M.I.T. LABORATORY FOR COMPUTER SCIENCE

Computer Systems and Communications

February 24, 1981

Request for Comments No. 204

The Design of a Routing Service for Campus-wide Internet Transport

Vineet Singh

Attached is my recently accepted Master's thesis proposal.

Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science

Proposal for Thesis Research in Partial Fulfillment

of the Requirement for the degree of

Master of Science

Title: The design of a Routing Service for campus-wide internet transport

Submitted by: Vineet Singh
Ashdown House, # 611D
305, Memorial Drive
Cambridge, Mass. 02139

Signature of Author

Date of Submission: February 23, 1981

Expected date of Completion: June, 1981

Brief Statement of the Problem:

A campus environment requires many subnetworks connected by gateways, and it has a relatively loose administration. Modularization of network implementation is important in this environment if we want to make the best use of ever improving technologies and protocols. One of the implications of this is that we now need a network service that deals exclusively with routing information i.e. a Routing Service. The problem is to design a Routing Service that is best suited for the campus environment.

Supervision Agreement:

The program outlined in this proposal is adequate for a Master's Thesis. The supplies and facilities required are available, and I am willing to supervise the research and evaluate the thesis report.

Jerome H. Saltzer, Professor of Computer Science and Engineering

1. Introduction

I propose to design a Routing Service for the campus environment as a thesis toward the degree of Master of Science in Computer Science. This work will be performed under the supervision of Professor Jerome H. Saltzer at the Laboratory of Computer Science at MIT.

1.1 The campus environment

The campus environment has been characterized in considerable detail in [1]. Perhaps the single most important property of a campus-wide network is that it is limited to a single political and administrative boundary and, therefore, permits installation of low-cost, high-bandwidth technology. The number of data nodes that need interconnection is in the thousands and, therefore, local network interconnection strategies like [8],[9],[10],[11],[12], or [13] are not feasible solutions. Also, a campus wide network is expected to have a relatively loose administration and it will be administered only partly with central planning. A campus-wide network can also be expected to have a multiplicity of protocols. In fact, the diversity of technologies available in the computer world today and the variety of interests likely in a campus community will be inevitably reflected in a campus environment. Therefore, it is imperative that any interconnection strategies for the campus environment should be designed to work with all kinds of technologies for network hardware, protocols etc. and, in fact, the strategies should ideally evolve with changing technologies. This calls for a modular approach to the whole problem so that changes in implementation, in response to changing technologies, remain local as far as possible. One important consequence of this line of thinking is that routing strategies should deal only with routing and they should require minimum assumptions about other network functions and in turn these routing strategies should impose as few constraints as possible on the rest of the system.

1.2 Source routing for campus-wide internet transport

Source routing has been proposed in [1] as the mechanism to support internet addressing. The main advantage of Source Routing seems to be the modularization of network function that it provides--target identification and routing can now be separately done. There are other advantages to using a Source Routing mechanism. Gateways are much simpler because routing and target identification decisions are no longer made there. Simpler gateways lead to faster routing and this in turn leads to a better utilization of the available bandwidth. A precise control over routes can now be achieved and this directly aids trouble location, policy implementation, class-of-service implementation, and FIFO implementation. Source Routing allows the coexistence of several experimental routing policies without a major rehaul of network implementation. The problem now is that for any node to be able to send a packet over the network, the node has to somehow deduce the complete route that the packet must take to reach its destination. This either calls for extremely sophisticated nodes or a sophisticated "Routing Service". Clearly, it is not feasible to ask each node to undertake the onerous task of finding out routes on its own. Using a network service for routing to be shared by each node in the network brings the benefit of economy of scale. The task of this Routing Service would be to maintain current information about network topology and perhaps class-of-service information about various subnetworks and gateways and to use this information to act as an identity resolver and a routing information dispenser. What should be the key requirements of such a service? What are the major problems that will have to be solved in order to design such a service? These questions are discussed in the following section.

2. A closer look at the problem

This section looks at the problem in more detail. It describes what exactly a Routing Service should do and the key requirements along with a discussion of the major problems.

2.1 What exactly should a Routing Service do?

It is important to realize that services, nodes, ports, and paths (or routes) are four distinct entities in a network—the service is the resource that we are looking for, nodes are the places (or pieces of hardware) that a service may reside on (there may be zero or more nodes associated with a service and these may change with time), ports are the connection points in the network (there may be zero or more for each node in the network), and a path is how to get from one port in the network to another (there may be zero or more paths possible for each pair of ports). A name service is expected to map the name of a service into the nodes where the service resides. The Routing Service accepts a query from a port about a destination node and returns a path from the port (that made the enquiry) to one of the ports to which the destination node is connected. As part of its job the Routing Service may first map the destination node into the ports that it is connected to and then find a path from the port that originated the enquiry to one of the ports that the destination node was connected to. The criterion for choosing one path out of the several possible is a design decision and an important one at that.

Looking back, one can see that the problem is really one of keeping tabs on bindings that may change with network operation and finding them when the need arises—firstly the nodes that a service is bound to, secondly the ports that a node is bound to, and thirdly the paths that a port is bound to. The Name Server takes care of binding services to nodes and it is left to the Route Server to take care of the remaining bindings. Since the Route Server will return only one route and several may be possible, it is necessary to decide what mechanism the Routing Service should use to select one out of the many possible routes that the destination node may be bound to with respect to an origin port.

2.2 The key requirements of the design

a) We are working in a distributed environment:

All nodes in the network are pretty much autonomous and independent entities and they communicate with each other by sending messages to each other over the network to which they are connected. This structure has to be reflected in the design of the Routing Service. Although, a campus wide network is expected to have a lot of inexpensive bandwidth, sending messages over the network can still be quite expensive in terms of the amount of time taken for a message to travel from one end of the network to another. The number of messages that a Routing Service has to send and receive for it to function properly should be kept low especially when this directly affects the time that it takes to respond to users.

Although, local net routing and long-haul net routing have also to operate in a distributed environment, significant design differences can arise because bandwidth in the campus environment is a more scarce commodity than in the local net case and a less scarce commodity than in the long-haul net case.

b) The Routing Service should be reliable:

This requirement might be partially met by keeping the Routing Service as simple as possible. Therefore, in the design of the Service an effort will be made to keep the frills out. If at all any frills are retained it will be those that will be extremely hard to incorporate in the design once the Routing Service is implemented i.e. those frills whose design would require an effort quite orthogonal to the current design effort are most likely to be discarded. Simplicity in the design of the Routing Service is also likely to effect an improvement in maintenance cost, recovery time, trouble location etc..

Reliability of the Routing Service also means that it should face up to any changes in the network topology or its connectivity. Reliability is a necessary requirement even in the local net and long haul case but the kinds of changes possible and their frequency is different in the campus environment. What kinds of changes are possible in the campus environment in any case?

1. A gateway, subnetwork, or node breaks down or comes up again.
2. A new node is installed.
3. A new gateway or subnetwork is installed.

Therefore, routes should not be established only once at the time of network installation but rather the Routing Service should be able to update routes based on new information. However, it is entirely plausible that some changes may be such rare events that it might be wiser to just restart the Routing Service instead of building into it the ability to respond sensibly when such a change does occur. For example, a new gateway or subnetwork is not likely to be installed every day and, therefore, it is is not at all necessary for the Routing Service to be incrementally responsive to such changes.

c) The Routing Service should be reasonably fast:

It was mentioned in 1.2 that one the advantages of Source Routing is that it makes gateways almost trivially simple and that this might help to make good use of the cheap and abundant bandwidth available. If we now had an inordinately slow Routing Service, it would wipe out all the advantages of trying to speed up routing through gateways. In fact, this brings up another point. In the design process, we will have the option at several stages to make the Routing Service vary in sophistication along various axes e.g. the information gathered by the Routing Service may range from just topology or connectivity information to detailed class of service information about the traffic, gateways, subnets etc. Another example is the amount of computation required to compute good, better, or best routes. In all these cases it should be remembered that the philosophy behind a campus environment and a long haul environment (like the ARPANET) is most drastically affected by the fact that the first has a lot of bandwidth to throw around and in the second case bandwidth is a critical resource. Therefore, there is no need to squeeze out the last ounce of bandwidth in the campus environment. The Routing Service can afford to be sub-optimal in conserving bandwidth if in the process we have bought ourselves simplicity, or we have decreased the time that it takes the Routing Service to process queries. To sum up this requirement, the Routing Service should be designed so as not to be a bottleneck in the campus environment.

d) The Routing Service should require minimal support from the rest of the system:

This requirement is mainly to ensure that we actually use the opportunity to modularize the routing function. As I mentioned in 1.2, target identification and routing can be separately performed in the campus environment. However, we have to ensure that no built-in dependencies creep into the design of the Routing Service. Moreover, trouble location and recovery are facilitated by keeping the dependencies low. Distributed systems, in general, have a potential for being more reliable than other systems based on central processors. Making the Routing Service self-supportive to as great an extent as possible will go a long way towards making the Service robust and modular.

e) The Routing Service should scale gracefully for larger networks:

A serious attempt should be made to provide scaleability in performance (in terms of response time, reliability etc.) for the Routing Service.

It is claimed that distributed systems are intrinsically more reliable than centralized systems. This claim is only justified if the system is designed to exploit the existing potential. If, for example, the system is designed to scale gracefully, redundancy can be used to make the system more reliable.

It is entirely plausible that the campus-wide network may grow much too large for one Routing Service to handle efficiently. One approach to the problem is to partition the network into smaller units which single Routing Servers can handle. There may be other good reasons, in fact, for the network to be partitioned into several smaller units. Much like telephone zones, it is likely that there will be zones consisting of several adjacent subnets where most of the traffic originating in those zones will be directed to nodes within the respective zones. It is quite wasteful in this situation to force the Routing Server in one such zone to maintain information on all the nodes in the network. It is also not a good idea to compute routes to all the nodes in the network if 98% (say) of the routes to nodes outside a zone are not used at all. There is another scenario in which it makes good sense to partition the network into smaller units. Let us consider the case of two large campus-wide networks -- one in M.I.T. and another in Harvard -- connected together by exactly one gateway. It clearly does not make sense to require the Routing Service in M.I.T.'s campus-wide network to know about the Harvard network in any intricate detail if all the messages from the MIT net are going to go through a single gateway connecting the two nets anyway. Moreover, the Harvard administration may not want outsiders to know about the innards of their network. Now that we have built up a good case for partitioning the network, how can we actually make the system tick with a different Routing Server for each part?

Figure 1 shows a network partitioned into ten different parts. Each part consists of a number of subnetworks and different parts are connected by any number of gateways. It is useful to consider the general case of a network in which some part is isolated from the rest of the system. The network may be designed to be completely connected but failures may cause parts to be isolated temorarily. Let us also assume that each part is administered separately by a Routing Server. Therefore, the Routing Server for part 5, for example, will only calculate routes from nodes within 5 to any other node in 5. Now, if a node (call it 'abcd') within 5 wishes to communicate with another node (call it 'efgh') in part 7 and asks its Routing Server to find the route to 'efgh', then Routing Server 5 will have to do something like the following to find such a route. Route Server 5 will send out requests to adjacent Route Servers i.e. Route Servers 1,2,3,4,6,9, and 10 to help it find the route to a node called 'efgh'. Each adjacent route server will then look at the request and find that 'efgh' is not in its part and then each Route Server will send further requests on behalf of Route Server 5 to help find a route from part 5 to 'efgh'. One such request (let's say the one from 6) will eventually reach Route Server 7 which will eventually notice that 'efgh' belongs to its jurisdiction. Route Server 7 will send some information to 6 which will itself add some more routing information to the previous information and finally send all this information to Route Server 5. Route Server 5 will then finally put together a route from 'abcd' to 'efgh' and send this information to 'abcd'. The problem is to coordinate this whole process so that it works in all situations.

f) The Routing Server should maintain a good user interface:

As far as possible, the Routing Service should pamper the user. Let me give an example. Let's say that the Routing Service maintains class-of-service information about subnets and gateways. The user should be able to inspect the class-of-service information about the gateways and subnets on any route. If, for example, the user wanted to get a route which did not pass through a certain gateway because its class-of-service information did not meet the user's requirement, it should be possible to do so.

However, having a good user interface will only be a secondary objective. This requirement will lose out if it conflicts with the requirement of reliability or of the Routing Service being fast etc.

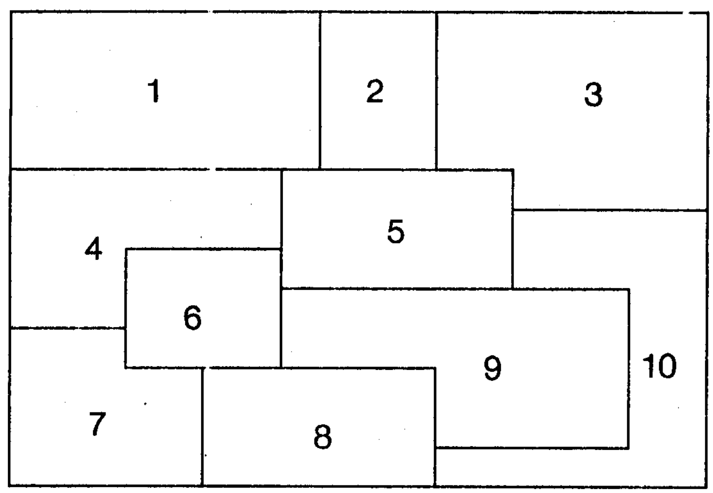

Figure 1

The campus-wide network partitioned into ten parts

3. Schedule of Tasks

Assuming a nominal 360 hours for the research and preparation of the thesis, the division of time for the various sections is outlined as follows: approximately one third of the time (120 hours) was spent during the fall term of 1980 in researching the work done in routing by other people, and coming up with a rough outline of the proposed design. Approximately one third of the time (120 hours) will be spent in further refining these ideas towards a final design. This work should be done by the beginning of April. The last third (120 hours again) will be spent writing the thesis and getting it into a final form. The thesis should be complete by June 1981.

4. Bibliography

[1] Saltzer, J.H., "Source Routing for Campus-Wide Internet Transport," Paper presented at IFIP Working group 6.4 Workshop on Local Area Networks in Zurich (August 27-29, 1980), proceedings to be published.

[2] Svobodova, L., Liskov, B., and Clark, D.D., "Distributed Computer Systems: Structure and Semantics," Mass. Inst. of Tech., Lab. for Comp. Sc., Technical Report, TR-215 (March, 1979).

[3] McQuillan, J.M., Richer, I., and Rosen, E.C., "The New Routing Algorithm for the ARPANET," *IEEE Trans. on Comm. COM-28*, 5 (May, 1980), pp. 711-719.

[4] McQuillan, J.M., "Routing algorithms for Computer Networks—A survey," *Proc. National Telecomm. Conf.* (1977) pp. 28:1-6.

[5] Rudin, H., "On Routing and Delta Routing: A Taxonomy and Performance Comparison of Techniques for Packet-Switched Networks," *IEEE Trans. on Comm. COM-24*, 1 (January, 1976), pp. 43-59.

[6] Friedman, D.U., "Communication Complexity of Distributed Shortest Path Algorithms," Master's Thesis, Mass. Inst. of Tech., Dept. of Elec. Eng. and Comp. Sc., Cambridge, MA, (December 1979).

[7] Boggs, D.R., Shoch, J.F., and Taft, E.A., "Pup: An Internetwork Architecture," *IEEE Trans. on Comm. COM-28*, 4 (April 1980), pp. 612-623.

[8] Metcalfe, R.M., and Boggs, D.R., "Ethernet: Distributed Packet Switching for Local Computer Networks," *Comm. ACM 19*, 7 (July 1976), pp. 395-404.

[9] Okuda, N., Kunikyo, T., and Kaji, T., "Ring Century Bus: An Experimental High Speed Channel for Computer Communications," *Proc. Fourth Int. Conf. on Computer Comm.* (September, 1978), pp. 161-166.

[10] Clark, D.D., Pogran, K.T., and Reed, D.P., "An Introduction to Local Area Networks," *Proc. IEEE 66*, 11 (November, 1978), pp. 1497-1517.

[11] Thornton, J.E., Christensen, G.S., and Jones, P.D., "A New Approach to Network Storage Management," *Computer Design* (November, 1975), pp. 81-85.

[12] Hopkins, G.T., "Multimode Communications on the MITRENET," *Proc. Local Area Comm. Network Symp.*, MITRE Corp., Bedford, Mass. (May, 1979), pp. 169-177.

[13] Wilkes, M.V., and Wheeler, D.J., "The Cambridge Digital Communication Ring," *Proc. Local Area Comm. Network Symp.*, MITRE Corp., Bedford, Mass. (May, 1979), pp. 47-61.