Network Working Group
Request for Comments: 1383
C. Huitema
INRIA
December 1992

An Experiment in DNS Based IP Routing

Status of this Memo

Table of Contents

1. Routing, scaling and hierarchies

Several recent studies have outlined the risk of "routing explosion" in the current Internet: there are already more than 5000 networks announced in the NSFNET routing tables, more than 7000 in the EBONE routing tables. As these numbers are growing, several problems occur:

*   The size of the routing tables grows linearly with the number of connected networks; handling this larger tables requires more resources in all "intelligent" routers, in particular in all "transit" and "external" routers that cannot rely on default routes.

*   The volume of information carried by the route exchange protocols such as BGP grows with the number of networks, using more network resources and making the reaction to routing events slower.

*   Explicit administrative decisions have to be exercised by all transit networks administrators which want to implement "routing policies" for each and every additional "multi-homed" network.

The current "textbook" solution to the routing explosion problem is to use "hierarchical routing" based on hierarchical addresses. This is largely documented in routing protocols such as IDRP, and is one of the rationales for deploying the CIDR [3] addressing structure in the Internet. This textbook solution, while often perfectly adequate, as a number of inconveniences, particularly in the presence of "multihomed stubs", e.g., customer networks that are connected to more than one service providers.

The current proposal presents a scheme that allows for simple routing. It is complementary with the classic "hierarchical routing" approach, but provides an easy to implement and low cost solution for "multi-homed" domains. The solution is a generalization of the "MX record" scheme currently used for mail routing.

2. Routing based on MX records

The "MX records" are currently used by the mail routing application to introduce a level of decoupling between the "domain names" used for user registration and the mailbox addresses. They are particularly useful for sending mail to "non connected" domains: in that case, the MX record points to one or several Internet hosts that accept to relay mail towards the target domain.

We propose to generalize this scheme for packet routing. Suppose a routing domain D, containing several networks, subnetwork and hosts, and connected to the Internet through a couple of IP gateways. These gateways are dual homed: they each have an address within the domain D -- say D1 and D2 -- and an address within the Internet -- say I1 and I2 --. These gateways also have a particularity: they retain information, and don't try to announce to the Internet any reachibility information on the networks contained within "D". These networks however have been properly registered; a name server accessible from the Internet contains the "in-addr.arpa" records that enable reverse "address to name" lookup, and also contains the network level equivalent of "MX records", say "RX records". Given any host address Dx within D, one can get "RX records" pointing to the Internet addresses of the gateways, I1 and I2.

A standard Internet router Ix cannot in principle send a packet to the address Dx: it does not have any corresponding routing information. However, if the said Internet router has been modified to exploit our scheme, it will query the DNS with the name build up from "Dx" in the "in-addr.arpa" domain, obtain the RX records, and forward the packet towards I1 (or I2), using some form of "source routing". The gateway I1 (or I2) will receive the packet; its routing tables contain information on the domain D and it can relay the packet to the host Dx.

At this stage, the readers should be convinced that we have presented a scheme that:

* avoid changes in host IP addresses as topology changes, without requiring extra overhead on routing (provided that the routing employs some form of hierarchical information aggregation/abstraction),

* allow to support multihomed domains without requiring additional overhead on routing and without requiring hosts to have explicit knowledge of multiple addresses.

They should also forcingly scratch their head, and mumble that things can't be so simple, and that one should perhaps carefully look at the details before assuming that the solution really works.

3. Evaluation of DNS routing

Several questions come to mind immediately when confronted to such schemes:

- Should all relays access the DNS? What about possible loops?

- Will the performances be adequate?

- How does one choose the best gateway when several are announced? What happens if the gateway is overloaded, or unreachable?

- What if the directory cannot be accessed?

- How does it work in the reverse direction?

- Should we use tunnelling or loose source routing?

- Can we be more general?

There may indeed be more questions, but these ones, at least, have been taken into account in the setting of our experiment.

3.1. Loops and relays

In the introduction to DNS-IP routing, we mentioned that the packets would be directed towards the access gateway I1 or I2 by means of "source routing" or "tunnelling". This is not, stricto sensu, necessary. One could imagine that the packet would simply be routed "as if it was directed towards I1 or I2". The next relay would, in turn, also access the DNS to get routing information and forward the packet.

Such a strategy would have the advantage of leaving the header untouched and of letting the transit nodes choose the best routing towards the destination, based on their knowledge of the reachability status. It would however have two important disadvantages:

- It would oblige all intermediate relays to access the DNS,

- It would oblige all these relays to exploit consistently the DNS information.

Obliging all intermediate gateways to access the DNS is impractical in the short term: it would mean that we would have to update each and every transit relay before deploying the scheme. It could also have an important performance impact: the "working set" of transit relays is typical much wider than that of stub gateways, and the argument presented previously on the efficiency of caches may not apply. This would perhaps remain impractical even in the long term, as it the volume of DNS traffic could well become excessive.

The second argument would apply even if the performance problem had been solved. Suppose that several RX records are registered for a given destination, such as I1 and I2 for Dx in our example, and that a "hop by hop routing" strategy is used. There would be a fair risk that some relays would choose to route the packet towards I1 and some others towards I2, resulting in inefficient routing and the possibility of loops.

In order to ensure coherency, we propose that all routing decisions be made at the source, or by one of the first relays near the source.

3.2. Performances and scaling

The performance impact of using the DNS for acquiring routing information is twofold:

* The initial DNS exchanges required for loading the information may induce a response time penalty for the users,

* The extra DNS traffic may contribute to overloading the network.

We already have some experience of DNS routing in the Internet for the "mail" application. After the introduction of the "MX record", the mail routing slowly evolved from a hardwired hierarchy, e.g., send all mail to the addresses in the ".FR" domain to the french gateway, towards a decoupling between a name hierarchy used for registration and the physical hierarchy used for delivery.

If we consider that the mail application represent about 1/4th of the Internet traffic, and that a mail message seldom include more than half a dozen packets, we come to the point that DNS access is already needed at least once for every 24 packets. The performances are not apocalyptic -- or someone would have complained! In fact, if we generalize this, we may suppose that a given host has a "working set" of IP destinations, and that some caching strategy should be sufficient to alleviate the performance effect.

In the scheme that we propose, the DNS is only accessed once, either by the source host or by an intelligent router located near the source host. The routing decision is only made once, and consistent routing is pursued in the Internet until reaching an access router to the remote domain.

The volume of DNS traffic through the NSFNET, as collected by MERIT, is currently about 9%. When a host wants to establish communication with a remote host it usually need to obtain the name - IP address mapping. Getting extra information (I1 or I2 in our example) should incur in most cases one more DNS lookup at the source. That lookup would at most double the volume of DNS traffic.

3.3. Tunneling or source routing

Source directed routing, as described above, can be implemented through one of two techniques: source routing, or a form of encapsulation protocol. For the sake of simplicity, we will use source routing, as defined in [1]: we don't have to define a particular tunnelling protocol, and we don't have to require hosts to implement a particular encapsulation protocol.

3.4. Choosing a gateway

A simplification to the previous problem would be to allow only one RX record per destination, thus guaranteeing consistent decisions in the network. This would however have a number of draw-backs. A single access point would be a single point of failure, and would be connected to only one transit network thus keeping the "customer locking" effect of hierarchical routing.

We propose that the RX records have a structure parallel to that of MX records, i.e., that they carry associated with each gateway address a preference identifier. The source host, when making the routing decision based on RX records, should do the following:

- List all possible gateways,
- Prune all gateways in the list which are known as "unreachable" from the local site,
- If the local host is present in the list with a preference index "x", prune all gateways whose preference index are larger than "x" or equal to "x".
- Choose one of the gateway in the list. If the list is empty, consider the destination as unreachable.

Indeed, these evaluations should not be repeated for each and every packet. The routers should maintain a cache of the most frequently used destinations, in order to speed up the processing.

3.5. Routing dynamics

In theory, one could hope to extract "distance" information from the local routing table and combine it with the preference index for choosing the "best" gateway. In practice, as shown in the mail context, it is extremely difficult to perform this kind of test, and one has to rely on more heuristical approaches. The easiest one is to always choose a "preferred gateway", i.e., the gateway which has the minimal preference index. One could also, alternatively, choose one gateway at random within the list: this would spread the traffic on several routes, which is known to introduce better load sharing and more redundancy in the network.

As this decision is done only once, the particular algorithm to use can be left as a purely local matter. One domain may make this decision based purely on the RX record, another based purely on the routing information to the gateways listed in the RX record, and yet the third one may employ some weighted combinations of both.

Perhaps the most important feature is the ability to cope rapidly with network errors, i.e., to detect that one of the route has become "unreachable". This is clearly an area where we lack experience, and where the experiment will help. One can think of several possible solutions, e.g.,:

* Let intermediate gateways rewrite the loose source route in order to replace an unreachable access point by a better alternative,

* Monitor the LSR options in the incoming packets, and use the reverse LSR,

* Monitor the "ICMP Unreachable" messages received from intermediate gateways, and react accordingly,

* Regularly probe the LSR, in order to check that it is still useful.

A particularly interesting line would be to combine these connectivity checks with the transport control protocol acknowledgments; this would however require an important modification of the TCP codes, and is not practical in the short term. We will not try any such interaction in the early experiments.

The management of these reachability informations should be taken into account when caching the results of the DNS queries.

3.6. DNS connectivity

It should be obvious that a scheme relying on RX records is only valid if these records can be accessed. By definition, this is not the case of the target domain itself, which is located at the outer fringes of the Internet.

A domain that want to obtain connectivity using the RX scheme will have to replicate its domain name service info, and in particular the RX records, so has to provide them through servers accessible from the core of the Internet. A very obvious way to do so is to locate replicated name servers for the target domain in the access gateways "I1" and "I2".

3.7. On the way back

A source located in the fringe domain, when accessing a core Internet host, will have to choose an access relay, I1 or I2 in our example.

A first approach to the problem is to let the access gateway relay the general routing information provided by the routing domains through the fringe network. The fringe hosts would thus have the same connectivity as the core hosts, and would not have to use source directed routing. This approach has the advantage of leaving the packets untouched, but may pose problems should the transit network need to send back a ICMP packet: it will have to specify a source route through the access gateway for the ICMP packet. This would be guaranteed if the IP packets are source routed, as the reverse source route would be automatically used for the ICMP packet. We are thus led to recommend that all IP packets leaving a fringe domain be explicitly source routed.

The source route could be inserted by the access gateway when the packet exits the fringe domain, if the gateway has been made aware of our scheme. It can also be set by the source host, which would then have to explicitly choose the transit gateway, or by the first router in the path, usually the default router of the host sending the packets. As we expect that hosts will be easier to modify than routers, we will develop here suitable algorithms.

The fringe hosts will have to know the set of available gateways, of which all temporarily unreachable gateways shall indeed be pruned. In the absence of more information, the gateway will be chosen according to some preference order, or possibly at random.

It is very clear that if a "fringe" host wants to communicate with another "fringe" host, it will have to insert two relays in the LSR, one for the domain that sources the packet, and one for the domain where the destination resides.

3.8. Flirting with policy routing

The current memo assumes that all gateways to a fringe domain are equivalent: the objective of the experiment is to test and evaluate a simple form of directory base routing, not to provide a particular "policy routing" solution. It should be pointed out, however, that some form of policy routing could be implemented as a simple extension to our RX scheme.

In the proposed scheme, RX records are only qualified by an "order of preference". It would not be very difficult to also qualify them with a "supported policy" indication, e.g., the numeric identifier of a particular "policy". The impact on the choice of gateways will be obvious:

- When going towards a fringe network, one should prune from the usable list all the gateways that do not support at least one of the local policies,

- When exiting a fringe network, one should try to assess the policies supported by the target, and pick a corresponding exit gateway,

- When going from a fringe network towards another fringe network, one should pick a pair of exit and access gateway that have matching policies.

In fact, a similar but more general approach has been proposed by Dave Clark under the title of "route fragments". The only problem here are that we don't know how to identify policies, that we don't know whether a simple numeric identifier is good enough and that we probably need to provide a way for end users to assess the policy on a packet per packet or flow per flow basis. In short, we should try to keep the initial experiment simple. If it is shown to be successful, we will have to let it evolve towards some standard service; it will be reasonable to provide policy hooks at this stage.

4. Rationales for deployment

Readers should be convinced, after the previous section, that the DNS-IP routing scheme is sleek and safe. However, they also are probably convinced that a network which is only connected through our scheme will probably enjoy somewhat less services than if they add have full traditional connectivity. We can see two major reasons for inducing users into this kind of scheme:

- Because they are good network citizen and want to suffer their share in order to ease the general burden of the Internet,

- Because they are financially induced to do so.

We will examine these two rationales separately.

4.1. The good citizens

A strong tradition of the Internet is the display of cooperative spirit: individual users are ready to suffer a bit and "do the right thing" if this conduct can be demonstrated to improve the global state of the network -- and also is not overly painful.

Restraining to record your internal networks in the international connectivity tables is mainly an advantage for your Internet partners, and in particular for the backbone managers. The normal way to relieve this burden is to follow a hierarchical addressing plan, as suggested by CIDR. However, when for some reason the plan cannot be followed, e.g., when the topology just changed while the target hosts have not yet been renumbered, our scheme provides an alternative to "just announcing one more network number in the tables". Thus, it can help reducing the routing explosion problem.

4.2. The commercial approach

Announcing network numbers in connectivity tables does have a significant cost for network operators:

- larger routing tables means more memory hence more expensive routres,

- more networks means larger and more frequent routing messages, hence consume more network resources,

- more remote networks means more frequent administrative decisions if policies have to be implemented.

These costs are very significant not only for regionals, but also for backbone networks. It would thus be very reasonable, from an economical point of view, for a backbone to charge regionals according to the number of networks that they announce. A similar line of reasoning can be applied by the regionals, which could thus give the choice to their customers between:

- being charged for announcing an address of their choice,

- or being allocated at a lower cost a set of addresses in an addressing space belonging to the regional.

Our scheme may prove an interesting tool if the charge for individual addresses, which are necessary for "multi homed" clients, becomes too high.

5. The experimental development

The experimental software, implemented under BSD Unix in a "socket" environment, contains two tasks:

- a real time forwarder, which is implemented inside the kernel and handles the source demanded routes,

- a DNS query manager, which transmit to the real time forwarder the source routing information.

In this section, we will describe the real time forwarder, the query manager, the format of the DNS record, and the interface with the standard IP routers.

5.1. DNS record

In a definitive scheme, it would be necessary to define a DNS record type and the corresponding "RX" format. In order to deploy this scheme, we would then have to teach this new format to the domain name service software. While not very difficult to do, this would probably take a couple of month, and will not be used in the early experimentations, which will use the general purpose "TXT" record.

This record is designed for easy general purpose extensions in the DNS, and its content is a text string. The RX record will contain three fields:

- A record identifier composed of the two characters "RX". This is used to disambiguate from other experimental uses of the "TXT" record.

- A cost indicator, encoded on up to 3 numerical digits. The corresponding positive integer value should be less that 256, in order to preserve future evolutions.

- An IP address, encoded as a text string following the "dot" notation.

The three strings will be separated by a single comma. An example of record would thus be:

domain	type	record	value
-			
*.27.32.192.in-addr.arpa	IP	TXT	RX, 10, 10.0.0.7

which means that for all hosts whose IP address starts by the three octets "192.32.27" the IP host "10.0.0.7" can be used as a gateway, and that the preference value is 10.

5.2. Interface with the standard IP router

We have implemented our real time forwarder "on the side" of a standard IP router, as if it were a particular subnetwork connection: we simply indicate to the IP router that some destinations should be forwarded to a particular "interface", i.e., through our real time forwarder.

Of particular importance is indeed to know efficiently which destinations should be routed through our services. As the service would be useless for destinations which are directly reachable, we have to monitor the "unreachable" destinations. We do so by monitoring the "ICMP" messages which signal the unreachable destination networks, and copying them to the DNS query manager.

There are indeed situations, e.g., for fringe networks, where the router knows that destinations outside the local domain will have to be routed through the real time forwarder. In this case, it makes sense to declare the real time forwarder as the "default route" for the host.

5.3. The DNS query manager

Upon reception of the ICMP message, the query manager updates the local routing table, so that any new packet bound to the requested destination becomes routed through the real time forwarder.

At the same time, the query manager will send a DNS request, in order to read the RX records corresponding to the destination. After reception of the response, it will select a gateway, and pass the information to the real time forwarder.

5.4. The real time forwarder

When the real time forwarder receives a packet, it will check whether a gateway is known for the corresponding destination. If that is the case, it will look at the packet, and insert the necessary source routing information; it will then forward the packet, either by resending it through the general IP routing program, or by forwarding it directly to the network interface associated to the intermediate gateway.

If the gateway is not yet known, the packet will be placed in a waiting queue. Each time the query manager will transmit to the real time forwarder new gateway information, the queue will be processed, and packets for which the information has become available will be forwarded. Packets in this waiting queue will "age"; their time to live counts will be decremented at regular intervals. If it become null, the packets will be destroyed; an ICMP message may be forwarded.

The DNS query manager may be in some cases unable to find RX information for a particular destination. It will in that case signal to the real time forwarder that the destination is unreachable. The information will be kept in the destination table; queued packet for this destination will be destroyed, and new packets will not be forwarded.

The information in the destination table will not be permanent. A time to live will be associated to each line of the table, and the aging lines will be periodically removed.

5.5. Interaction with routing protocols

The monitoring of the "destination unreachable" packets described above is mostly justified by a desire to leave standard IP routing, and the corresponding kernel code, untouched.

> If the IP routing code can be modified, and if the local host has full routing tables, it can take the decision to transmit the packets to the real time forwarder more efficiently, e.g., as a default action for the networks that are not announced in the local tables. This procedure is better practice, as it avoids the risk of loosing the first packet that would otherwise have triggered the ICMP message.

6. Acknowledgments

We would like to thank Yakov Rekhter, which contributed a number of very helpful comments.

7. Conclusion

This memo suggests an experiment in directory based routing. The author believes that this technique can be deployed in the current Internet infrastructure, and may help us to "buy time" before the probably painful migration towards IPv7.

The corresponding code is under development at INRIA. It will be placed in the public domain. Interested parties are kindly asked to contact us for more details.

8. References

[1] Postel, J., "Internet Protocol - DARPA Internet Program Protocol Specification", STD 5, RFC 791, DARPA, September 1981.

[2] Clark, D., "Building routers for the routing of tomorrow", Message to the "big-internet" mailing list, reference <9207141905.AA06992@ginger.lcs.mit.edu>, Tue, 14 Jul 92.

[3] Fuller, V., Li, T., Yu, J., and K. Varadhan, "Supernetting: an Address Assignment and Aggregation Strategy", RFC 1338, BARRNet, cisco, Merit, OARnet, June 1992.

9. Security Considerations

Security issues are not discussed in this memo.

10. Author's Address

Christian Huitema
INRIA, Sophia-Antipolis
2004 Route des Lucioles
BP 109
F-06561 Valbonne Cedex
France

Phone: +33 93 65 77 15
EMail: Christian.Huitema@MIRSA.INRIA.FR