$ netstat -r
Routing tables
Destination  Gateway    Flags  Refcnt  Use   Interface
localhost    localhost  UH     2       38    lo0
default      ROUTER-B   UG     0       3     le0
A            ROUTER-B   UG     3       296   le0
SERVER-A     SERVER-B   UGH    8       758   le0
B            HOST       U      22      516   le0
```

Figure 3. Routing table with host route to S_A

least some devices which are "black boxes" and thus not amenable to this solution. FastPath gateways for AppleTalk, which support only RIP [11] and which cannot be enhanced with new protocols by the user, are a good example.

For better or worse, RIP appears to be the only reasonable choice available at this time if one wishes to dynamically communicate routing information to as wide a variety of hosts as possible. (Even this doesn't cover some hosts, which only support static routes to a default gateway, such as DOS machines using FTP Software's PC/TCP [12].) The question, then, is whether or not RIP is adequate for the job.

Suitability of RIP

A RIP packet is broadcast on each attached network by a router or other device which has routes to advertise. For each route, the packet includes a destination, which may be a host, a network, or a special value for default routes, and a *metric*, which indicates the quality of the route. The metric is an integer, customarily representing the number of *hops* or intermediary nodes required to reach the destination. For the network in figure 1, R broadcasts on network B a packet advertising a route to network A with a metric of 1. If a host sees a route with a lower metric than one it is already using, it switches to the new route. In addition, a host route to a destination on a given network will be replaced, or *subsumed*, by a network route to the destination's network which has an equal or lower metric.

Nothing in RIP requires that the metric be a hop count, however. The only restriction is that values greater than or equal to 16 are considered to be *unreachable* — a RIP metric of 16 is commonly discussed as being "infinite." This is useful, since a host route to S_A via S_B with a metric of 1 would be subsumed by a network route to A via R_B with a metric of 1. If the network route instead has a metric of 2, the subsumption rule does not come into play and the host route is used, which is exactly what is desired.

It is evident that RIP is capable of conveying the necessary routing information. The remaining problem is whether or not the servers and routers can be made to generate the desired RIP advertisements. A server should advertise host routes to each of its interfaces with a metric of 1; it should not advertise any other routes since routers will handle traffic between networks. A router must simply *inflate* its metrics to 2 instead of the usual 1.

Policy control in RIP implementations

The BSD **routed** utility doesn't offer much in the way of control by external policies. In particular, one can keep it from generating any advertisements, but there is no way to have it generate host routes to its own interfaces while suppressing other advertisements, nor is there a way to inflate metrics.

Cornell's **gated** offers a great deal of tailorability in addition to its support for a number of routing protocols. A **gated** configuration which will implement the desired policy for servers is illustrated in figure 5.

```
#
# Enable RIP
#
rip yes;

#
# Static routes to our own interfaces -- handles for advertisements
#
static {
    host S_A gateway 127.0.0.1 noinstall;
    host S_B gateway 127.0.0.1 noinstall;
};

#
# Advertise only host routes to our various addresses
#
export proto rip {
    proto rip restrict;
    proto static {
        all metric 1;
    };
};
```

Figure 5. gated configuration for server S

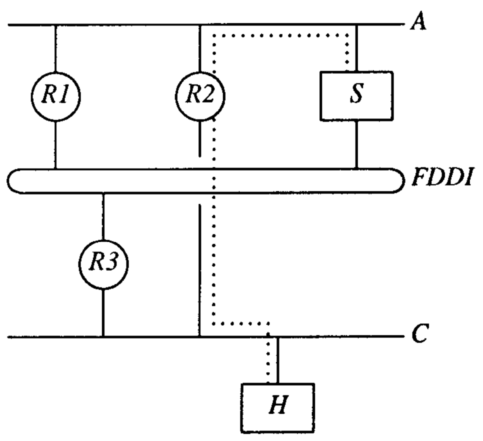
The first section in this configuration enables RIP. The second declares host routes to each of the server's addresses, which are not installed in the kernel routing table but which provide the necessary host routes for the RIP advertisements. The third section restricts RIP from advertising routes learned from other servers and routers, and causes the static host routes to be advertised with a metric of 1.

Few vendors distribute **gated** but, fortunately, porting is not as big a concern as for most hosts, since servers typically come from few vendors, compared to the multitude of sources for other hosts on the network. The sources from Cornell readily build on a variety of common UNIX platforms, and porting to VAX/VMS is not an issue since TGV provides **gated** with their MultiNet TCP/IP software. Porting to other non-UNIX platforms would probably be more difficult.[2]

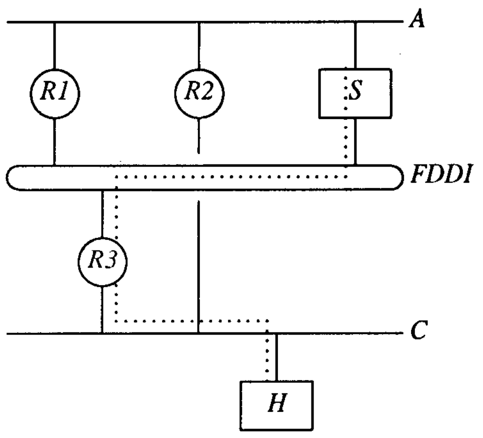
The necessary configuration for routers is simpler, yet much more difficult to generalize, since there is no common ground between vendors for configuration options. All of SLAC's routers are from cisco Systems. Because ciscos are quite popular and constitute the majority of the author's experience, only the cisco configuration will be considered. Fortunately, the necessary metric inflation is easily accomplished using the configuration fragment in figure 6. The key here is the **offset-list** command, which causes 1 to be added to the metric of advertised routes to destinations matched by access list 42, which simply matches all destinations.

One difficulty is that cisco routers do not currently understand host routes. They will be implemented in the upcoming 9.1 release of the cisco software. Until then, the impact is not too great since traffic that does not require a router will not be affected. Only packets which must already pass through one router run the risk of passing through more routers than necessary, and many of the benefits of a direct path disappear at the first router hop. Reliability should be handled for the most part by redundant router paths, which are needed to provide reliable service for other traffic anyway.

[2]Some creativity might be required to implement the necessary policies on peculiar servers. For particularly intractable systems, a hacked version of **routed** might prove more fruitful than trying to port **gated**. The author has so far successfully avoided the chore of porting **gated** to VM/CMS, though this necessity looms ominously.



7a. route with single Ethernet hop



7b. FDDI plus host route (preferred)

Figure 7. Alternate routes from H to S_A

Scalability of RIP solution

With the addition of another network and another router, the metric for the route to the distant network increases to 4, with each of the routers making a contribution of 2. As the network grows larger, with an increasing number of router hops between the most distant portions of the network, the "infinite"

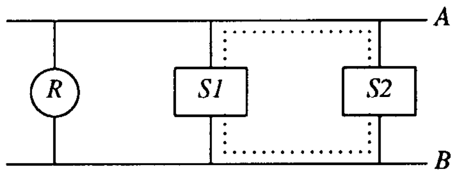
```
! metric inflation -- match all addresses
access-list 42 permit 0.0.0.0 255.255.255.255

! configure RIP routing process
router rip
offset-list 42 out 1
```

Figure 6. Configuration fragment for cisco router

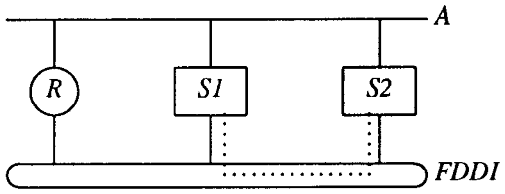
metric of 16 will be approached. With a metric value of 2 for each router this implies a maximum path or *network diameter* of seven routers. This is probably beyond the reasonable limit for RIP due to inherent limitations in the protocol.

Introduction of another type of network adds a new level of complexity to the problem. So far all networks have been considered to be equal. If one of the networks is an FDDI ring, while the others are Ethernets, preference should almost surely be given to the faster and higher-bandwidth FDDI network, as depicted in figure 7. RIP is still able to describe this topology, though the limits of scalability are much closer. The optimal route, in figure 7b, has a metric of 3 — 1 for the host route from S_{FDDI} to S_A and another 2 added by the traversal of router $R3$. Therefore the Ethernet hop via $R2$ must have a metric of at least 4. This brings the limit on network diameter to only three routers, or four if one pair is connected via the FDDI network.

Routes between multi-homed servers that share two common networks are also a concern. In figure 8a, either of the available routes is as good as the other, but if one of the networks is superior to the other, as in figure 8b, it should be the preferred path between the servers.



8a. either route may be chosen



8b. best route is via *FDDI*

Figure 8. Network preference amongst host routes

While the theory of using RIP to support both multi-homed hosts and a heterogeneous collection of networks may be interesting, it clearly shows signs of pushing the abilities of the protocol to its limits. Moreover, when one moves from theory to implementation, the problem becomes intransigent. The most

metric	destination	via	description
1	server	FDDI	host route
2	server	Ethernet	server advertises higher metric
3	—	—	only seen within server
4	network	(any)	single hop to network
	default	(any)	from firewall router
5	—	—	only seen within server
6	default	(any)	from router on FDDI
7	default	Ethernet	from router **not** on FDDI

9a. metrics as seen by host or router

metric	destination	via	description
1	server	FDDI	another server on FDDI
2	—	—	never seen by server
3	server	Ethernet	biased by receiving server
4	network	FDDI	single hop to network
	default	FDDI	from firewall router
5	network	Ethernet	single hop (biased by server)
	default	Ethernet	from firewall router (biased)
6	default	FDDI	from router on FDDI
7	default	Ethernet	from router on FDDI (biased)
8	default	Ethernet	from router **not** on FDDI (biased)

9b. metrics as seen by server

Figure 9. Interpretations of RIP metrics

apparent obstacle is that cisco does not provide any means of inflating the metrics of some routes to 2 while others are inflated to 4. This alone eliminates the possibility of solving the problem with only RIP.

Beyond RIP

The unique ability of RIP to convey routing information to a wide variety of hosts compels its use with the rank and file hosts. Its inadequacy for heterogeneous networks suggests that an ideal solution would use a more sophisticated protocol between routers and perhaps those servers attached to different sorts of networks, while retaining RIP for the low-end routing. With routers using RIP only for advertisements, and not for communicating with each other, the risk of routing loops that's usually associated with running multiple routing protocols is avoided. Translation of metrics from another protocol into RIP metrics is a problem, however.

The solution currently being pursued at SLAC is to use IGRP between the routers, with each router using RIP only to advertise a default route and routes to attached networks. The metrics of the default routes are tuned for each router to reflect the static distance from the firewall router, which attaches the SLAC network to the Internet. Multi-homed servers use RIP, with a bias against Ethernet routes which is explained below. A topic for future study is whether the servers should participate in the same routing protocol as the routers, which would force a move from IGRP to something else, probably OSPF or perhaps IS-IS.

The current solution still produces optimal routing locally, with packets from a host to a server on the same network going directly to the server and packets going to a network only one hop away going via the single-hop router. For destinations which are at least two hops away, a suboptimal route may be chosen in some circumstances. This does not seem to be too high a cost when set against the maintenance and reliability benefits of dynamic routing, and the route used is never worse than the routing produced by the static default route scheme which is being replaced.

While IGRP provides sufficient information to allow the routers to prefer FDDI routes, the multihomed servers rely on inflation of RIP metrics to implement this preference. Host routes advertised to Ethernets are advertised with a inflated metric of 2 instead of 1, so routers or other servers which see both advertisements will choose the FDDI route because of the lower metric. In addition, servers add 1 to the metrics of all routes learned via Ethernets. This second addition or *biasing* is redundant for routes to other servers (which will end up with a metric of 3), but is required because the routers aren't flexible enough to inflate their own Ethernet advertisements.[3] (Figure 9 summarizes the interpretations of the metrics as seen from the differing viewpoints of servers and other devices.) With the addition of an FDDI connection, the first two sections of the **gated** configuration from figure 5 are modified as illustrated in figure 10.

With a metric of 3 representing a host route to a server via an Ethernet, routers use 4 as the metric for advertising routes to their attached networks. The firewall router also advertises a default route with a metric of 4. When fully implemented, the FDDI

[3]The routers don't listen to host routes, either, so the inflated metric of 2 applied by servers to host routes they advertise on Ethernets isn't yet used. The inflation is implemented on the servers now so the necessary information is available when host routes become supported by the routers.

```
#
# Enable RIP
#
rip yes {
    interface en0 metricin 2 metricout 1;    # bias against non-FDDI
    interface en1 metricin 2 metricout 1;    # bias against non-FDDI
};

#
# Static routes to our own interfaces -- handles for advertisements
#
static {
    host S_A gateway 127.0.0.1 noinstall;
    host S_B gateway 127.0.0.1 noinstall;
    host S_FDDI gateway 127.0.0.1 noinstall;
};
```

Figure 10. gated configuration modifications for server on FDDI network will be the primary backbone of the SLAC network, and most Ethernets will only be a single router hop away from the FDDI, so routers attached to the FDDI advertise a default route with a metric of 6, metric 5 representing a (biased) Ethernet route to the firewall router. Other routers advertise a default route with even higher metrics. The relevant portions of the configuration for a typical router on the FDDI are shown in figure 11.

This approach assumes that moving toward the FDDI network is good if the destination is not nearby, which may not always be the best choice. It also requires individual tuning of each router, though only in the metric for the default route and according to relatively straightforward rules. It thus is less general than one would wish for, though it seems to be approaching the best that can be accomplished within the myriad constraints, and without resorting to Byzantine configurations which aren't easily demonstrated to be correct.

Conclusions

Multi-homing of important servers, in concert with intelligent routing by hosts, provides a useful tool for improving performance and availability when client hosts exist on multiple networks. In an installation with hosts from many different vendors, the only protocol reasonable for distribution of routing information to the hosts is likely to be RIP. Despite limitations of both the protocol and the implementations, RIP can, with reasonable effort, provide optimal routing even in the presence of multi-homed servers so long as all networks use the same or comparable technologies. Introduction of a much faster or higher bandwidth network complicates the problem beyond the capabilities of RIP, though RIP can still convey essential routing information to hosts and allow them to route packets optimally to nearby destinations.

References

1. Stern, Hal. *Managing NFS and NIS.* O'Reilly & Associates, Inc., Sebastopol, California, 1991.
2. Pendry, Jan-Simon and Williams, Nick. *AMD — The 4.4 BSD Automounter Reference Manual.* Imperial College, London, March 1991.
3. *System and Network Administration,* pp. 382-384. Sun Microsystems, Mountain View, California, March 1990.
4. "The Network Information Service," in *System and Network Administration,* pp. 469-512. Sun Microsystems, Mountain View, California, March 1990.
5. Mockapetris, P. *Domain Names — Implementation and Specification (RFC 1035).* USC/Information Sciences Institute, Los Angeles, California, November 1987.
6. *MultiNet System Administrators' Guide,* p. 23-6. TGV, Inc., Santa Cruz, California, May 1991.
7. Hedrick, Charles. *Routing Information Protocol (RFC 1058).* Rutgers University, New Brunswick, New Jersey, June 1988.
8. Information Technologies/Network Resources, Cornell University. "GateDaemon (Gated)" slides. Cornell University, Ithaca, New York, October 17, 1991.
9. *Router Products Configuration and Reference (vol. 2),* pp. 14-71 — 14-73. Cisco Systems, Inc., Menlo Park, California, April 1992.
10. Deering, S. (ed.). *ICMP Router Discovery Messages (RFC 1256).* Xerox PARC, Palo Alto, California, September 1991.
11. *Shiva FastPath 5 Installation Manual,* p. 30. Shiva Corporation, Cambridge, Massachusetts, June 1991.

```
! configure RIP routing process (default is metric 6)
router rip
distribute-list 2 in
default-metric 3
network SU-SLAC
offset-list 1 out 3

! configure IGRP routing process
router igrp 4
network SU-SLAC

! metric inflation -- match all addresses
access-list 1 permit 0.0.0.0 255.255.255.255

! RIP redistribution -- only default route
access-list 2 permit 0.0.0.0
access-list 2 deny   0.0.0.0 255.255.255.255
```

Figure 11. Routing configuration for typical router on FDDI

12. *PC/TCP Installation Guide*, p. 3-48. FTP Software, Inc., Wakefield, Massachusetts, January, 1991.

Trademarks

The following are trademarks and/or registered trademarks of their respective owners:

UNIX — AT&T
cisco — cisco Systems, Inc.
VAX, VMS — Digital Equipment Corp.
PC/TCP, FTP Software — FTP Software, Inc.
PC-DOS, DOS, VM, CMS — IBM Corp.
MS-DOS — Microsoft Corp.
FastPath — Shiva Corp.
Sun, NFS — Sun Microsystems, Inc.
MultiNet — TGV, Inc.
Ethernet — Xerox Corp.

Acknowledgements

Many people provided a variety of assistance in this effort, including Mark Barnett, George Berg, Chuck Boeheim, Bob Cook, Les Cottrell, Krissie Griffiths, John Halperin, Fred Hooker, Terry Hung, Tony Li, Jo Ann Malina, Greg Mushial, Don Pelton, Tim Streater, Mike Sullenberger, and to others whose contribution is not diminished by my failure to remember them. My gratitude goes to all of them. Thanks, too, to Alexander, for his patient tolerance, though I could have managed without the skunk.

Author Information

Karl Swartz is a member of the Systems and Networking Group within SLAC Computing Services at the Stanford Linear Accelerator Center, where he leads the UNIX support efforts. Prior to joining SLAC, he worked at the Los Alamos National Laboratory on computer security and nuclear materials accounting, and in Pittsburgh at Formtek, a start-up that is now a subsidiary of Lockheed, on vector and raster CAD systems. He attended the University of Oregon where he studied computer science and economics. Karl instructs at high-speed driving schools and enjoys good food and good beer (though not while driving) and long hikes with his Golden Retriever. Reach him electronically at kls@unixhub.slac.stanford.edu.