Internet Engineering Task Force
INTERNET-DRAFT

Kannan Varadhan
Deborah Estrin
Steve Hotz
USC/ISI
Yakov Rekhter
IBM
July 5, 1994

SDRP Route Construction

Abstract

SDRP packet forwarding and setup have been described in another document, and a prototype implementation completed. However, we are presently relying on manual mechanisms to construct the source routes. This document explores several design options for automating the process of SDRP route construction and thereby making it more dynamic. The proposed approaches make use of IDRP RIB information, and protocol mechanisms.

Status of this Memo

This document in an Internet Draft, and can be found as draft-ietf-sdr-route-construction-00.txt in any standard internet drafts repository. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

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

Please check the I-D abstract listing contained in each Internet Draft directory to learn the current status of this or any other Internet Draft.

Contents

[0] The work of S. Hotz was supported by ARPA Graduate Fellowship, administered through UMIACS contract no. Z984114. The work of D. Estrin and K. Varadhan was supported by the National Science Foundation under contract number NCR-92-06418. Systems research at USC is supported through NSF infrastructure grant, award number CDA-9216321. The work of Y. Rekhter was supported by the National Science Foundation, under contract NCR-92-19216. Views and conclusions expressed in this paper are not necessarily those of the National Science Foundation or of the Advanced Research Projects Agency.

1 Introduction

SDRP forwarding and setup have been implemented, but currently we rely on hand configured routes. This approach obviously does not scale and will limit the use of SDRP in the long (and maybe even short) term. The motivation for this white paper, and the work proposed herein, is to support automatic route construction for SDRP.

We view the problem as one of obtaining routes or route fragments from various sources, and concatenating them to derive complete and valid source routes. We can then install the source routes in the kernel and use them appropriately. This document addresses the problem of obtaining these route fragments[1].

We first note that the RIBs and FIBs used in IDRP route exchange are an instantiation of route fragments, i.e., each entry contains a path vector. We therefore propose to use IDRP to query remote peers for FIB and RIB information. For example, by peering with a remote IDRP speaker and obtaining its FIB, a domain client can obtain routes directly from the selected provider; in particular, routes that are not otherwise propagated/advertised by domains in between the selected provider and the client domain. This effects a relatively trivial policy, referred to as provider selection.

To support the wider range of policies for which SDRP was designed, we propose additional mechanisms: (1) a generalization of the IDRP query called RIB Query (Section 3.1), and (2) initiation of on-demand IDRP computations, called Path Explorer (Section 3.2).

The various requirements for route computation, and the applicable techniques are summarized in table 1.

Source of Route Information	Computation Mechanisms	
	Pre-Computed	On-Demand
Remote	Provider Selection	RIB Query
Local	IDRP/N^2	Path Explorer

Table 1: Classification of the techniques for route computation

2 A First Step: Using SDRP to Extend IDRP for Provider Selection Flexibility

Consider a service provider, "SP," shown in figure 1, connecting to 2 backbone providers. A stub client connects to "SP." We assume that all the domains either speak IDRP or are capable of doing so.

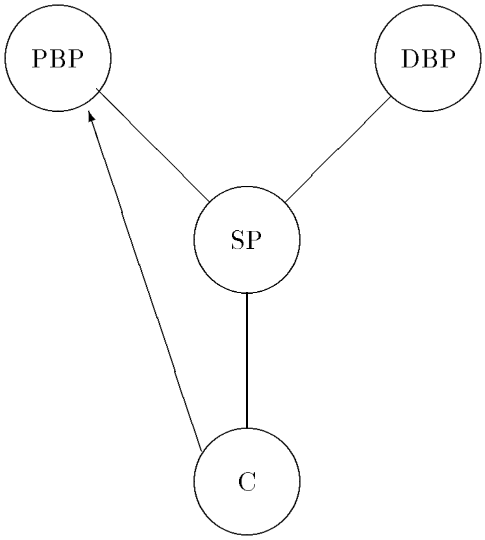
The problem is that in the HbH paradigm of IDRP, domain "C" is constrained to use as the default provider, the same domain that its service provider uses.

Given that the domains are speaking IDRP to exchange routes amongst themselves, if domain "C" can speak IDRP directly to its preferred provider "PBP," to obtain "PBP"'s routes, it can use SDRP to loose source route its traffic via its domain of choice. This permits the client "C" to establish a "virtual adjacency" with the preferred backbone provider, "PBP."

Requirements for Provider Selection

- Protocol specifications to overcome the common subnetwork restrictions in IDRP.
- Mechanisms to install SDRP routes learned in this manner in the FIB. This mechanism will be useful for all other route construction techniques we discuss in this document.

[1] Rules for concatenating these fragments automatically will be discussed elsewhere.



SP: Service Provider
DBP: Default Backbone Provider
PBP: Preferred Backbone Provider
C: Client

Figure 1: Provider Selection in the Internet today, using the unified routing architecture

There is another technique called direct provider selection documented in [4], where a host in the domain requests a choice of provider other than that chosen by the BR (or RS) for that domain. Allowing hosts to select routes directly, instead of relying on a mapping by their BRs, is an important but orthogonal issue and will not be addressed further in this document.

3 Medium Term Goals

Provider selection only extends available routes to those to which the selected provider has access. When the routes provided by the local domain and by the selected providers do not meet the policy or service requirements of the client, we require mechanisms to construct alternative routes. We discuss two such mechanisms: RIB Query and Path Explorer. In the first, transits are queried as in provider selection, however RIB information for select destinations is requested, instead of complete FIB information for all destinations; moreover the request is done on demand, and the information is not continuously updated. In the second, an IDRP computation is initiated on-demand for a select destination.

3.1 RIB Query

We note that a RIB in an IDRP speaker contains route fragments, akin to the FIB. Barring policy constraints, one could connect to a remote peer, request and accumulate their RIBs, and build SDRP routes. A general strategy is to connect to an IDRP peer, request a dump of its RIBs, and install the desired routes locally (in BRs).

A source domain may indicate a preference for certain transit domains and not others. The source domain may not get the desired routes via IDRP in all instances. In such a case, the source domain can use RIB Query to obtain RIBs for the required destination from the preferred transit network. The source domain must obtain at least two route fragments to be successful; one between the transit network and the required destination, and another between the transit network and itself. The complete source route can be derived by concatenating the route fragments appropriately. A mechanism for computing route fragments (or complete source routes when the system is small) is described in the next section.

Requirements for RIB Query

- modifications to the IDRP protocol spec to permit dumping RIBs to a peer.
- Protocols to do the RIB query.
- Code to do the RIB query.
- Methods for source domains to specify transit domain preferences.
- We also need heuristics on whom to query.

3.2 Path Explorer

We define a new mechanism for computing Route Fragments called Path Explorer. Path Explorer was first proposed by Brijesh Kumar(UCL). Path Exploration is an IDRP distributed route computation initiated on-demand in which intermediate nodes: (1) do not exert their own route selection preferences (but they do apply transit policies), (2) propagate but do not store the particular routes. This has the added advantage of permitting the destination to influence the set of choices available to the source.

This path exploration can be done in any number of different ways. The simplest of such methods is for the source to request the destination to compute a path from it to the source; i.e.

explore(start-node)

This "brute force" *explore* computes end to end routes but is too expensive because the number of routes propagated grows exponentially with path length. To constrain route propagation we can exploit the following:

- Routes are already constrained by specified attributes in IDRP such as QoS etc.
- A hop count will constrain route propagation and frequently very long paths are not desired anyway.
- The source can specify a ranking of qualifying routes, and a cut-off, which the destination can use to limit the number of routes propagated.
- If the source knows of a small set of providers via which it will route its packets, it can request a path explorer from the destination to those intermediate proxy points, instead of all the way back to itself. For example:

  proxy-explore(start-node, intermediate-node, originator) or

multi-proxy-explore(start-node, (set of intermediate-nodes), originator)

Requirements for Path Explorer

- Protocols to initiate the path explore and proxy.
- Specifications of the preference functions.
- Modifications in IDRP to perform the following:

  to propagate certain routes without applying local selection criteria.

  to not propagate routes that violate transit policies.

  to not store these routes in the longer term, either in the RIBs or the FIB.

  To store these on demand routes in a short lived cache for the purpose of evaluating viable routes to the cached route to decide which routes to propagate.

  modifications to the IDRP route selection process to use the source specified preference function.
- Implementations of path explorers.

3.3 Path Verifier

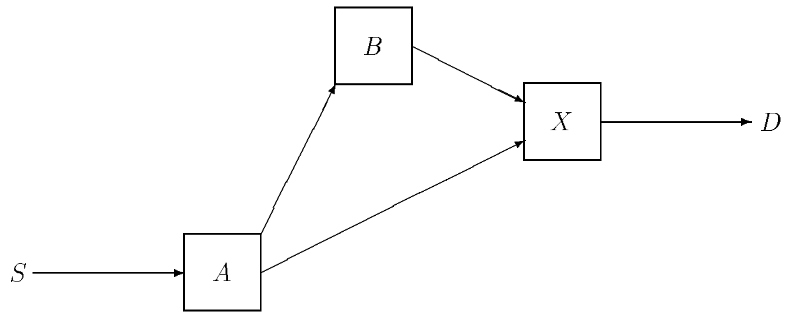
We expect the results of RIB queries and Path Explorers to be cached. Cacheing implies that there is potential for the data to become invalid when it is actually used, or re-used. The data could become invalid either because one or more of the links used in the source route are down; or because the characteristics of the path changed from when they were queried for and computed, to when they are being used. Therefore, we need additional tools to verify the paths that are computed, either by RIB query, or path explorer.

There is also the problem that since we have constructed a source route using a combination of RIB and FIB information from various IDRP peers, the constructed route could violated some packet forwarding constraints. Consider, for instance, an SDRP packet with a source route of $<S, A, B, X, D>$, constructed with the RIBs from A. A's RIBs and the FIB are shown in figure 2. When A chooses to send the packet to B, it recognizes B as an adjacent domain, but really forwards the packet through X. When the packet contains a loose source route, then it is merely counter-intuitive behaviour; on the other hand, it is a clear violation when the packet contains a strict source route.

Even though the functional specifications of the two tasks above are very different, the same tool can be used to satisfy both the requirements. We call this mechanism the path verifier mechanism. This tool will instantiate any RIB routes on paths along the way, as well as verify the characteristics of the route as a sanity check.

Requirements for Path Verifier

- We need to specify the different components of the verification mechanism to check the characteristics of the route.
- We need to specify that an IDRP speaker that provides any subset of its RIBs to a third gateway must be prepared to instantiate any route requested from that subset of the RIBs.



RIBs for A	FIB for A
$<A, B>, <A, B, X>$	$<A, X, B>$
$<A, X>, <A, X, B>$	$<A, X>$

Figure 2: Possible Packet Forwarding Violations when using IDRP RIBs

4 Integrating Route Construction into the NSFNET Routing Architecture

The new NSFNET architecture consists of a very high speed backbone service (the vBNS); other Network Service Providers (NSPs); Network Attachment Points (NAPs) where the NSPs connect to the vBNS and to each other; and the Routing Arbiter (RA) to manage the routing process for the US portion of the global Internet.

The RA architecture[2] consists of the following sub-components:

- Routing Registry Database (RRDB)

  The RRDB contains the configured topology of the Internet. As part of this data, it contains domain level connectivity information, transit and route selection policies.

- Route Servers (RSs) on each NAP

  Each RS contains configurations computed by the RRDB. These configurations govern the exchange of routing information with the different routers connected to the RSs' NAP.

- RA-NOC

  The RA-NOC is the entity that monitors routing in the Internet. As such it does not itself influence route computation. We will not discuss this component in the remainder of the document.

In the following section, we will see how each of the techniques discussed can leverage off the RRDB and the RSs.

4.1 Application of the Route Construction Mechanisms

4.1.1 Provider Selection

Provider Selection is the technique of querying a particular domain for the routes it has precomputed for itself, i.e. its FIB. By using these routes, a client is selecting the particular provider over others that were selected by other BGP/IDRP speakers closer to the client.

The routes maintained by a provider at a NAP may be obtained either directly from the provider, or from one of the RSs at that NAP.

Permitting the client domain to obtain its routes from the RS simplifies connection and authentication issues for the provider. For example, an NSFnet backbone services provider may be required to carry traffic for any NSF specified site, and may not wish to authenticate every single requester individually. On the other hand, a commercial backbone services provider might want to authenticate every single client acquiring its routes and would prefer not to delegate this control to the RS.

The client cannot contact the RRDB for this information, since the RRDB does not contain dynamic information.

4.1.2 RIB Query

The RIB Query mechanism is used to obtain the domain level connectivity information available to the domain. This could be obtained in one of two ways: (1) by querying the RRDB directly for the appropriate connectivity, or (2) by querying a particular domain directly for its RIBs.

Contacting the RRDB is attractive because it refers only to a single point of contact. However it has the disadvantage that it is static information and does not reflect the active status of the topology at any time. Moreover, if the RRDB is contacted, a better option would be to have the RRDB compute the entire route. We discuss this further in section 4.3 below.

Contacting the domain provides more relevant information about the topology, but may require multiple queries to multiple domains to compute a complete route. Using an RS to make the query might simplify the computation.

The FIB for one domain is a subset of the RIBs to a neighbour domain. Given that the RSs at a NAP contain the FIB for all domains incident on the NAP, the RSs have some subset of the RIBs for all domains incident on the NAP.

The RIB query mechanism can directly connect to the RS, and ask for a subset of the RIBs for a single domain of interest. By definition then, the RS can provide the RIBs for the domains that peer with it. Note that this may not be the complete set of RIBs for the domain, especially if it has RIBs from other domains not directly attached to the NAP.

The querying client can evaluate the amount of information available to it from the RS by comparing this information to the domain level connectivity information available to it from the RRDB.

Further evaluation is needed to determine whether performing the RIB query through an RS will be sufficient to compute routes in most instances.

4.1.3 Path Explorer

The source requests the destination domain to start an IDRP computation of routes with certain characteristics.

The source can request the RSs incident on a NAP that the domain is connected to, to start the appropriate IDRP flood for either a proxied, or multi-proxied path explorer. The source can additionally use one or more of the RSs as proxy agents.

4.1.4 Path Verifier

This tool is used operationally prior to actually using a computed route to verify that the route chosen will be usable to the instant it is being used. By itself, this tool will have very little impact for the RA architecture. Hence if the RRDB computes full source routes on demand, this would be a much needed tool.

4.2 Requirements for the mechanisms

There are a number of issues to be resolved to compute routes in the RA architecture. Some of these requirements are:

- Policies need to be established regarding who shall and shall not connect to the RS and the RRDB to compute on-demand routes.
- We need protocols to query the RS and RRDB for various route requests and compute requests.
- Implementations to perform the query.

4.3 Computing Routes Via the RRDB

Since the RRDB is intended contain most of the domain-level topology of the Internet, and the RRDB is used to constructed configuration files for the RSs, it follows that the RRDB could also compute valid source routes that satisfy TOS requirements as well as route selection and transit policies.

Some of the advantages and disadvantages of using the RRDB to compute source routes is a set of tradeoffs discussed below:

- The RRDB is an ideal central location to obtain all information.

  The tradeoff with this is that this may be a hot spot deluged by requests to compute a large number of special case routes.

- The RRDB has complete information about domain level connectivity of the Internet.

  It should be noted that this complete information is necessarily static, and does not reflect the current status of the links and/or domains about which information is provided.

- The RRDB could pre-compute a large number of source routes for various policies and TOS.

  It is not clear that this solution of pre-computing special purpose routes will be tractable.

The tradeoffs must be carefully considered, and the scalability of this approach needs to be investigated carefully.

A Interfaces with IDRP

The following outlines existing IDRP constraints that should be modified for easier interoperation.

1. IDRP requires peers to share a common subnetwork. This requirement should be relaxed to allow a remote peer to either choose a preferred provider, as outlined in section 2, or to query a remote peer for RIBs, as described in section 3.1.
2. IDRP must provide its Adj-RIB-In on demand to support mechanisms such as RIB query (section 3.1).
3. IDRP must provide mechanisms for a client to specify source specific preference functions, which it must then apply in lieu of its standard selection criteria for the requested path.
4. IDRP must provide a mechanism for a peer to advertise a set of routes as being on-demand routes.
5. IDRP must provide a mechanism by which a peer that gets a route marked as being an on-demand route must not apply any local selection criteria.
6. An IDRP peer need not store on-demand routes in its local RIB.
7. IDRP peers may maintain a short-lived cache to compare viable on-demand routes and decide which ones they should propagate.

Since a route that violates transit criteria is not a valid route, an IDRP peer continues to apply its transit criteria in all of the techniques outlined in this paper.

Security Considerations

Security Considerations are not discussed in this memo.

Author's Present Addresses

Kannan Varadhan
USC/Information Sciences Institute
4676, Admiralty Way,
Marina Del Ray, Ca 90272.
Phone: +1 310 822 1511 x 402
email: kannan@isi.edu

Deborah Estrin
USC/Information Sciences Institute
4676, Admiralty Way,
Marina Del Ray, Ca 90272.
Phone: +1 310 822 1511 x 253
email: estrin@isi.edu

Steve Hotz
USC/Information Sciences Institute
4676, Admiralty Way,
Marina Del Ray, Ca 90272.
Phone: +1 310 822 1511
email: hotz@isi.edu

Yakov Rekhter
T.J. Watson Research Center, IBM Corporation
P.O. Box 218
Yorktown Heights, NY 10598
Phone: +1 914 945 3896
e-mail: yakov@watson.ibm.com

References

[1] D. Estrin, Y. Rekhter, and S. Hotz. Scalable inter-domain routing architecture. In *Proceedings of the ACM SigComm*, pages 40–52, Baltimore MD, August 1992. ACM.

[2] Deborah Estrin, Jon Postel, and Yakov Rekhter. Routing arbiter architecture. Work in Progress.

[3] Deborah Estrin, Daniel Zappala, Tony Li, Yakov Rekhter, and Kannan Varadhan. Source demand routing: Packet format and forwarding specification (version 1). *Internet Working Draft, SDR Working Group*, March 1994. Work in Progress.

[4] Y. Rekhter. Selecting a direct provider. Internet Working Draft, BGP Working Group, 1994. Work in progress.