DNS and IPv6

By Alex Lightman
CEO, Innofone.com, Inc.

DNS, or the Domain Naming System, has been around for many years. The existing Internet would not be able to scale by adding nodes through people with such varying skills without it. The basic function of DNS is to map hierarchical domain names (e.g. www.usipv6.com) onto IP addresses (e.g. 123.45.67.89, for IPv4), which is what is actually used in packet headers for addressing on the wire (See Figure 1). You can think of DNS as the Internet's (automated) telephone book, as a start.

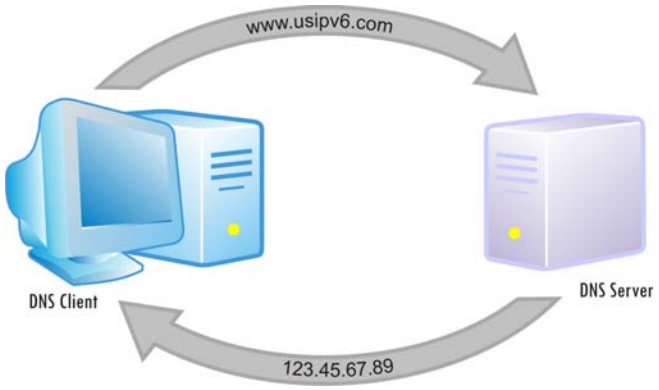

Figure 1 – Domain Naming System

DNS also does IP address to domain name mapping (like a "reverse phone book," available to business users), and allows you to publish certain things that must be known to others, such as the nodename of your preferred mail server(s), which is done with MX records. Few people realize that you can also advertise the preferred LDAP server for your domain name and other such things, or use it to map a universal telephone number to one or more URLs (using ENUM), for e-mail, IM, VoIP with SIP, etc.

There are a number of widely used DNS servers, the most popular of which is a public domain program originally developed at UC Berkeley, known as the Berkeley Internet Naming Daemon, or BIND. Microsoft also created their own, now included with Windows Server. Other folks have created free ones and commercial ones (e.g. Nominum, headed by Paul Mockapetris, who invented DNS).

Every network has to have a DNS server, or else the nodes in your network can only be addressed by others using numeric IP addresses. In a purely Microsoft network (or other SMB/CIFS based system such as SAMBA), it is possible to instead deploy WINS (Windows Internet Naming System), which is used to automatically register and publish "NetBIOS" names (an alternative namespace that can map onto various network addressing schemes, including TCP/IP). However, this system does not scale very well beyond the LAN, and is not used by most widely used Internet applications, such as e-mail, ftp, telnet, or web.

Today, even Microsoft recommends you use DNS even for SMB based network operations, and not deploy WINS.

IPv4 uses 32-bit addresses, and “dotted decimal” notation (with four groups of 8 bits, each represented with decimal values from 0 to 255). It is still practical, if annoying, for users to specify other nodes using numeric addresses. Even there, however, DNS vastly simplifies the task and makes it possible for less technical users to use the network.

There are actually two naming contexts commonly found in networks: LAN and Internet. For servers that need only be accessible from within your LAN (e.g., your Windows Server Domain Controllers or File/Print servers) many people just enter those few nodes on internal DNS (or even WINS) servers, using internal addresses (e.g., from RFC 1918, like 10.x.x.x/8, 172.16.x.x/16 or 192.168.x.x/24). If you have some servers that must be accessible from outside your LAN (e.g. mail or web servers), then those nodes must be published in an externally accessible DNS server (often running at your ISP or Registrar), using valid (routable) external static addresses (unique on the Internet).

A more experienced network administrator will enter the name and address of *every* node on your LAN into the internal DNS. Some administrators may deploy one (or a redundant pair of) DNS servers for external users (with external addresses) in your DMZ, and a separate server (or pair of servers) for internal users (with internal addresses) in your internal network segment. A very experienced one will publish both internal addresses for use by internal nodes, and external addresses for use by external nodes, using multiple BIND “views,” on a single DNS server (or redundant pair of servers), accessible from both inside and outside.

Many administrators are familiar with supporting “forward” references (mappings from domain name to IP address, as in “What is the IP address of the node with name www.innofone.com?”). Fewer are familiar with supporting “reverse” references (mappings from IP address to domain name, as in “What is the node name whose IP address is 123.45.67.89?”).

The first type of reference (forward) is done by creating “forward zones” that contain A or AAAA records. The other (reverse) reference has “reverse zones” that contain PTR records. In a properly designed network, all nodes should have both forward and reverse mappings defined in DNS servers. Such networks work far better than ones with only forward zones. In fact, for mail servers, many servers today will routinely try to reverse map the IP address a connection comes from, and if they can’t, or the mapping does not match the nodename it claims to be connecting from, the assumption is the transmission is from a bogus server (hacker, zombie, spammer, etc.), and the connection is refused. If you deploy an e-mail server, you need to be sure both forward and reverse DNS mappings work right, or some of your outgoing mail will be rejected.

So far, everything we’ve discussed is equally applicable to IPv4 and IPv6.

What is different in IPv6?

Well, for one thing, while some people might be willing to type the occasional dotted decimal IPv4 address, few are willing to (or able to correctly) type a longer and more complicated IPv6 address. In IPv6 it is more important than ever to have a correctly configured DNS server that includes all nodes, i.e., is "comprehensive." All nodes that might possibly be accessed from outside your LAN (which in IPv6 could be *all* nodes – end-to-end connectivity now being restored) should be entered in the externally accessible DNS. If you want any VoIP phone in the world to be able to connect to your VoIP phone, you need to publish its address in an externally accessibly DNS server (possibly even using the DNS ENUM feature).

In IPv4, a forward (A) record, might look like this:

```
www     IN      A       123.45.67.89
```

The corresponding reverse (PTR) record might look like this:

```
89.67.45.123.in-addr.arpa       PTR     www.whazzamattau.edu.
```

An abbreviated form might use the BIND $origin feature to simplify entry (especially if there are a lot of reverse records all sharing the same suffix, where the origin need only be specified once):

```
$origin 45.123.in-addr.arpa
89.67   PTR     www.whazzamattau.edu.
```

Note that the 8-bit fields (octets) are written in reverse order, and the magic phrase "in-addr.arpa" is appended.

In IPv6. a forward (AAAA) record might look like this:

```
www     IN      AAAA    2001:ec8:4008::1234:5678:9abc:def0
```

The corresponding PTR record might look like this (using the $origin feature): (see Figure 2)

```
$origin 8.40.c8.e.1.20.ipv6.arpa
f0.de.bc.9a.78.56.34.12.0.0     PTR     www.whazzamattau.edu.
```

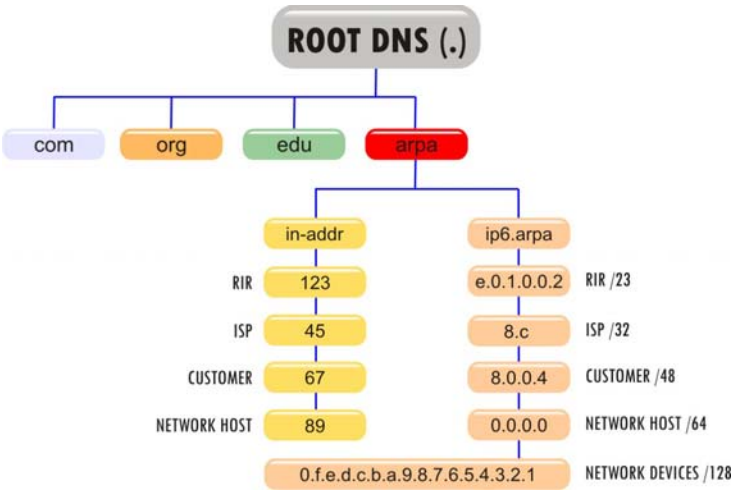

Figure 2 - DNS Reverse Mapping Tree

If you've seen the older ".ipv6.int" suffix, it is now deprecated and should not be used. Note that each 4-hex digit (16-bit) field is split into two 2-hex digit (8-bit) fields, and again the whole thing is written in reverse byte order. IPv6 PTR records are notoriously difficult to get right the first time in a manually administered BIND server.

If all this looks complicated, it is. *Which brings me to the point of this column*. I've recently seen one of the first commercial IPv6 infrastructure appliances, called SolidDNS (from InfoWeapons Corporation in Asia, where, as I've pointed out in dozens of previous columns for 6Sense, IPv6 deployment is ahead of that in the US). At heart, there is still a copy of BIND running (albeit in native 64-bit mode on an AMD Opteron). However, that is where the similarity to how most people are deploying IPv6 DNS ends. They have embedded BIND in an appliance with military grade security (using lots of DARPA technology, and meeting or exceeding all DoD security standards, they claim). It also has a very intuitive GUI interface (actually two GUI interfaces, which we'll get to in a minute). The developers have a world-class passion for IPv6, and it's their focus.

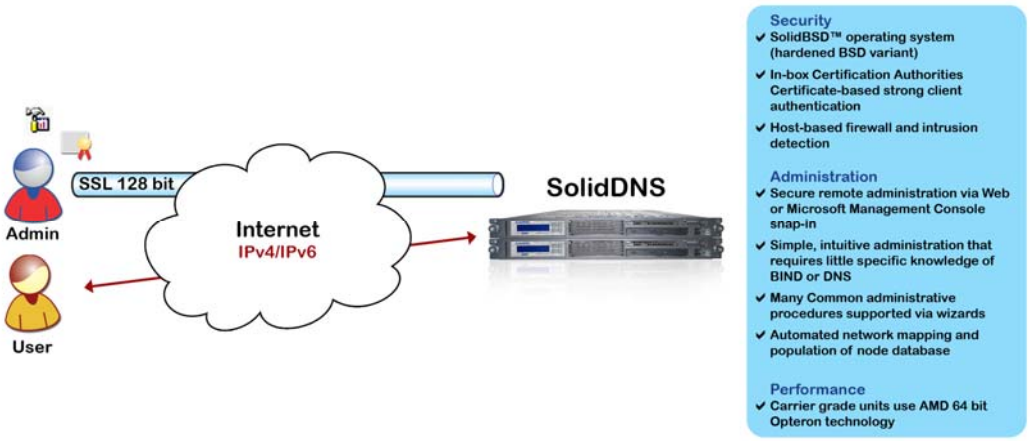

Figure 3 – InfoWeapons SolidDNS Appliance

You can quickly specify your domains and networks (with prefix length), then enter each node and its corresponding address. When you click the “restart” button, SolidDNS generates new, syntactically perfect forward and reverse zone files, with configuration guaranteed to be correct (all glue records, etc.). You can even import old BIND zone files, if you have them.

If it only supported IPv4, this would already be useful, but InfoWeapons has made it unusually easy to enter and manage IPv6 addresses, and generate syntactically perfect forward and reverse zones for IPv6 as well (some people may say that if they never have to create another IPv6 BIND reverse zone by hand it will be too soon).

Realizing that entry of IPv6 addresses is time-consuming, difficult and error-prone, they have allowed you to define “network names,” with a given (typically 64-bit) prefix. When entering nodes, you can choose any defined network name from a pull-down list, then enter just the low 64 bits. The truly amazing thing is that each address remembers the network name it was defined with. You can go back in later and redefine the prefix associated with that network name, and regenerate all forward and reverse zone files using the new prefix. This accomplishes what the (now deprecated) A6 DNS records were supposed to do, with *no downside*. The generated BIND zone files contain perfectly normal AAAA records. With this appliance you can do prefix renumbering in a matter of seconds, which could easily take weeks using BIND directly for a large site.

Dynamic DNS (having DHCP automatically insert DNS records) is somewhat of a problem for an appliance that generates new configuration files from a database, as normally the new data is inserted into the BIND configuration files (which get overwritten). SolidDNS intercepts the dynamic registrations (from a DHCP or other source), adds them into the database and updates the GUI on the fly. It is really cool to see new nodes and their corresponding addresses magically appear in the GUI. And of course, this works with both IPv4 and IPv6.

All administration can also be done over IPv4 or IPv6, using TLS with optional strong client authentication (using X.509 client digital certificates generated on the box or from any external PKI). As with many other appliances, there is a Web-based admin tool (created with PHP for higher security than products using Java-based admin tools). But they have provided a *second* interface as a Microsoft Management Console “snap-in,” which looks and acts amazingly like the admin tool included with Microsoft DNS server, but is actually managing BIND running on an incredibly hardened UNIX. Even inexpensive MCSEs can learn to use this interface in a matter of minutes, and build composite admin tools that include other snap-ins along with the InfoWeapons one.

This admin tool also works over TLS with strong client authentication, and supports the “network name” shortcuts and renumbering. Some users will prefer the web/PHP interface and some will prefer the MMC snap-in. You can even

switch back and forth if you like. The two interfaces come together in the appliance's internal database.

When you start to migrate from an IPv4 to a dual-stack network, you quickly discover that you must deploy a dual-stack DNS server, and populate it with all of your nodes and their addresses. Roughly 30% of the effort of migrating a network to IPv6 is involved in this process. It appears as thought InfoWeapon's SolidDNS appliance could potentially save a great deal of this time, and insure that the resulting configuration will work very well every time. Some 70% of DNS servers on the IPv4 Internet are misconfigured today – what percent of dual stack DNS servers done using BIND directly do you think will be misconfigured?

It is encouraging to see real (and amazingly mature) IPv6-compliant products starting to hit the market. These products are the "supply" in the supply and demand equation, and can only help to accelerate the adoption of IPv6 by organizations at all levels. US companies should get such DNS boxes in-house to start climbing up the IPv6 learning curve. It looks like a company from the Philippines has released a product that makes its US competitors look like Model T Fords.

For details, see www.infoweapons.com (or www.infoweapons.net for a site that will accept connections over IPv4 or IPv6, and even show you what address you are connecting from). The latter site's domain is managed on a pair of its own SolidDNS appliances.

If it takes 10 steps to accomplish something, the reason for the first step is to get to the second step, and so forth. DNS is so basic that acquiring a box like that of InfoWeapons is almost a reality check for whether a company is actually implementing IPv6, or still in the thinking but no action phase. I encourage our readers to consider one of these boxes and to let me know whether and how it accelerated and made more "real" your IPv6 efforts.

Innofone recently signed a distributor agreement with InfoWeapons, and has boxes for live demonstration at its offices in Fairfax, VA and in Santa Monica, CA.

Please write me at alex@usipv6.com if you have a product that can accelerate IPv6 adoption, or have put one to work in your operations. Thank you.