Network Working Group
Request for Comments: 4183
Category: Informational

E. Warnicke
Cisco Systems

September 2005

A Suggested Scheme for DNS Resolution of Networks and Gateways

Status of This Memo

Copyright Notice

IESG Note

Abstract

This document suggests a method of using DNS to determine the network that contains a specified IP address, the netmask of that network, and the address(es) of first-hop routers(s) on that network. This method supports variable-length subnet masks, delegation of subnets on non-octet boundaries, and multiple routers per subnet.

1. Introduction

As a variety of new devices are introduced to the network, many of them not traditional workstations or routers, there are requirements that the first-hop router provide some network service for a host. It may be necessary for a third-party server in the network to request some service related to the host from the first-hop router(s) for that host. It would be useful to have a standard mechanism for such a third-party device to find the first-hop router(s) for that host.

DNS-based mechanisms have been defined for the resolution of router addresses for classful networks (RFC 1035 [1]) and of subnets (RFC 1101 [2]). RFC 1101 suffers from a number of defects, chief among

which are that it does not support variable-length subnet masks, which are commonly deployed in the Internet. The present document defines DNS-based mechanisms to cure these defects.

Since the writing of RFC 1101, DNS mechanisms for dealing with classless networks have been defined, for example, RFC 2317 [3]. This document describes a mechanism that uses notation similar to that of RFC 2317 to specify a series of PTR records enumerating the subnets of a given network in the RFC 2317 notation. This lookup process continues until the contents of the PTR records are not an in-addr.arpa.-derived domain name. These terminal PTR record values are treated as the hostname(s) of the router(s) on that network. This RFC also specifies an extension to the method of RFC 2317 to support delegation at non-octet boundaries.

2. Generic Format of a Network Domain Name

Using the Augmented BNF of RFC 2234 [4], we can describe a generic domain name for a network as follows:

```
networkdomainname = maskedoctet "." *( decimaloctet / maskedoctet
".") "in-addr.arpa."
maskedoctet = decimaloctet "-" mask
mask = 1*2DIGIT ; representing a decimal integer value in the
                ; range 1-32
decimaloctet = 1*3DIGIT ; representing a decimal integer value in
                        ; the range 0 through 255
```

By way of reference, an IPv4 CIDR notation network address would be written

```
IPv4CIDR = decimaloctet "." decimaloctet "." decimaloctet "."
decimaloctet "/" mask
```

A "-" is used as a delimiter in a maskedoctet instead of a "/" as in RFC 2317 out of concern about compatibility with existing DNS servers, many of which do not consider "/" to be a valid character in a hostname.

3. Non-Octet Boundary Delegation

In RFC 2317, there is no mechanism for non-octet boundary delegation. Networks would be represented as being part of the domain of the next octet.

Examples:

```
10.100.2.0/26  -> 0-26.2.100.10.in-addr.arpa.
10.20.128.0/23 -> 128-23.20.10.in-addr.arpa.
10.192.0.0/13 -> 192-13.10.in-addr.arpa.
```

In the event that the entity subnetting does not actually own the network being subnetted on an octet break, a mechanism needs to be available to allow for the specification of those subnets. The mechanism is to allow the use of maskedoctet labels as delegation shims.

For example, consider an entity A that controls a network 10.1.0.0/16. Entity A delegates to entity B the network 10.1.0.0/18. In order to avoid having to update entries for entity B whenever entity B updates subnetting, entity A delegates the 0-18.1.10.in-addr.arpa domain (with an NS record in A's DNS tables as usual) to entity B. Entity B then subnets off 10.1.0.0/25. It would provide a domain name for this network of 0-25.0.0-18.1.10.in-addr.arpa (in B's DNS tables).

In order to speak about the non-octet boundary case more easily, it is useful to define a few terms.

Network domain names that do not contain any maskedoctets after the first (leftmost) label are hereafter referred to as canonical domain names for that network. 0-25.0.1.10.in-addr.arpa. is the canonical domain name for the network 10.1.0.0/25.

Network domain names that do contain maskedoctet labels after the first (leftmost) label can be reduced to a canonical domain name by dropping all maskedoctet labels after the first (leftmost) label. They are said to be reducible to the canonical network domain name. So for example 0-25.0.0-18.1.10.in-addr.arpa. is reducible to 0-25.0.1.10.in-addr.arpa. Note that a network domain name represents the same network as the canonical domain name to which it can be reduced.

4. Lookup Procedure for a Network Given an IP Address

4.1. Procedure

1. Take the initial IP address x.y.z.w and create a candidate network by assuming a 24-bit subnet mask. Thus, the initial candidate network is x.y.z.0/24.

2. Given a candidate network of the form x.y.z.n/m create an in-addr.arpa candidate domain name:

1. If the number of mask bits m is greater than or equal to 24 but less than or equal to 32, then the candidate domain name is n-m.z.y.x.in-addr.arpa.

2. If the number of mask bits m is greater than or equal to 16 but less than 24, then the candidate domain name is z-m.y.x.in-addr.arpa.

3. If the number of mask bits m is greater than or equal to 8 but less than 16, then the candidate domain name is y-m.x.in-addr.arpa.

4. The notion of fewer than 8 mask bits is not reasonable.

3. Perform a DNS lookup for a PTR record for the candidate domain name.

4. If the PTR records returned from looking up the candidate domain name are of the form of a domain name for a network as defined previously (Section 2), then for each PTR record reduce that returned domain name to the canonical form p1-q1.z1.y1.x1.in-addr.arpa. This represents a network x1.y1.z1.p1/q1.

   1. If one of the x1.y1.z1.p1/q1 subnets contains the original IP address x.y.z.w, then the PTR record return becomes the new candidate domain name. Repeat steps 3-4.

   2. If none of the x1.y1.z1.p1/q1 subnets contain the original IP address x.y.z.w, then this process has failed.

5. If the PTR record(s) for the candidate network is not of the form of a network domain name, then they are presumed to be the hostname(s) of the gateway(s) for the subnet being resolved.

6. If the PTR lookup fails (no PTR records are returned).

   1. If no candidate network PTR lookup for this IP address has succeeded in the past and the netmask for the last candidate network was 24 or 16 bits long, then presume a netmask of 8 fewer bits for the candidate network and repeat steps 2-4.

   2. If no candidate network PTR lookup for this IP address has succeeded in the past and the netmask of the last candidate network was not 24 or 16 bits long, then increase the netmask by 1 bit and repeat steps 2-4.

3. If a candidate network PTR lookup for this IP address has succeeded in the past or the netmask of the last candidate network was 32 bits, then this process has failed.

7. Perform a DNS A record lookup for the domain name of the gateway to determine the IP number of the gateway.

4.2. IPv6 Support

RFC 3513 [5] requires all IPv6 unicast addresses that do not begin with binary 000 have a 64-bit interface ID. From the point of view of identifying the last hop router for an IPv6 unicast address, this means that almost all hosts may be considered to live on a /64 subnet. Given the requirement that for any subnet there must be an anycast address for the routers on that subnet, the process described for IPv4 in this document can just as easily be achieved by querying the anycast address via SNMP. Therefore, this document does not speak to providing a DNS-based mechanism for IPv6.

4.3. Example

Imagine we begin with the IP number 10.15.162.3.

1. Form a candidate network of 10.15.162.0/24.

2. Form a domain name 0-24.162.15.10.in-addr.arpa.

3. Look up the PTR records for 0-24.162.15.10.in-addr.arpa.

4. Suppose the lookup fails (no PTR records returned), then

5. Form a new candidate network 10.15.0.0/16.

6. Form a domain name 0-16.15.10.in-addr.arpa.

7. Look up the PTR records for 0-16.15.10.in-addr.arpa.

8. Lookup returns:
    1. 0-17.15.10.in-addr.arpa.
    2. 128-18.15.10.in-addr.arpa.
    3. 192-18.15.10.in-addr.arpa.

9. So 10.15.0.0/16 is subnetted into 10.15.0.0/17, 10.15.128.0/18, and 10.15.192.0/18.

10. Since 10.15.162.3 is in 10.15.128.0/18, the new candidate domain name is 128-18.15.10.in-addr.arpa.

11. Look up the PTR records for 128-18.15.10.in-addr.arpa.

12. Lookup returns
    1. 128-19.128-18.15.10.in-addr.arpa.
    2. 0-25.160.128-18.15.10.in-addr.arpa.
    3. 128-25.160.128-18.15.10.in-addr.arpa.
    4. 0-24.161.128-18.15.10.in-addr.arpa.
    5. 162-23.128-18.15.10.in-addr.arpa.

13. The canonical network domains for these returned records are
    1. 128-19.15.10.in-addr.arpa.
    2. 0-25.160.15.10.in-addr.arpa.
    3. 128-25.160.15.10.in-addr.arpa.
    4. 0-24.161.15.10.in-addr.arpa.
    5. 162-23.15.10.in-addr.arpa.

14. So the network 10.15.128.0/18 is subnetted into 10.15.128.0/19, 10.15.160.0/25, 10.15.160.128/25, 10.15.161.0/25, 10.15.162.0/23.

15. Since 10.15.162.3 is in 10.15.162.0/23, the new candidate domain name is 162-23.128-18.15.10.in-addr.arpa.

16. Look up the PTR records for 162-23.128-18.15.10.in-addr.arpa.

17. Lookup returns:
    1. gw1.example.net.
    2. gw2.example.net.

18. Look up the A records for gw1.example.net. and gw2.example.net.

19. Lookup returns
    1. gw1.example.net: 10.15.162.1
    2. gw2.example.net: 10.15.162.2

So the 10.15.162.3 is in network 10.15.162.0/23, which has gateways 10.15.162.1 and 10.15.162.2.

5. Needed DNS Entries

The example of the lookup procedure (Section 4.3) would require DNS records as follows:

In entity A's DNS zone files:

```
0-16.15.10.in-addr.arpa.  IN  PTR 0-17.15.10.in-addr.arpa.
0-16.15.10.in-addr.arpa.  IN  PTR 128-18.15.10.in-addr.arpa.
0-16.15.10.in-addr.arpa.  IN  PTR 192-18.15.10.in-addr.arpa.
0-17.15.10.in-addr.arpa.  IN  NS ns1.example.org
128-18.15.10.in-addr.arpa.  IN  NS ns1.example.net
192-18.15.10.in-addr.arpa.  IN  NS ns1.example.com
ns1.example.net           IN  A  10.15.0.50
ns1.example.org           IN  A  10.15.128.50
ns1.example.com           IN  A  10.15.192.50
```

In entity B's DNS zone files:

```
128-18.15.10.in-addr.arpa.  IN  PTR
128-19.128-18.15.10.in-addr.arpa.
128-18.15.10.in-addr.arpa.  IN  PTR
0-25.160.128-18.15.10.in-addr.arpa.
128-18.15.10.in-addr.arpa.  IN  PTR
128-25.160.128-18.15.10.in-addr.arpa.
128-18.15.10.in-addr.arpa.  IN  PTR
0-24.161.128-18.15.10.in-addr.arpa.
128-18.15.10.in-addr.arpa.  IN  PTR
162-23.128-18.15.10.in-addr.arpa.
162-23.128-18.15.10.in-addr.arpa.  IN  PTR gw1.example.net.
162-23.128-18.15.10.in-addr.arpa.  IN  PTR gw2.example.net.
gw1.example.net.  IN  A 10.15.162.1
gw2.example.net.  IN  A 10.15.162.2
```

6. Alternate Domain Suffix

Proper functioning of this method may required the cooperation of upstream network providers. Not all upstream network providers may wish to implement this method. If an upstream provider does not wish to implement this method, the method may still be used with an alternate domain suffix.

For example, if the upstream network provider of example.com did not wish to provide glue records in its branch of the in-addr.arpa. domain, then example.com might elect to use the suffix in-addr.example.com as an alternate domain suffix for that purpose.

For this reason, implementations of clients intending to use this method should use in-addr.arpa. as the default suffix, but allow for configuration of an alternate suffix.

7. Security Considerations

Any revelation of information to the public internet about the internal structure of your network may make it easier for nefarious persons to mount diverse attacks upon a network. Consequently, care should be exercised in deciding which (if any) of the DNS resource records described in this document should be made visible to the public internet.

8. Informative References

[1] Mockapetris, P., "Domain Names - Implementation and Specficication", STD 13, RFC 1035, November 1987.

[2] Mockapetris, P., "DNS Encoding of Network Names and Other Types", RFC 1101, April 1989.

[3] Eidnes, H., de Groot, G., and P. Vixie, "Classless IN-ADDR.ARPA delegation", RFC 2317, March 1998.

[4] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[5] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

[6] Alvestrand, H., "The IESG and RFC Editor Documents: Procedures", BCP 92, RFC 3932, October 2004.

Author’s Address

Edward A. Warnicke
Cisco Systems Inc.
12515 Research Blvd., Building 4
Austin, TX 78759
USA

Phone: (919) 392-8489
EMail: eaw@cisco.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.