Network Working Group
Internet-Draft
Expires: January 16, 2005

S. Woolf
Internet Systems Consortium, Inc.
D. Conrad
Nominum, Inc.
July 18, 2004

# Identifying an Authoritative Name `Server draft-ietf-dnsop-serverid-02

## Status of this Memo



## Copyright Notice



## Abstract

With the increased use of DNS anycast, load balancing, and other mechanisms allowing more than one DNS name server to share a single IP address, it is sometimes difficult to tell which of a pool of name servers has answered a particular query. A standardized mechanism to determine the identity of a name server responding to a particular query would be useful, particularly as a diagnostic aid. Existing ad

hoc mechanisms for addressing this concern are not adequate. This document attempts to describe the common ad hoc solution to this problem, including its advantages and disadvantasges, and to characterize an improved mechanism.

## 1. Introduction

With the increased use of DNS anycast, load balancing, and other mechanisms allowing more than one DNS name server to share a single IP address, it is sometimes difficult to tell which of a pool of name servers has answered a particular query. A standardized mechanism to determine the identity of a name server responding to a particular query would be useful, particularly as a diagnostic aid.

Unfortunately, existing ad-hoc mechanisms for providing such identification have some shortcomings, not the least of which is the lack of prior analysis of exactly how such a mechanism should be designed and deployed. This document describes the existing convention used in one widely deployed implementation of the DNS protocol and discusses requirements for an improved solution to the problem.

## 2. Rationale

Identifying which name server is responding to queries is often useful, particularly in attempting to diagnose name server difficulties. However, relying on the IP address of the name server has become more problematic due the deployment of various load balancing solutions, including the use of shared unicast addresses as documented in [RFC3258].

An unfortunate side effect of these load balancing solutions is that traditional methods of determining which server is responding can be unreliable. Specifically, non-DNS methods such as ICMP ping, TCP connections, or non-DNS UDP packets (e.g., as generated by tools such as "traceroute"), etc., can end up going to a different server than that which receives the DNS queries.

The widespread use of the existing convention suggests a need for a documented, interoperable means of querying the identity of a nameserver that may be part of an anycast or load-balancing cluster. At the same time, however, it also has some drawbacks that argue against standardizing it as it's been practiced so far.

## 3. Existing Conventions

Recent versions of the commonly deployed Berkeley Internet Name Domain implementation of the DNS protocol suite from the Internet Software Consortium [BIND] support a way of identifying a particular server via the use of a standard, if somewhat unusual, DNS query. Specifically, a query to a late model BIND server for a TXT resource record in class 3 (CHAOS) for the domain name "HOSTNAME.BIND." will return a string that can be configured by the name server administrator to provide a unique identifier for the responding server (defaulting to the value of a gethostname() call). This mechanism, which is an extension of the BIND convention of using CHAOS class TXT RR queries to sub-domains of the "BIND." domain for version information, has been copied by several name server vendors.

For reference, the other well-known name used by recent versions of BIND within the CHAOS class "BIND." domain is "VERSION.BIND." A query for a TXT RR for this name will return an administratively re-definable string which defaults to the version of the server responding.

### 3.1 Advantages

There are several valuable attributes to this mechanism, which account for its usefulness.

1. This mechanism is within the DNS protocol itself. An identification mechanism that relies on the DNS protocol is more likely to be successful (although not guaranteed) in going to the same machine as a "normal" DNS query.
2. It is simple to configure. An administrator can easily turn on this feature and control the results of the relevant query.
3. It allows the administrator complete control of what information is given out in the response, minimizing passive leakage of implementation or configuration details. Such details are often considered sensitive by infrastructure operators.

### 3.2 Disadvantages

At the same time, there are some forbidding drawbacks to the VERSION.BIND mechanism that argue against standardizing it as it currently operates.

1. It requires an additional query to correlate between the answer to a DNS query under normal conditions and the supposed identity of the server receiving the query. There are a number of situations in which this simply isn't reliable.
2. It reserves an entire class in the DNS (CHAOS) for what amounts to one zone. While CHAOS class is defined in [RFC1034] and [RFC1035], it's not clear that supporting it solely for this

purpose is a good use of the namespace or of implementation effort.

3. It is implementation specific. BIND is one DNS implementation. At the time of this writing, it is probably the most prevalent, for authoritative servers anyway. This does not justify standardizing on its ad hoc solution to a problem shared across many operators and implementors.

The first of the listed disadvantages is technically the most serious. It argues for an attempt to design a good answer to the problem that "I need to know what nameserver is answering my queries", not simply a convenient one.

## 4. Characteristics of an Implementation Neutral Convention

The discussion above of advantages and disadvantages to the HOSTNAME.BIND mechanism suggest some requirements for a better solution to the server identification problem. These are summarized here as guidelines for any effort to provide appropriate protocol extensions:

1. The mechanism adopted MUST be in-band for the DNS protocol. That is, it needs to allow the query for the server's identifying information to be part of a normal, operational query. It SHOULD also permit a separate, dedicated query for the server's identifying information.
2. The new mechanism should not require dedicated namespaces or other reserved values outside of the existing protocol mechanisms for these, i.e. the OPT pseudo-RR.
3. Support for the identification functionality SHOULD be easy to implement and easy to enable. It MUST be easy to disable and SHOULD lend itself to access controls on who can query for it.
4. It should be possible to return a unique identifier for a server without requiring the exposure of information that may be non-public and considered sensitive by the operator, such as a hostname or unicast IP address maintained for administrative purposes.
5. The identification mechanism SHOULD NOT be implementation-specific.

## 5. IANA Considerations

This document proposes no specific IANA action. Protocol extensions, if any, to meet the requirements described are out of scope for this document. Should such extensions be specified and adopted by normal IETF process, the specification will include appropriate guidance to IANA.

## 6. Security Considerations

Providing identifying information as to which server is responding can be seen as information leakage and thus a security risk. This motivates the suggestion above that a new mechanism for server identification allow the administrator to disable the functionality altogether or partially restrict availability of the data. It also suggests that the serverid data should not be readily correlated with a hostname or unicast IP address that may be considered private to the nameserver operator's management infrastructure.

Propagation of protocol or service meta-data can sometimes expose the application to denial of service or other attack. As DNS is a critically important infrastructure service for the production Internet, extra care needs to be taken against this risk for designers, implementors, and operators of a new mechanism for server identification.

## 7. Acknowledgements

The technique for host identification documented here was initially implemented by Paul Vixie of the Internet Software Consortium in the Berkeley Internet Name Daemon package. Comments and questions on earlier drafts were provided by Bob Halley, Brian Wellington, Andreas Gustafsson, Ted Hardie, Chris Yarnell, Randy Bush, and members of the ICANN Root Server System Advisory Committee. The newest draft takes a significantly different direction from previous versions, owing to discussion among contributors to the DNSOP working group and others, particularly Olafur Gudmundsson, Ed Lewis, Bill Manning, Sam Weiler, and Rob Austein.

## Intellectual Property Statement



## Disclaimer of Validity



## Copyright Statement



## Acknowledgment

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