Internet Routing and DNS Voodoo in the Enterprise

D. Brian Larkins – Lucent Technologies

ABSTRACT

This paper describes the process used to transition from a legacy intranet to a modern Internet access architecture. During the period of the Lucent/AT&T/NCR tri-vestiture, much care was given to re-engineer and re-design Lucent's data networking infrastructure with a modern and flexible design. As a part of this re-structuring, the existing Internet access architecture was viewed as archaic and in desperate need of redesign. The legacy intranet was isolated from the Internet in many ways including separate root name servers, a complete lack of routing information to or from the Internet, and everything was passed through home-grown application-layer proxy software. To remedy this, a project was created to provide transparent proxyless access to Internet hosts and applications. This project entailed designing a routing architecture that provided connectivity, redundancy, and manageability. In addition to routing issues, Lucent's DNS infrastructure would also need a redesign to handle new responsibilities given to it.

Introduction

In mid-1996, new packet-filtering firewalls were deployed to replace the existing (and aging) application-layer firewalls which proxied web traffic as well as telnet and ftp. These new firewalls were deployed in conjunction with new web proxy-caching servers. This alleviated much of the older firewalls' load, and provided web access faster by an order of magnitude. The plan at that time was to phase in the proxyless firewalls by making big improvements in web access and adding in support for generic TCP services later. The initial deployment of these firewalls with web proxies was called Phase 1.

The goal of the Phase 2 deployment of the proxyless firewall architecture was to enable internal clients to access Internet based TCP services. In order to provide this type of access to internal users, three primary problems needed to be solved. First, a mechanism for provid ing routing information to and from the Internet consonant with our routing policy had to be determined. The most difficult technical problem of this entire endeavor was designing the routing architecture to be resilient and robust, but also to not mess up our stateful packet-filtering firewalls with asymmetric routing. Second, the DNS needed to be modified to allow internal hosts the ability to resolve Internet hosts, while still retaining policy-based controls over such things as outgoing email, name resolution of joint-ventures, acquisitions, etc. Thirdly, (and perhaps the most stressful of all) was developing an outbound access policy that appeased both the R&D communities and the corporate security folks. This document primarily focuses on the technical problems related to implementing a proxyless routing and DNS infrastructure and will leave in-depth discussions on policy to other brave souls.

Historical Background

Many of the difficulties with deploying what might otherwise have been a simple design result from being split off from a 100-year old company with 310,000 employees. At the time Lucent was spun-off from AT&T, there were the equivalent of at least five ''intranets'' connected together in a tenuous fashion at best. During the separation of these networks, a new internal backbone was created [Umali96]. Using the hierarchal properties of the OSPF routing protocol, business-unit specific networks, regional networks (Europe, Asia, etc.), and special-purpose networks (e.g., research) were all knitted together [Moy94]. Also at this time, the WWW was becoming more and more vital to business. The result of the massive growth in IP network usage and the churn caused by splitting one of the world's largest private networks was a little too much for the old R&D-based Internet gateways to handle [Umali97].

At the time, the only way to access the Internet was through email, a web-proxy, or through a custom application-layer proxy [Cheswick94]. The proxy software required a customized client or linking source to a custom library. On systems which were well maintained by able system administrators this wasn't too much of a burden, but for developers who had unique environments, re-compiling and/or porting was troublesome. For most PC users, Internet access beyond the web or email wasn't an option. Given the 120,000+ PC users within the company, this was inadequate connectivity for a data/telecommunications equipment manufacturer.

In early 1997, a project was initiated to evaluate, design, and deploy a mechanism for providing access to the Internet without the need for customized client software. It became clear early on that this would be a difficult task. The existing intranet infrastructure was never designed for easy integration with the Internet. In many ways Lucent was ‘‘self-rooted’’.

The primary internal name servers were configured to be root name servers in order to make it easy to control DNS policy such as mail flow. This was a carry over from the AT&T days. Self-rooting the DNS made the transition from AT&T to Lucent much easier, but it made it more difficult to integrate with the Internet. If requests arrived at the internal root name servers for external hostnames, they would be dropped instead of being forwarded to the appropriate DNS authority.

Likewise, the OSPF backbone was the core of Lucent's intranet. If the backbone didn't have a route to the destination network, the packet was dropped. This was more fortuitous than the DNS case because the lack of a default route on the backbone allowed us to consider it's use as a possible solution to some of our Internet routing problems.

The two most difficult issues to deal with were organizational issues and security policy issues. The sheer size of Lucent becomes readily apparent with any attempt to change the fundamental underpinnings of it's data network. The number of organizations involved in the operations and engineering of the network is truly staggering. Currently there are at least four major separate groups supporting WAN engineering, two groups supporting DNS, and another two supporting firewall policy and engineering. Getting buy-in from all the right groups was critical for the project's success, but was a project in and of itself to find out who needed to be involved and coordinating communication between them.

The organizational issues created extra requirements that might not ordinarily have appeared with a homogeneous networking group. It became desirable to partition responsibilities around organizational boundaries instead of technical ones. The one benefit that this provided inherently was broad peer review. As a result many design flaws were found and eliminated early in the design process.

Besides organizational issues, security policy quickly rose to the list of hot topics. Lucent traditionally has aligned itself more with financial institution security rather than Silicon Valley start-up. On one hand there is the R&D community which would like a university-like environment, and the corporate security folks on the other, which would prefer retina-scans prior to Internet usage. Discussions continue to this day regarding the overall access policy for Internet usage.

Design Philosophy

In order to decompose the proxyless Internet project into meaningful and manageable tasks, we split the effort into three principal areas: routing, DNS, and security policy issues. This left us with two technical problems and one policy problem. With respect to routing and DNS, traditional engineering principals were used to further break down the project into tasks. To help guide the design process we adopted goals for both the routing and DNS problems which helped specify the ideal architecture. This section presents the guidelines that were generated for the routing architecture. The DNS issues are presented below.

Routing Design Philosophy

The philosophy behind the routing architecture is as below.

1. A proxyless routing architecture should be as simple as possible, but no simpler.

The less complexity that there is in a routing design, the less possibility for errors. These errors include configuration problems, troubleshooting difficulties, route pollution, route loops, etc. A design that meets all the requirements for a proxyless routing architecture should be as simple as possible to implement, but robust enough to meet all design criterion.

2. A proxyless routing architecture should be fault tolerant.

An Internet access outage at any single point should not cause significant loss of service for the corporation. Redundancy should exist in all designs to prevent significant outages. Failure of connections active at the time of a fault is acceptable. All connections initiated after a fault should proceed through an alternate path.

3. A proxyless routing architecture should be dynamic.

Fault detection should be transparent and automatic. Routing around a lost network egress point should happen quickly and without human intervention. The faults that should be detected are a any loss of path connectivity to the Internet from a backbone/Internet border router. A loss of any component in the path should trigger a new route to be advertised to the corporate backbone.

4. A proxyless routing architecture should be symmetric.

Internet destined traffic should route out a selected gateway point. The response traffic from the external server should return to the original exit point, to be correctly routed to the internal host. In addition, routes should be stable internally, to ensure that Internet bound traffic is routed through the same gateway when originated by an internal host. Path flapping is not a desired method of traffic flow, even in the interest of load balancing.

NOTE: This design goal could change with the ability to effectively share stateful firewall connection information. It will still be desirable for all Internet bound traffic to take a deterministic path, though this path may not be strictly symmetric.

5. **A proxyless routing architecture should be secure.**

Routes generated from Internet Service Providers are not trusted. All routing information derived from exterior route peers is suspect, and thus should not be allowed to significantly influence traffic patterns within the corporate network. We should not trust that our service providers will protect our internal route tables from being polluted. For example, we should protect against accepting internal routing information from the Internet. Additionally, filtering needs to prevent the leaking of internal information (either routing updates or data traffic itself) to the Internet.

Further, route distribution mechanisms themselves should not pose significant security risks to the internal data network. Router to router communications should be strictly limited and significant protocol filtering should occur to limit the risk of contamination. This is even more important when the flow of information crosses between exterior (untrusted) and interior (trusted) networks.

Problems to be Solved – Part I: Routing

In order to provide routes to the Internet, it is essential that the corporate backbone routers be aware of exit points to the Internet. By the same token it is essential that the Internet have routes by which the traffic can be returned to the Lucent internal network. Solving the problem of how to route internal packets to the Internet is separate from how to route Internet originating packets back to the corporate intranet. We'll break the following discussion into two parts; the first examining traffic originating from the intranet and destined for the Internet and the second covering the converse.

Internal Routing Issues

Traffic destined for the Internet must be routed through one of the corporate Internet firewalls. Within the Lucent data network, routing is hierarchal. End-users are attached to sub-areas which typically run OSPF within the sub-area. Areas (in the OSPF sense) are assigned based on geography in the case of our European and Asian regions, organization in the case of some business units, as well as history for much of the R&D community.

No matter which routing protocol is used within each sub-area, the routing information is distributed into OSPF at the backbone through an OSPF *Area Border Router* (ABR). The ABR is a member of both the sub-area's routing domain as well as the backbone's area [Khan97]. Using OSPF terminology, the backbone is known as *Area 0*. As a result of the route redistribution performed by the ABR, the Area 0 backbone contains all the routing information for the entire company.

The benefit of this is that we can assume that for any packet, a local router either knows itself the path it should take within the sub-area or will forward it "up" to the backbone. A direct consequence of this is that routing packets to the Internet can be reduced to having the Area 0 routers be "Internet aware". The remaining routers within the intranet already forward packets destined to unknown networks to the Area 0 routers.

There are three primary ways to provide route information to the backbone routers with respect to Internet reachability: default routing, partial Internet routing, and full Internet routing. Default routing entails the distribution of either static or dynamically sourced default routes into the OSPF backbone. Any traffic that the backbone doesn't have a known route for will be forwarded to the Internet. Partial and full Internet routing means obtaining a partial or complete list of all the routes announced on the Internet and distributing them to routers attached to the backbone. By having a complete list of routes, it is possible for packets to take the shortest path out and perform load balancing.

Border Gateway Protocol 4 – Your Friend

No matter which alternative is chosen, the de facto routing protocol used to communicate between *Autonomous Systems* (AS) is the *Border Gateway Protocol* (BGP) [Chandra97] [Halabi97]. Autonomous systems are the way that the Internet is broken up into organizationally separate networks. BGP is the glue that knits every corporation, organization, university, and service provider together to the Internet. BGP is an *exterior gateway protocol* (EGP) because it was designed specifically to deal with routing between distinct autonomous systems. BGP contains features which allow network administrators to carefully control the sending and receiving of routing information. Whereas OSPF and IGRP are *interior gateway protocol*s (IGPs) and designed to distribute routing information within an autonomous system, BGP is a tool to implement an organizations' routing policy in addition to exchanging routes.

BGP has two flavors which are determined by the manner in which BGP is configured [Rekhter95b]. First, exterior BGP (EBGP) is used when two BGP peers are in differing AS's. EBGP is used when exchanging routing information between our routers and that of our ISP's. Alternatively, interior BGP (IBGP) is used when two BGP peers are within the same AS. The primary difference between the two is in the rules that are used to exchange routes so as to prevent route loops. Another important difference is that EBGP peers should be directly connected, while IBGP peers only need to be able to connect via TCP [Rekhter95a]. The ramifications of this will be apparent later.

A Return to the Internal Routing Problem

The principal behaviors that are desired in the proxyless architecture are fault-tolerance (i.e. a single ISP can fail without a significant loss in connectivity), and symmetry (traffic that leaves a specific gateway, returns through that gateway). These constraints and the network topology during the design phase of the proxyless routing architecture will lead us to the selected design.

Organizationally, the Internet perimeter is engineered and maintained by a separate organization than the corporate backbone. Consequently, operational and maintenance issues at the time prohibited the use of BGP directly on the core backbone [Umali97]. In addition, the routers used at each of the seven corporate Internet gateways are only connected through the corporate backbone. I.e. there are no dedicated WAN links between the perimeter routers.

These constraints effectively rule out the use of either partial or full Internet routing as a solution to the internal routing problem. When packets bubble up to the Area 0 backbone, they will eventually get to a BGP- speaking perimeter router. If the router has a full routing view of the Internet, it might decide that another gateway is actually closer to the end destination. Since the backbone is unaware that the alternative path is better (it's only running OSPF, remember?), it will simply forward the packet back to the perimeter BGP router that it forwarded it to last time. Voilà, route loop. In case you're lost, the rule is that IBGP must be running on every router in between multiple AS exit points when attempting to do shortest path routing (as with full or partial Internet routing). Since we have a constraint that doesn't allow BGP on the backbone and another constraint that doesn't allow dedicated WAN links to connect the perimeter routers, we outsmarted ourselves right into a default based architecture.

Since IBGP doesn't require that peers be directly connected, we can run IBGP between each Internet gateway site. The only requirement for IBGP peering is that peers can establish a TCP session with one another. By using IBGP to peer with each border router, exit points can be agreed upon and the corresponding routes injected into the Lucent backbone in a deterministic fashion. Using IBGP between multiple peers does however mandate that there exist a full mesh, or IBGP connections between all peers. The reason for this requirement is to preserve a loop-free topology. The use of BGP route reflectors or confederations [Halabi97] can minimize the complexity of such a topology, thus allowing BGP information to easily be ‘‘tunnelled’’ over our native OSPF backbone.

This strategy also provides redundancy and symmetry. The Lucent BGP routers can each accept a default route from each ISP. We can then weight these routes to prefer a single entry/exit point and inject the route for it into the backbone. If a failure should occur with the primary exit point, the default will cease to be advertised into BGP, which will select the next highest weight default route and proceed to advertise that into OSPF (only one default at a time). This will be discussed in depth below. While we trade optimal routing for a primary/backup type solution, we're making the best of the network we have and a huge improvement over the existing configuration.

In addition to the general traffic destined to the Internet there are various servers (mail, DNS, web proxy) that are located at each gateway location. The desired behavior is that these various servers should always use the local firewall, instead of following a BGP generated default. This causes these servers to always use the Internet connection that is local, as opposed to routing high-volume concentrated services such as web access and mail out the single primary egress point.

This effect can be accomplished with policy routing, which is a way to alter the route traffic takes based on it's source address as opposed to traditional routing based on destination address. In the case of an outage it is possible to have proxy traffic be routed to another firewall exit point, but again may cause load problems on the backbone. The exact mechanism used to accomplish this will be discussed in detail below.

External Routing

When routing traffic back to the Lucent data network, there are issues of symmetry, route disclosure, route announcement restrictions, and provider address independence to consider. There are two primary alternatives to provide routes back into Lucent's internal network: address translation or full route disclosure.

Network address translation (NAT) provides an elegant solution to issues of symmetry and route disclosure. By mapping the internal hosts' IP address to a small pool of Internet addresses we can fix the connection path between remote Internet server and Lucent perimeter gateway. This allows us to avoid a plethora of asymmetric routing problems that can cause serious trouble to stateful packet filters.

Address translation enables us to control the return path of outbound traffic. NAT also allows us to advertise reachability information to the Internet, without disclosing information about the internal network's topology. By hiding the original source address, it is non-trivial to discover our use of address space, and the way our networks are configured. This enhances the level of security provided by our firewalls. An additional constraint of using a NAT-based solution is that the internal route a host takes to the gateway must be consistent. For example, if the internal network isn't configured appropriately it would be possible for two packets to exit through separate gateways. Each gateway would translate the source address to a different mapped address which to the server would look like coming from two different hosts.

When considering a NAT solution it is important to understand how a given implementation of address translation scales. Since Lucent has more than 200,000 hosts, a one-to-one mapping of IP addresses could theoretically consume more than three full class B networks. A TCP or UDP connection can be uniquely identified by the 5-tuple of (*src addr*, *src port*, *dst addr*, *dst port*, *protocol*) [Stevens94]. Given this, it is possible to map both the source port and address for many simultaneous connections to a single translated address if the source port is used to uniquely identify the original source address and port. Overloading a single IP address for hiding multiple internal addresses is called *port address translation* (PAT). Using PAT to hide internal addresses involves ugly issues like trusting your firewall to correctly handle ICMP, well-behaved expiration of translation table entries, etc. Our past experiences with address translation brought considerable skepticism that NAT/ PAT schemes could scale well. Even more devious, troubleshooting complicated problems with NAT in the loop has been exceedingly difficult when using off-the-shelf firewall software.

On the other hand, a full announcement solution would require the external border routers to announce routes for all internal Lucent networks. Lucent has around 115 class B-sized blocks (/16's) that would all have to be announced. Announcing these addresses discloses slightly more information than using NAT. By freely distributing these routes, it becomes easier to determine our internal network topology, and we also disclose the true IP address of an internal machine that accesses an external server.

Full announcement can be very beneficial by removing the added overhead and complexity of NAT. Most packet-filtering firewalls operate with higher performance without NAT policies installed. This also addresses complications that NAT can cause with higher-level protocols that encode the source IP address at the application layer (e.g. FTP). Also, the handling of ICMP packets that may have a translated address in the payload, instead of the expected internal IP address can at times be problematic as well.

Whichever solution is used, some routing information will need to be distributed to the Internet. If a NAT solution is chosen, the address blocks used for the NAT pools must be announced to the Internet by someone. This can be done by either the ISP or ourselves. If we announce routes ourselves, BGP4 is required to peer with our ISP's. On the other hand, we can trade control for complex routing if our ISP's announce the NAT blocks for us.

The Routing Design

After comparing all the positives and negatives of each possible design, we chose to take a primary/backup solution. This means that all direct Internet traffic flows out a single exit point, but with a dynamic failover. In the case of a failure with the primary site, a backup gateway will come online automatically and transparently. We also elected to announce all of our address space to the Internet, eliminating the need for NAT.

There are four primary components to the routing design:

- To advertise routes for Lucent's networks to the Internet
- To accept routes to the Internet from our ISP's
- To select appropriate routes for outbound traffic
- To advertise selected routes into the OSPF backbone

Inbound Traffic Route Announcements

In order for traffic originating from Lucent networks to the Internet to be routed back correctly, the Internet must have routes for Lucent's networks. The preferred way to do this is to perform full route announcement for all of Lucent's networks to the Internet. This simplifies the firewall configurations considerably, and also bypasses some side effects caused by firewalls performing address translation.

To announce all of the Lucent routes to the Internet, it is necessary to register RIPE-181 or RPSL compliant objects with the *Internet Routing Registry* (IRR). The IRR is a collection of several distributed routing databases. In particular, Lucent needs to submit it's routing updates to the Route Arbiter Database (RADB) which is a principal U.S. routing registry. Many ISP's use the IRR to directly generate filters that control the propagation and distribution of routes.

In addition to the routing registries, it is also vital to provide direct information via an external routing protocol directly with an ISP's router. The only way to exchange routes with all of our ISP's is by using BGP. The BGP peering that takes place between the different autonomous systems (Lucent's AS and the ISP's AS) is via External BGP, or EBGP.

To ensure that routing is symmetric, Lucent's routes will be announced at several gateways. A BGP trick to prioritize routes from the announcing end is to add additional hops to less desirable routes. This is achieved traditionally by prepending extra copies of one's own AS number on all outgoing route announcements. Routers elsewhere on the Internet will get one route from each gateway, but some will have extra AS hops tacked onto the AS path information. Since these routes take a longer path to reach the same network, the primary return path will be preferred.

In the case of a primary gateway failure, Internet routers will simply use the next shortest path, even though it may contain extra AS path information.

External Outbound Route Acceptance

In order for internal traffic to be dynamically routed to the Internet, routes must be advertised from our ISP's to the Lucent backbone. It is necessary to accept a default route generated from each ISP to determine the best available default to inject into the OSPF backbone.

If the physical link should fail between the EBGP peers, the BGP session will disappear between the two routers. When the session is dropped between peers the routing information will expire, typically in about 90 seconds. This causes the default route to be withdrawn, which may or may not affect the route selection process.

Internal Route Selection

After the default routes have been received by all Lucent BGP speaking routers, the best available entry/exit point must be selected. ''Best available'' is determined by a policy decision based on usage of our internal WAN links and concentration of users. Each candidate gateway is given a weight according to policy (e.g. the primary gateway will have the best weight, the first secondary the next best, etc.)

By configuring a BGP attribute called *local preference*, each gateways' default routes are assigned weights. As part of the BGP route selection process, local preference is used to determine which route is selected from the BGP tables and entered into the router's route table.

OSPF Default Route Origination

In order to propagate this information into the OSPF backbone, a default route must be distributed based on the results of the route selection process used by BGP. This means that the router which announces the default route to the corporate intranet must speak both BGP and OSPF. This originally caused some consternation with the backbone engineering teams and eventually led to the introduction of the *innie-outie router* (described below).

Routing Implementation

This section references Figure 1 and walks through each component specifying the tasks that should be performed there.

ISP Router (isp-rtr)

BGP processes run on both the Lucent external router and also on the ISP's border router. These two BGP neighbors peer, and due to the differing AS numbers, agree to speak EBGP. The external ISP router advertises, via EBGP, a default route (to the Internet) to the external Lucent router (ext-rtr).

The configuration of this router is the responsibility of our ISP and entails negotiation with their engineering staff for proper setup. Different ISP's are willing to support different configurations. Minor adjustments to the architecture may need to be made to support the requirements of each ISP's infrastructure.

External Router (ext-rtr)

The external router maintains an EBGP peering session with the ISP router and receives a default route from it. It also peers with the innie-outie router (io-rtr) as well. The peering between the innie-outie router and the external router is still BGP, but because both routers are within the same AS, they speak Internal BGP or IBGP.

The external router announces reachability information for all Lucent networks to the Internet. Route information is obtained dynamically from the io-rtr. Because the outbound announcements are being generated from dynamically updated information, a link failure forces a revocation of these routes, causing traffic destined for the internal network to be routed to the next highest preferred gateway.

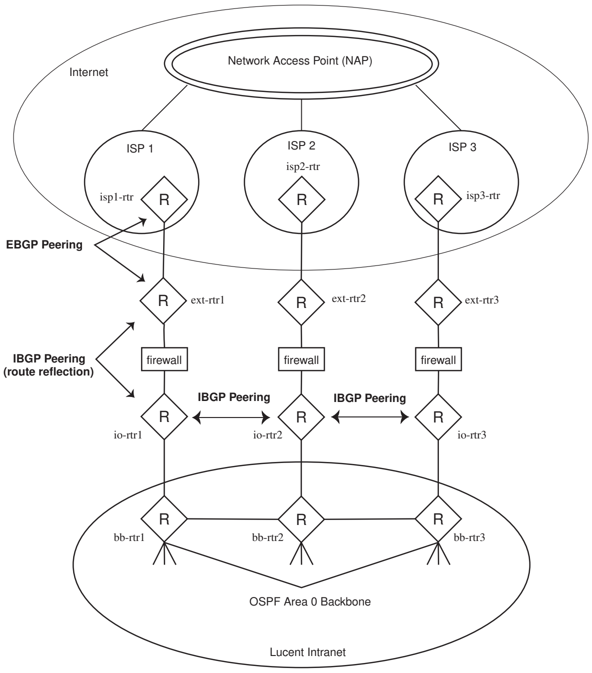

Figure 1: Network overview.

The AS path information is modified to prefer a return path through a particular entry/exit point [Halabi97]. When BGP routers on the Internet receive a packet destined to a Lucent network, they look in their BGP tables for all possible routes. If there are multiple routes in the table for the same network (each firewall location advertises routes for all Lucent networks), a comparison is done between all the entries to select the best route. The list of AS's that a route has traversed is called the AS-path. The shortest AS-path is considered the ''best'' route. By prepending multiple copies of Lucent's AS number, we can propagate multiple routes for the same network, but force traffic through a preferred location by extending the AS-paths of all the other routes. This mechanism allows us to weight preferred return paths back to the Lucent network.

It is important to note, that since routes are advertised and/or revoked by changes that affect the updating of Lucent routes to the external router, any link change may disrupt outgoing route announcements. When a route appears and disappears in rapid succession, this is called flapping. Since route flapping can bring the Internet to its knees, it is highly discouraged, and in most cases flapping routes will be suppressed until they are stable. This could cause a serious service outage, and is to be avoided. This implies extreme care in taking down links or hosts that lie between the ext-rtr and the io-rtr.

To further reduce the risk of flapping routes, the external router will only announce aggregate (or "supernet") routes to the Internet [Fuller93]. This prevents the announcement of superfluous routing information and improves stability.

Firewall

Currently, the existing firewalls are stateful packet-filtering firewalls that act as Layer 2 networking devices (i.e., bridges). This may not always be true, and we want to retain the ability to choose firewalls that operate as Layer 3 networking devices (i.e., routers). Since the firewall component of this architecture may change, it is critical to be able to support either configuration in the routing architecture.

By using IBGP to exchange routing information between the io-rtr and the ext-rtr, there is no need to use a routing protocol on the firewall itself. IBGP allows a peering session to be set up between two routers that are not directly connected, but have IP reachability. If we had to use a routing protocol that mandated peers to be directly connected, the firewall would be required to run a routing protocol and peer with both external and internal routers. This could cause security problems and is not a good security practice.

Although the firewall is not running a routing protocol, it is vital that the firewall (if implemented as a router) have the ability to route information either to the Internet, or to the internal network. This is most easily accomplished by setting the default route on the firewall to the ext-rtr, and installing static routes for all internal networks into the routing table. This is quite a sizeable route table with a network the size of Lucent which leaves us predisposed to bridging firewalls. There is no need for any of the routing hassles on the firewall if it is implemented as a bridge [Limoncelli99].

A direct consequence of advertising all Lucent networks to the Internet is that Network Address Translation does not need to be configured on the firewall, avoiding complex firewall configurations and the woes that accompany them.

Innie-Outie Router (io-rtr or I/O router)

The io-rtr is responsible for several roles in the architecture. It accepts the default route from it's corresponding ext-rtr, and installs it into the BGP table. The BGP process on the io-rtr selects the preferred default route out, and injects that route into the OSPF backbone [Halabi95].

In order to exchange routing information between the ext-rtr and the io-rtr, an IBGP peering session must be established. Recall that IBGP has some requirements on its configuration to prevent route loops and other network inconsistencies. IBGP peers must be configured in a full mesh, but they do not have to be directly connected. When there is no direct connection between peers, it is essential that a route exists within the route table to reach the IBGP peer.

The problem with IBGP peering is that it mandates a full mesh between peers. This would have TCP sessions established between external routers and all other IBGP routers at other locations. This configuration is less than desirable.

By running the io-rtr as a BGP route reflector, it enables a single IBGP session between the external IBGP speaker and the innie-outie router, while the io-rtr's are fully meshed between themselves. [Chandra96] Traditionally, route-reflectors are used in situations where meshing is not feasible due to the number of BGP speakers, but it solves an otherwise untenable problem for this architecture. Route reflection in this application is akin to having one router proxy the IBGP mesh to the hard-to-reach external routers.

Once an IBGP session has been established, the ext-rtr and the io-rtr can exchange routes. Since the ext-rtr is not part of the trusted internal network and the io-rtr is behind the firewall, the io-rtr needs to be configured to accept only a default route from the external Lucent router. This enhances security by providing another layer of route filtering [Raza97].

Route distribution filters can be applied to updates sent or received from specific peers. By employing route filters on received routes, it's trivial to determine the origin of a route (i.e., the ext-rtr, or an io-rtr at another location). The origin can be used as a selector which weights the route to match traffic exit policy (by setting or comparing the local-preference attribute). The I/O router then selects the default route with the highest local-preference and installs it into the routing table. This default route is then injected into the OSPF backbone by the I/O router that is receiving the highest weighted default from its corresponding ext-rtr. At any one time, only one default route is ever injected into the OSPF backbone. This selection process is the core of the primary/backup architecture. The setting of local-preference values strictly determines the primary gateway and also the order in which backup gateways come online in the event of a failure. Proper configuration of the filters and local-preference manipulation is crucial to the correct operation of the architecture.

In order to route traffic back in, the io-rtr must be member of the OSPF Area 0 as an OSPF *Autonomous System Border Router* (ASBR). This allows OSPF to provide a complete list of all internal routes without having to statically define any routes on the io-rtr. These routes must be redistributed into BGP, but only for the benefit of the ext-rtr. Each io-rtr will receive all Lucent routes from a local OSPF neighbor, but will not exchange OSPF-originated routes via BGP to other io-rtr's. This prevents any route loops or other sub-optimal routes from being generated [Rekhter94] [Varadhan92]. The IBGP configuration on the io-rtr will allow the ext-rtr to receive routing updates about Lucent networks. If there is a link outage between the io-rtr and ext-rtr, the ext-rtr stops advertising routes (after they timeout without an update) and the Internet will route traffic back to an alternate firewall location.

There are also web proxy servers, mail servers, and various other Internet servers that are connected to another interface on the io-rtr, which require that traffic always default out the local firewall. By following the BGP selected defaults, all web proxy traffic would be routed out a single firewall. In order to prevent this, policy routing can be used to force all web proxy traffic out the local firewall. Policy routing is used to alter the next hop address based on the packets source IP address (as opposed to classic routing, which determines next hop based on destination address). Any packet originating from these subnets will default route out the local exit point.

Although policy routing handles traffic from these servers to the Internet, it does not force proxy traffic back through the firewall that it went out of. The two ways to ensure that this happens are to advertise the servers' network from only a single gateway, or to perform address translation on the outbound proxy traffic. In order to be consistent with other routing policies, it is preferred to announce routes for networks that will normally be routed out the local gateway, instead of relying on NAT.

DNS Design Philosophy

The philosophy behind the DNS architecture is as below.

1. A proxyless DNS architecture should be as simple as possible, but no simpler.

Similar to the routing issues, as complexity increases so does the likelihood for errors. Again, a design that meets all the requirements for a proxyless routing architecture should be as simple as possible to implement, but robust enough to meet all design criterion.

2. A proxyless DNS architecture should be fault tolerant.

This requirement is also similar to the corresponding routing design goal. Internet access outage at any single point should not cause a significant loss of service for the corporation. Redundancy should exist in all designs to prevent significant outages. Failure of DNS resolution requests pending at the time of a fault is acceptable. All post-fault name lookup requests should automatically proceed to an alternate server.

3. A proxyless DNS architecture should be consistent.

When internal hosts resolve addresses, reverse lookups on that IP address should yield the corresponding name. If masquerading or NAT techniques are employed within any part of the proxyless design, the DNS should provide a consistent view of the network. Additionally, any controls for policy management of e-mail, joint ventures, mergers, or acquisitions should be similarly consistent.

4. A proxyless DNS architecture should handle policy based management of name resolution.

The Lucent/AT&T/NCR tri-vestiture signaled a new era of intranet churn in which the network should be resilient to faults, but adaptable to change. With mergers, acquisitions, and joint ventures happening on a monthly basis, it is critical that any new DNS infrastructure be able to support name resolution for non-Lucent business partners or recent acquisitions as well as Internet name resolution.

This includes the ability for internal hosts to send and receive email through a policy defined mechanism (i.e., through the internal link, or via the Internet). Also provisions for accessing both internally accessible partner sites as well as their Internet sites.

5. A proxyless DNS architecture should be secure.

While it is a necessity to be able to resolve Internet names and addresses, it is not desirable to release internal topology and structural information to the Internet. For example, it's desirable to access *ftp.abc.test*, but it's probably not wise to reveal the internal host as *5ess.source.lucent.com*.

In addition to not releasing internal host and domain names, it is also a risk to accept and trust information from Internet based DNS sources. As direct consequence, care should be taken when obtaining such information, as well as distributing the information internally without validation or screening of some sort.

Problems to be Solved – Part II: DNS

In order for any internal DNS name servers to provide answers to internal queriers, these servers must be able to send and receive queries to Internet name servers. Similar to setting up an intranet, routeability is a precursor to name resolution. For the sake of this discussion we'll presume that sufficient routing exists to facilitate whatever architecture is most appropriate.

The principal problem is convincing our internal top-level name servers (which you may remember are self-rooted) to forward queries for unknown domains to the Internet. At the same time, it extremely important to maintain strict control over both external and internal views of our DNS, as well as using the DNS to control email routing policies. Besides simply providing this functionality, these problems should be addressed in a way which doesn't add any significant additional security risks.

There are many well-known risks of blindly trusting information distributed to the public DNS

[Bellovin95]. Specifically, we want to avoid problems with contamination. Although there are several methods to do this, we were predisposed to some researchware called *dnsproxy* [Cheswick96]. Dnsproxy handles many of the policy and security issues in a unique and elegant way. The specific usage of dnsproxy is described in depth below.

The view we present the Internet of Lucent's internal network also must be carefully designed. Whether NAT or full route announcement is used, the IP addresses that appear on the Internet should resolve to valid hostnames. While the hostnames which are visible to Internet hosts should be valid, there is no requirement that they match the internal hostnames that the IP addresses correspond to. In fact, it is in the interest of security that they should not match the internal address.

The DNS Design

To simplify the DNS design, we decided to not reinvent the wheel and instead capitalize on a design solution from our compatriots in Bell Labs'. There are two principal problems to be solved with DNS, internal name resolution, and external reverse name resolution.

First, the internal DNS needs to be able to resolve Internet hosts. We chose to accomplish this by configuring the pre-existing root nameserver to forward unknown requests to a dnsproxy server. Dnsproxy acts as a switch and filter for DNS requests. Given a query for any resource record type it can switch the set of name servers to query for the correct answer. In addition to it's switching capabilities, it also provides filtering to protect internal queriers from DNS mischief. The dnsproxy source is approximately 4,000 lines of C code. More information on dnsproxy can be found in [Cheswick96].

The typical dnsproxy configuration from research involved determining whether the query should be routed internally, or to Internet name servers. For general corporate use we required some more complicated configuration and eventually further customized the server software. The many joint ventures, mergers, acquisitions, and spinoffs each require specialized DNS handling. Specifically, the treatment of MX records by name servers can get especially insane.

Our default policy is that the corporate email gateways handle all outgoing email. Beyond this, each back-door network connection should have mail either handled over the Internet or through the internal network, depending on the legal agreement.

Custom software again comes to the rescue to help solve the external reverse lookup problems. As stated above, we want to prevent the announcement of the *5ess.source.lucent.com* name to the Internet, but we still want to have functional DNS entries for every host. The way we chose to do this was by creating a new domain in the external DNS, *outland.lucent.com*. Any host that was not explicitly in the external DNS would be given an entry in the *outland.lucent.com* domain. Normally this would be done by creating PTR and A records for each possible internal IP address that pointed to a made-up name. We use the following format: for host 10.2.3.4, the PTR record points to h10-2-3-4.outland.lucent.com, and the A record for h10-2-3-4.outland.lucent.com is 10.2.3.4.

Unfortunately, creating all these records involves maintaining a nameserver with nearly 10 million entries (for all of Lucent's address space). To avoid having to deal with this, it was simpler to write a small piece of software to generate answers dynamically

```
realm
        inside          ns1.lucent.com, ns2.lucent.com, ns3.lucent.com
        outside         ns.isp1.net, ns.isp2.net, ns.isp3.net

switch
        inside  any     bell-labs.com
        inside  any     lucent.com
        inside  any     merger.com
        inside  any     spinoff.com
        inside  any     localhost
        inside  any     135.in-addr.arpa
        inside  any     11.192.in-addr.arpa
        outside any     *

filter  outside block   * NS *
        outside block   * A 127/8
        outside insist  * 28800 MX 100 mailgw1.lucent.com
        outside insist  * 28800 MX 150 mailgw2.lucent.com
        outside insist  * 28800 MX 200 mailgw3.lucent.com
```

Figure 2: Sample dnsproxy.conf.

depending on the question. This server was also written in-house. The design is to have a simple server which creates consistent A and PTR records depending on the question, and delegate the appropriate *in-addr.arpa* domains toward the bogus nameserver. The source code for this server is not available at this time, although its implementation is nearly trivial.

DNS Implementation

The DNS implementation entailed adding two new classes of servers to the Lucent infrastructure. First dnsproxy servers were setup at the backbone. These servers were configured with a configuration file similar to that in Figure 2. Two *realms*, which specify a collection of name servers that can be queried, are defined. One realm is for internal lookups and another for Internet lookups. The *switch* directive determines the realm to query for each domain, based on resource record type. We explicitly tell the dnsproxy to query internal name servers for both our own internal domains and also those that we have special connectivity arrangements with. We also absorb all the in-addr.arpa domains for Lucent's networks, as we (internally) are authoritative for them. All other queries are directed to external name servers.

Filter rules assign actions to perform on certain responses. Our configuration prevents the retrieval of NS records because internal hosts shouldn't be concerned with real Internet name servers. All queries should be directed up through the internal DNS hierarchy and eventually end up at the dnsproxy server. Additionally, we don't accept suspicious addresses from the outside. Finally, all internal queries for MX records are answered by the dnsproxy itself, in the form of an *insist* directive. The insist directive was a customization due to our Internet mail handling policy.

The other major change made to our DNS infrastructure was by modifying the behavior of the root name servers. Under the old architecture, the internal roots were authoritative for everything, period. Now, we are still self-rooted, but the root name servers have delegated most of the top level domains to the dnsproxy server.

Our internal root servers handle name serving for *lucent.com*, and some other second-level domain names. All other top-level domains are delegated to the dnsproxy, as well as the remainder of the *com* domain. This allows us to handle special cases by direct local name serving or internal delegation, while the dnsproxy handles everything else.

Hard Lessons

At this point, it may appear as if everything has been fairly well thought out and that implementation probably went without a hitch. Unfortunately, this is not the case. During the deployment, Lucent's corporate backbone was brought to it's knees no less than three times (during off hours and scheduled change windows of course, but still...)

The largest problem we had was by far the unpredictably of changes made at the Internet perimeter and the backbone on downstream routing domains. Specifically, propagation of default routing information was made quite difficult by the pockets of our intranet still using IGRP. The notion of a default route is a bit different than that of OSPF or BGP.

To work around IGRP's peculiarities, we had to modify the notion of the IGRP gateway of last resort. Originally, the core backbone was announced as the gateway of last resort. The problem with this was that the backbone routers running both IGRP and OSPF would use IGRP's default gateway, which was the directly connected backbone network. This was remedied by pointing the new default network to an upstream, Internet sourced route which caused traffic to be forwarded to the innie-outie routers instead of being dropped at the IGRP backbone router.

We also ran into some issues with the new DNS architecture. The original implementation of dnsproxy allowed for the rewriting of responses for MX queries to our own internal mail gateways. This worked fine except for those sites which didn't have MX records, but did have valid A records. The query would go out, but no MX query would be returned to rewrite, so internal mailers would lookup address record information. The internal mail host would then attempt to connect to the SMTP port of the destination host which would be blocked by the firewall. Thanks to the persistence of most mailers, the message would be deferred and spooled for later delivery. This caused mail spool directories to get quite large for awhile until a workaround was implemented. Additional modifications were required to dnsproxy in order to send back a forged MX response for any query. This caused all outbound mail to be routed to the corporate mail gateways (even to non-existent domains). Poorly addressed mail was simply bounced at the gateway instead of on the sending host.

Another issue that came up with the DNS was the failure of some internal name servers to resolve Internet names. This would happen during the handling of some recursive queries. Since the dnsproxy server filters requests for NS records, internal name servers would be unable to recursively follow delegations and answer the query. This problem was remedied by configuring our internal name servers with *forwarder* directives which pointed unknown queries back to our root servers (which could directly ask the dnsproxy server).

Where To Go From Here

At this point, we have implemented the primary/backup routing architecture and deployed the dnsproxy-based DNS infrastructure. The DNS architecture has served well and isn't likely to change any time soon. On the other hand, the routing design has plenty of room for improvement.

Many of the constraints that led us to the existing routing design were historical, organizational, or otherwise political designs. There has been a realization within Lucent's WAN engineering groups that things need to change in order to manage the network effectively as well as supporting business needs. Listed below are a few of the directions that will improve the efficiency, reliability, and maintainability of our network.

1. Using Full BGP Routing

By receiving the full Internet routing table via BGP, we could route Internet-bound traffic out the best-path gateway. Recall, the reason this wasn't done initially because our existing topology didn't permit the internal BGP peers to be directly connected. Changes described below will soon change this.

BGP will readily determine the best exit-point to any given Internet host, but this is only half of the problem. Also recall that our stateful packet-filtering firewalls generate a requirement for symmetric routing. The difficult part isn't the best-path selection, rather assuring that the best-path to the Internet returns through the same firewall on the return path. Possible solutions to this problem are still in the whiteboard stage.

2. ATM Backbone Infrastructure

The existing backbone infrastructure has been converted to ATM, so we can now provide physical connectivity between our innie-outie routers. Currently the innie-outie routers are connected to the backbone via a fast ethernet switch. By replacing this with an ATM switch and interface, we can configure virtual circuits between the innie-outie routers at the various backbone locations.

3. BGP as the Backbone Routing Protocol

This is being investigated as a possibility to help manage the Internet routing as well as the plethora of mergers and acquisitions that occur regularly. The biggest advantage with respect to Internet routing is that internal routes could be tagged much easier upon redistribution into BGP than into OSPF. This tagging can be used to mark certain routes for specific egress points, which is necessary for symmetric Internet routing. This proposal is still being investigated.

Author Information

D. Brian Larkins is a Member of the Technical Staff at Lucent Technologies, where he is an engineer with the CIO Internet Services and Support Group. His responsibilities include architecture and design for securely connecting the Lucent WAN to the Internet, as well as engineering Lucent's e-commerce infrastructure. Brian began working on web technologies and Internet security in 1994 for AT&T's *www.att.com* site. He holds a B.S. in C.S. from the Ohio State University. Brian can be reached at <brian@lucent.com>.

References

[Bellovin95] Bellovin, S., "Using the Domain Name System for System Break-Ins," *Proceedings of the 5th Usenix Security Symposium*, 1995.

[Chandra96] Chandra, R., "Bates, T., BGP Route Reflection: an Alternative to Full Mesh IBGP," *RFC 1966*, June, 1996.

[Chandra97] Chandra, R., *Introduction to Border Gateway Protocol*, presentation at Cisco Networkers, June, 1997.

[Cheswick96] Cheswick, W., "A DNS Filter and Switch for Packet-filtering Gateways," *Proceedings of the 6th Usenix Security Symposium*, 1996.

[Cheswick94] Cheswick, W., *Firewalls and Internet Security; Repelling the Wily Hacker*, Addison-Wesley, 1994.

[Fuller93] Fuller, V., Li, T., Yu, J., Varadhan, K., "Classless Interdomain Routing: an Address Assignment and Aggregation Strategy," *RFC 1519*, September, 1993.

[Halabi97] Halabi, Bassam, *Internet Routing Architectures*, Cisco Press, Indianapolis, IN, 1997.

[Halabi95] Halabi, Bassam, *BGP4 Case Studies/Tutorial*, Cisco Systems, 1995.

[Huitema95] Huitema, Christian, *Routing in the Internet*, Prentice Hall, Englewood Cliffs, N.J., 1995.

[Khan97] Khan, A., *Designing and Troubleshooting OSPF Networks*, presentation at Cisco Networkers, June 1997.

[Limoncelli99] Limoncelli, T., "Tricks you can do if your Firewall is a Bridge," *Proceedings of the 1st Usenix Conference on Network Administration*, 1999.

[Moy94] Moy, John., "OSPF Version 2," *RFC 1583*, March, 1994.

[Raza97] Raza, K., *Configuring BGP Networks*, presentation at Cisco Networkers, June, 1997.

[Rekhter95a] Rekhter, Y., Gross, P. (editors), "Application of the Border Gateway Protocol in the Internet," *RFC 1772*, March, 1995.

[Rekhter95b] Rekhter, I., Li, T., "A Border Gateway Protocol (BGP-4)," *RFC 1771*, March, 1995.

[Rekhter94] Rekhter, Y., Gross, P., "BGP4/IDRP for IP – OSPF Interaction," *RFC 1745*, December, 1994

[Stevens94] Stevens, W. Richard., *TCP/IP Illustrated, Vol. 1*, Addison Wesley, Reading, MA, 1994.

[Umali97] Umali, T. (editor), *Lucent Technologies IP Data Network Architecture Document*, Internal Memorandum, 1997.

[Umali96] Umali, T. (editor), *Lucent Technologies IP Routing Implementation Document,* Internal Memorandum, 1996.

[Varadhan92] Varadhan, K., "BGP OSPF Interaction," *RFC 1364*, September 1992.