Stuart Cheshire
Document: draft-cheshire-dnsext-multicastdns-00.txt
Apple Computer
Expires 13th January 2002
13th July 2001

Performing DNS queries via IP Multicast

<draft-cheshire-dnsext-multicastdns-00.txt>

Status of this Memo

Abstract

Multicast DNS is a really obvious idea, whose time has finally come. This draft proposes one possible way of making it work.

1. Acknowledgements

This work builds upon original work done on Multicast DNS by Bill Manning and Bill Woodcock. The authors gratefully acknowledge their contribution to the current specification. Other contributors of valuable ideas include Bernard Aboba, Mark Andrews, Randy Bush, Levon Esibov, James Gilroy, Olafur Gudmundsson, Erik Guttman, Myron Hattig, Thomas Narten, Erik Nordmark and Dave Thaler.

I apologize humbly to anyone who feels their work has not been properly credited and I offer to buy dinner or drinks in compensation.

2. Introduction

This is a rough first draft. Its purpose is to describe the proposed idea well enough for meaningful discussion to take place. As such, while feedback concerning typographical mistakes and similar minutiae is always appreciated, the reader is advised that it is probably unwise to waste a lot of time on such trivia until after we find out whether this proposal will even live long enough to become a 'draft-01'.

When reading this document, familiarity with the concepts of Zero Configuration Networking [ZC] and automatic link-local addressing [v4LL] [RFC 2462] is helpful.

This document proposes no change to the structure of DNS messages, and no new operation codes, response codes, resource record types, or any other new DNS protocol values. This document simply discusses what needs to happen if DNS clients start sending DNS requests to a multicast address.

The primary difference between this document and "draft-ietf-dnsext-mdns-01.txt" is the philosophy about how subdomains of the "local.arpa." domain are delegated. That document proposes that hosts running Multicast DNS Responders each assert an SOA record, thereby claiming to be the sole authority for their own little zone within the "local.arpa." domain. That approach makes it difficult for different hosts to manage two or more resource records with the same name, a feature that has some benefits. This document proposes that subdomains of the "local.arpa." domain can never be delegated, and instead "local.arpa." is managed as a single zone implemented by a loose collection of hosts cooperatively executing a distributed algorithm. From that philosophical difference, a variety of implementation differences emerge.

There has been discussion of whether "local.arpa." is an appropriate domain to use. Perhaps it is not. Perhaps some other domain should, by IETF Standards Action, be declared a reserved name in the DNS protocol for this particular use. In any case, the text "local.arpa." in this document should be taken as a place holder for whatever reserved name or "domain" may eventually be allocated for this purpose.

There has been discussion of how much burden Multicast DNS might impose on a network. It should be remembered that whenever IPv4 hosts communicate they broadcast ARP packets on the network on a regular basis, and this is not disastrous. The approximate amount of multicast traffic generated by hosts using Multicast DNS is anticipated to be roughly the same order of magnitude as the amount

of broadcast ARP traffic those hosts already generate.

3. Conventions and Terminology Used in this Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [RFC 2119].

This document uses the term "host name" in the strict sense to mean a fully qualified domain name that has an address record. It does not use the term "host name" in the commonly used but incorrect sense to mean just the first DNS label of a host's fully qualified domain name.

4. Multicast DNS Names

The DNS domain "local.arpa." is (this document proposes) a special domain with special semantics, namely that "local.arpa." and all its subdomains are link-local, and names within this domain are meaningful only on the link where they originate, much as IPv4 addresses in the 169.254/16 prefix are link-local and meaningful only on the link where they originate.

Any DNS query for a name within the "local.arpa." domain MUST be sent to the all-DNS multicast address (224.0.0.251 or its IPv6 equivalent).

It is unimportant whether a name within the "local.arpa." domain occurred because the user explicitly typed in a fully qualified domain name ending in "local.arpa.", or because the user entered an unqualified domain name and the host software appended the "local.arpa." search domain to it. The "local.arpa." domain could appear in the search list because the user manually configured it, or because it was received in a DHCP option, or via any other valid mechanism for configuring the DNS search list. In this respect the "local.arpa." domain is no different to any other search domain that might appear in the list.

DNS queries for a names outside the "local.arpa." domain MAY be sent to the all-DNS multicast address, if no other conventional DNS server is available. This can allow hosts on the same link to continue communicating using each other's globally unique DNS names during network outages which disrupt communication with the greater Internet. This is a contentious issue, and this document does not discuss it in detail, instead concentrating on the issue of resolving local names using DNS packets sent to a multicast address.

A host which belongs to an organization that owns some portion of the DNS namespace can be assigned a globally unique name within that portion of the DNS namespace, for example, "cheshire.apple.com."

Another host, attempting and failing to resolve that name via conventional unicast DNS MAY elect to try resolving it via multicast,

which may be successful if the two hosts happen to be on the same link.

However, the majority home customers do not have easy access to any portion of the global DNS namespace within which they have the authority to create names as they wish. This leaves the majority of home computers effectively anonymous for practical purposes. These users MAY elect to give their computers link-local host names of the form: "single-dns-label.local.arpa." For example, my laptop computer answers to the name "stu.local.arpa." Any computer user is granted the authority to name their computer this way, providing that the chosen host name is not already in use on that link. Having named their computer this way, the user has the authority to continue using that name until such time as name conflict occurs on the link which is not resolved in the user's favour. When this happens, the computer (or its human user) SHOULD cease using the name, and may choose to attempt to allocate a new unique name for use on that link.

The point made in the previous paragraph is very important and bears repeating. It is easy for those of us in the IETF community who run our own name servers at home to forget that the majority of computer users do not run their own name server and have no easy way to create their own host names. When these users wish to transfer files between two laptop computers, they are frequently reduced to typing in dotted-decimal IP addresses because they simply have no other way for one host to refer to the other by name. This is a sorry state of affairs.

Allowing ad-hoc allocation of single-label names in a single flat "local.arpa." namespace may seem to invite chaos. However, operational experience with AppleTalk NBP names, which on any given link are also effectively single-label names in a flat namespace, shows that in practice name collisions happen extremely rarely and are not a problem. Groups of computer users from disparate organizations bring Macintosh laptop computers to events such as IETF Meetings, the Mac Hack conference, the Apple World Wide Developer Conference, etc., and complaints at these events about users suffering conflicts and being forced to rename their machines have never been an issue.

Enforcing uniqueness of host names (i.e. the names of DNS address records mapping names to IP addresses) is probably desirable in the common case, but this document does not mandate that. It is also permissible for a collection of coordinated hosts to agree to maintain multiple DNS address records with the same name, possibly for load balancing or fault-tolerance reasons. This document does not take a position on whether that is sensible, but it is important that the Multicast DNS protocol allows hosts to verify and maintain unique names for resource records where that behaviour is desired, and to

maintain multiple resource records with a single shared name where that behaviour is desired. This consideration applies to all resource records, not just address records (i.e. host names).

5. IP TTL Checks

A host sending a Multicast DNS request to a link-local address MUST verify that the TTL in reply packets is 255, and silently discard any reply packets where the TTL is not 255. Without this check, it could be possible for remote rogue hosts to send spoof answer packets (perhaps unicast to the victim host) which the receiving machine could misinterpret as having originated on the local link.

There has been some discussion that many current network programming APIs to not provide any indication of the TTL on received packets. This is unfortunate, and should be fixed for hosts that want to be able to guard against spoof packets arriving from off-link.

6. Reverse Address Mapping

Like "local.arpa." the domain "254.169.in-addr.arpa." is defined to be link-local. Any DNS query for a name within the "254.169.in-addr.arpa." domain MUST be sent to the all-DNS multicast address 224.0.0.251.

7. Requesting

There are three kinds of Multicast DNS Requests, one-shot requests of the kind made by today's conventional DNS clients, one-shot requests accumulating multiple replies made by multicast-aware DNS clients, and continuous ongoing Multicast DNS Requests used by IP network browser software.

A Multicast DNS Responder that is offering records that are intended to be unique on the local link MUST also implement a Multicast DNS Requester so that it can first verify the uniqueness of those records before it begins answering requests for them.

7.1 One-Shot Requests

An unsophisticated DNS client may simply send its DNS requests blindly to the 224.0.0.251 multicast address, without necessarily even being aware what a multicast address is. Indeed, certain existing DNS clients (e.g. Mac and Windows) can be persuaded to do this even today, simply by the user typing in that address as the 'name server address'.

Such an unsophisticated DNS client may not get ideal behaviour. Such a client may simply take the first response it receives and fail to wait to see if there are more, but in many instances this may not be a serious problem. If a user types "http://stu.local.arpa." into their Web browser and gets to see the page they were hoping for, then the protocol has met the user's needs in this case.

7.2 One-Shot Requests, Accumulating Multiple Replies

A more sophisticated DNS client should understand that Multicast DNS is not exactly the same as unicast DNS, and should modify its behaviour in some simple ways.

As described above, there are some cases, such as looking up the address associated with a unique host name, where a single response is sufficient, and moreover may be all that is expected. However, there are other DNS requests where more than one response is possible, and for these requests a more sophisticated Multicast DNS client should include the ability to wait for an appropriate period of time to collect multiple responses.

A naive DNS client retransmits its request only so long as it has received no reply. A more sophisticated Multicast DNS client is aware that having received one response is not necessarily an indication that it might not receive others, and has the ability to retransmit its request an appropriate number of times at appropriate intervals until it is satisfied with the collection of responses it has gathered.

A more sophisticated Multicast DNS client that is retransmitting a request for which is has already received some replies, MAY elect to implement duplicate suppression, as described below under "Duplicate Suppression". This indicates to responders who have already replied that their responses have been received, and they don't need to send them again in response to this repeated request.

A Multicast DNS Requester MAY place more than one question into the Question Section of a Multicast DNS Request.

7.3 Continuous Requesting

In One-Shot Requests, with either a single or multiple responses, the underlying assumption is that the transaction begins when the application issues a request, and ends when all the desired responses have been received. There is another type of operation which is more akin to continuous monitoring.

Macintosh users are accustomed to opening the "Chooser" window, selecting a desired printer, and then closing the Chooser window. However, when the desired printer does not appear in the list, the user will typically leave the "Chooser" window open while they go and check to verify that the printer is plugged in, powered on, connected to the Ethernet, etc. While the user jiggles the wires, hits the Ethernet hub, and so forth, they keep an eye on the Chooser window, and when the printer name appears, they know they have fixed whatever the problem was. This can be a useful and intuitive troubleshooting

technique, but a user who goes home for the weekend leaving the Chooser window open places a non-trivial burden on the network.

It is important that an IP network browser window displaying live information from the network using Multicast DNS, if left running for an extended period of time, should generate significantly less multicast traffic on the network than the old AppleTalk Chooser.

A Multicast DNS Requester asking the same question repeatedly for an indefinite period of time MUST implement duplicate suppression, as described below.

8. Duplicate Suppression

When a Multicast DNS Requester sends a request to which it already knows some answers, it populates the Answer Section of the DNS message with those cached resource records whose remaining TTL values indicate that they will remain valid for at least the time anticipated to send this DNS request, and the next, and the one after that. For example, if the Multicast DNS Requester is planning to wait four seconds after this request before sending the next, and then eight seconds after that, then only resource records with TTL values greater than twelve seconds should be included in the answer section. This is to ensure that when a resource record's TTL is close to expiration, the Multicast DNS Requester has *two* chances to refresh it before the cached record expires and has to be removed from the list.

A Multicast DNS Responder SHOULD NOT answer a Multicast DNS Request if the answer it would give is already included in the Answer Section with a TTL at least half the correct value. If the TTL of the answer as given in the Answer Section is less than half of the real TTL as known by the Multicast DNS Responder, the responder SHOULD send an answer so as to update the Requester's cache before the record becomes in danger of expiration.

A Multicast DNS Requester MUST NOT cache resource records observed in the Answer Section of other Multicast DNS Requests. The Answer Section of Multicast DNS Requests is not authoritative. By placing information in the Answer Section of a Multicast DNS Request the requester is stating that it *believes* the information to be true. It is not asserting that the information *is* true. Some of those records may have come from other hosts that are no longer on the network. Propagating that stale information to other Multicast DNS Requesters on the network would not be helpful.

A Multicast DNS Responder that implements duplicate suppression SHOULD implement EDNS0 [RFC 2671] to allow larger-sized requests and replies.

9. Responding

A Multicast DNS Responder MUST only reply when it has a positive non-null response to send. Error responses must never be sent. The non-existence of any name in a Multicast DNS Domain is ascertained by the failure of any machine to respond to the Multicast DNS query, not by NXDOMAIN errors.

A Multicast DNS Responder on Ethernet [IEEE802] and similar shared multiple access networks SHOULD delay its responses by a random amount of time selected with uniform random distribution in the range 0-10ms. If multiple Multicast DNS Responders were all to immediately reply to a particular request, a collision would be virtually guaranteed. By imposing a small random delay, the number of collisions is dramatically reduced. 10ms is a short enough time that it is not perceptible to a human user, but long enough to significantly reduce the risk of Ethernet collisions. On a full-sized Ethernet using the maximum cable lengths allowed and the maximum number of repeaters allowed, an Ethernet frame is vulnerable to collisions during the transmission of its first 256 bits. On 10Mb/s Ethernet, this equates to a vulnerable time window of 25.6us.

In the case where a Multicast DNS Responder has good reason to believe that it will be the only responder on the link with a positive non-null response, it MAY reply immediately, without the random delay. To do this safely, it MUST have previously verified that the requested name type and class in the DNS query are unique on this link. This may be appropriate for things like looking up the address record for a particular host name, when the host name has been previously verified unique. This is *not* appropriate for things like looking up PTR records used for DNS Service Discovery [NIAS], where a large number of responses may be anticipated.

Multicast DNS Responses MUST be sent to UDP port 53 (the well-known port assigned to DNS) on the 224.0.0.251 multicast address. Operating in a Zeroconf environment requires constant vigilance. Just because a name has been previously verified unique does not mean it will continue to be so indefinitely. By allowing all Multicast DNS Responders to constantly monitor their peers' responses, conflicts arising out of network topology changes can be promptly detected and resolved.

If the source UDP port in a received Multicast DNS Request is not port 53, this suggests that the client originating the request is an old naive client that is not entirely aware that it is using a multicast address. (The host OS needs to understand what an IP multicast address is in order to hash it to the correct Ethernet multicast address, but the user-level DNS client software does not

need to know anything about multicast to blindly send a UDP packet to the IP address 224.0.0.251.) In this case, after sending the usual

Multicast DNS Response to 224.0.0.251 port 53, the Multicast DNS Responder MUST also send a second identical UDP reply to the client via unicast to the request packet's source IP address and port.

Multicast DNS Responders MUST correctly handle DNS request packets containing more than one question, by answering any or all of the questions to which they have answers.

Multicast DNS Responders SHOULD implement EDNS0 [RFC 2671] to allow larger-sized requests and replies. Larger-sized requests are useful to allow longer duplicate suppression lists in the Answer Section.

10. Startup Procedure

Whenever a Multicast DNS Responder starts up, wakes up from sleep, receives an indication of an Ethernet 'Link Change' event, or has any other reason to believe that its network connectivity may have changed in some relevant way, it MUST perform two startup steps.

The first startup step is that for all those resource records that a Multicast DNS Responder desires to be unique on the local link, it MUST send a Multicast DNS Query asking for those resource records, to see if any of them are already in use. The primary example of this is its address record which maps its unique host name to its unique IP address. The ability to place more than one question in a Multicast DNS Request is useful here, because it can allow a host to use a single packet for all of its resource records instead of needing a separate packet for each. If any conflicting Multicast DNS replies are received, then the host MUST defer to the other host already using those names, and MUST select new names for its conflicting records which need to be unique. One second after the first query it should send a second, then two seconds after that a third. If, after a total of seven seconds, no conflicting Multicast DNS replies have been received, the host may move to the second step.

The second startup step is that the Multicast DNS Responder SHOULD send a gratuitous Multicast DNS Response containing, in the Answer Section, all those resource records that may be of interest to other hosts on the link. One example of this is the PTR records used by DNS Service Discovery [NIAS]. Since other hosts running Multicast DNS Requesters may have network browser windows open using an extremely long interval between Multicast DNS Request packets, the reception of a gratuitous Multicast DNS Response from a new device starting up allows the browser window to update immediately instead of having to wait until the next request is sent.

Up to ten of gratuitous Multicast DNS Responses may be sent, providing that the interval between gratuitous responses doubles

with every response sent, and the interval between the first two gratuitous responses is not less than one second.

Whenever a Multicast DNS Responder receives any Multicast DNS response (gratuitous or otherwise) containing a conflicting resource record, the conflict MUST be resolved as described below in "Conflict Resolution".

A Multicast DNS Responder MUST NOT send announcements in the absence of information that its network connectivity may have changed in some relevant way. In particular, a Multicast DNS Responder MUST NOT send regular periodic announcements as a matter of course.

11. Conflict Resolution

A conflict occurs when two resource records with the same name, type and class have inconsistent rdata. What may be considered inconsistent is context sensitive, except that resource records with identical rdata are never considered inconsistent, even if they originate from different hosts. In the case of a host desiring to have a unique host name, another address record with the same name but a different IP address is considered inconsistent.

Whenever a Multicast DNS Responder receives any Multicast DNS response (gratuitous or otherwise) containing a conflicting resource record, the Multicast DNS Responder must cease using that record and potentially reconfigure.

In the case of a typical laptop or desktop computer with a human user, reconfiguration is achieved by displaying an error message to the user and suggesting that they choose a new name. In the case of a device with no human operator, reconfiguration is achieved by its software programmatically generating a new name. In either case, the host must then test the new name for uniqueness as described above in "Startup Procedure".

It is important that the host that believes there is a conflict be the one to take action. In the case of two hosts using the same host name, where one has been configured to require a unique host name and the other has not, the one configured to require a unique host name must be the one to reconfigure, since the other one doesn't view the sharing of address records as a conflict and hence sees no reason why it should reconfigure. This algorithm could result in situations where both hosts reconfigure, but this will be rare. The uniqueness check described above in "Startup Procedure" helps reduces resource record conflicts to only those cases where two separate links are connected together, or a previously partitioned link is re-joined.

The examples in this section focus on address records (i.e. host names), but the same considerations apply to all resource records where uniqueness or some other defined constraint is desired.

12. Special Characteristics of Multicast DNS Domains

Unlike conventional DNS, the DNS domains "local.arpa." and "254.169.in-addr.arpa." have only local significance. Conventional DNS seeks to provide a single unified namespace, where a given DNS query yields the same answer no matter where on the planet it is performed or to which recursive DNS server the query is sent. (However, split views, firewalls, intranets and the like have somewhat interfered with this goal of DNS representing a single universal truth.) In contrast, each IP link has its own private "local.arpa." and "254.169.in-addr.arpa." namespaces, and the answer to any query for a name within those domains depends on where that query is asked.

Multicast DNS Domains are not delegated from their parent domain via use of NS records. Instead, all Multicast DNS Domains are delegated to the IP address 224.0.0.251 by (potential) IETF Standards Action (i.e. this document, should it become a standard). There are no NS records anywhere in Multicast DNS Domains.

The name server for a Multicast DNS Domain is 224.0.0.251. This is a multicast address; therefore it identifies not a single host but a collection of hosts, working in cooperation to maintain some reasonable facsimile of a competently managed DNS zone. Conceptually a Multicast DNS Domain is a single DNS zone, however its server is implemented as a distributed process running on cluster of loosely cooperating CPUs rather than as a single process running on a single CPU (or tightly coupled multiprocessor).

No delegation is performed within Multicast DNS Domains. Because the cluster of loosely coordinated CPUs is cooperating to administer a single zone, no delegation is necessary or desirable. Just because a particular host on the network may answer queries for a particular record type with the name "example.local.arpa." does not imply anything about whether that host will answer for the name "child.example.local.arpa.", or indeed for other record types with the "example.local.arpa."

Multicast DNS Zones have no SOA record. A conventional DNS zone's SOA record contains information such as the email address of the zone administrator and the monotonically increasing serial number of the last zone modification. There is no single human administrator for any given Multicast DNS Zone, so there is no email address. Because the hosts managing any given Multicast DNS Zone are only loosely coordinated, there is no readily available monotonically increasing serial number to determine whether or not the zone contents have changed. A host holding part of the shared zone could crash or be disconnected from the network at any time without informing the other hosts. There is no reliable way to provide a zone serial number that would, whenever such a crash or disconnection occurred, immediately

change to indicate that the contents of the shared zone had changed.

Zone transfers are not possible for any Multicast DNS Zone.

13. Multicast DNS for Service Discovery

This document does not describe using Multicast DNS for network browsing or service discovery. However, the mechanisms this document describes are compatible with (and support) the browsing and service discovery mechanisms proposed in "Discovering Named Instances of Abstract Services using DNS" [NIAS].

This document places few limitations on what DNS record types may be looked up in the "local.arpa." domain. In particular, a Multicast DNS request for the SRV record named "_dns._udp.local.arpa." may yield the port number and host name (and thence IP address) of a conventional DNS server willing to perform general recursive DNS lookups. The benefit of using this mechanism rather than a DHCP option to configure a host's DNS server address is that using DHCP is an outward-looking solution that makes DNS dependent on another protocol, which may not be running on every network (e.g. an IPv6 network using stateless address autoconfiguration [RFC 2462]). Locating a recursive DNS server using Multicast DNS is a self-sufficient solution that reduces DNS's need for support from other protocols. This possibility is not discussed futher here.

14. Resource Record TTL Values

Multicast DNS resource records used in typical 'One-Shot' requests should generally have fairly low TTL values, on the order of seconds, rather than hours or days. The transient nature of Zeroconf networks [ZC] [v4LL] means that information can change at any time, and a host caching ancient stale resource records with unreasonably long TTL values could be left trying to work with hopelessly out-of-date information.

Having hosts send gratuitous responses when configuration changes occur can somewhat mitigate this problem, but in the event of a network partition, or temporary signal fade in a wireless network, it is not safe to assume that all hosts will necessarily see all gratuitous responses.

The one exception to this recommendation is resource records expected to be used to populate network browser lists, such as the PTR records used for DNS Service Discovery [NIAS]. Using short TTL values here would force the network browser to be continuously sending Multicast DNS Requests to refresh records before they expired from the list. In this case, the harm done by stale data due to high TTL values is relatively mild. The appearance of names in the network browser list is merely an assertion that the name exists now or has existed in the recent past. In order to actually use any named service, the client has to perform another DNS request to find the IP address, and in the

case where the service has been forced to reconfigure to a new IP address (or has left the network entirely), the client will quickly discover that.

15. Enabling and Disabling Multicast DNS

The option to fail-over to Multicast DNS for names outside the "local.arpa." domain SHOULD be a user-configured option, and SHOULD be disabled by default because of the possible security issues related to unintended local resolution of apparently global names.

The option to lookup unqualified (relative) names in the "local.arpa." domain (or not) is controlled by whether or not "local.arpa." appears in the client's DNS search list.

No special control is needed for enabling and disabling Multicast DNS for names within the "local.arpa." domain. The user doesn't need a way to disable Multicast DNS for names within the "local.arpa." domain, because if the user doesn't want to use Multicast DNS, they can achieve this by simply not using names that end in ".local.arpa." If a user *does* enter a name ending in ".local.arpa." into their Web browser, then we can safely assume their intention was probably that it should work. Having user configuration options that can be (intentionally or unintentionally) set so that this doesn't work is just one more way of frustrating the user's ability to perform the tasks they want, perpetuating the view that, "IP networking is too complicated to configure and too hard to use." This in turn perpetuates the continued use of protocols like AppleTalk, and there's no DHCP option to disable that! If we want to retire AppleTalk, we need to offer users equivalent IP functionality that they can rely on to, "always work, like AppleTalk." A little Multicast DNS traffic may be a burden on the network, but it is an insignificant burden compared to continued widespread use of AppleTalk.

16. Considerations for Multiple Interfaces

A host should defend its host name (FQDN) on all active interfaces on which it is answering Multicast DNS requests.

In the event of a name conflict on *any* interface, a host should configure a new host name, if it wishes to maintain uniqueness of its host name.

When answering a Multicast DNS request, a multi-homed host with a link-local address (or addresses) should take care to ensure that any address going out in a Multicast DNS reply is valid for use on the interface on which the reply is going out.

Just as the same link-local IP address may validly be in use simultaneously on different links, the same link-local host name may validly be in use simultaneously on different links, and this is not

an error. A multi-homed host with connections to two different links

may be able to communicate with two different hosts that are validly using the same name. While this kind of name duplication should be rare, it means that a host which wants to fully support this case needs network programming APIs that allow applications to specify on what interface to perform a link-local Multicast DNS request and/or on what interface a Multicast DNS reply was received.

17. DNS Message Format

This section describes specific restrictions on the allowable values for the header fields of a Multicast DNS message.

17.1. ID (Query Identifier)

Multicast DNS clients SHOULD listen for gratuitous responses issued by hosts booting up (or waking up from sleep or otherwise joining the network). Since these gratuitous responses may contain a useful answer to a question for which the client is currently awaiting an answer, Multicast DNS clients SHOULD examine all received Multicast DNS response messages for useful answers, without regard to the contents of the ID field or the question section. In multicast DNS, knowing which particular query message (if any) is responsible for eliciting a particular response message is less interesting than knowing whether the response message contains useful information.

Multicast DNS clients MAY cache any or all Multicast DNS response messages they receive, for possible future use, providing of course that normal TTL aging is performed on these cashed resource records.

In multicast query messages, the Query ID SHOULD be set to zero on transmission.

In multicast responses, including gratuitous multicast responses, the Query ID MUST be set to zero on transmission, and MUST be ignored on reception.

In unicast response messages generated specifically in response to a particular (unicast or multicast) query, the Query ID MUST match the ID from the query message.

17.2. QR (Query/Response) Bit

In query messages, MUST be zero.

In response messages, MUST be one.

17.3. OPCODE

In both multicast query and multicast response messages, MUST be zero (only standard queries are currently supported over multicast, unless other queries are allowed by future IETF Standards Action).

17.4. AA (Authoritative Answer) Bit

In query messages, the Authoritative Answer bit MUST be zero on transmission, and MUST be ignored on reception.

In response messages for Multicast Domains, the Authoritative Answer bit MUST be one -- not setting this bit implies there's some other place where 'better' information may be found.

17.5. TC (Truncated) Bit

In query messages, the Truncated bit MUST be zero on transmission, and MUST be ignored on reception.

In response messages, if the message does not contain all the data the requester was looking for, the requester SHOULD open a TCP connection to the responder and repeat the query.

17.6. RD (Recursion Desired) Bit

In both multicast query and multicast response messages, the Recursion Desired bit MUST be zero on transmission, and MUST be ignored on reception.

17.7. RA (Recursion Available) Bit

In both multicast query and multicast response messages, the Recursion Available bit MUST be zero on transmission, and MUST be ignored on reception.

17.8. Z (Zero) Bit

In both query and response messages, the Zero bit MUST be zero on transmission, and MUST be ignored on reception.

17.9. AD (Authentic Data) Bit [RFC 2535]

In query messages the Authentic Data bit MUST be zero on transmission, and MUST be ignored on reception.

In response messages, the Authentic Data bit MAY be set. Resolvers receiving response messages with the AD bit set MUST NOT trust the AD bit unless they trust the source of the message and either have a

secure path to it or use DNS transaction security.

17.10. CD (Checking Disabled) Bit [RFC 2535]

In query messages, a resolver willing to do cryptography SHOULD set the Checking Disabled bit to permit it to impose its own policies.

In response messages, the Checking Disabled bit MUST be zero on transmission, and MUST be ignored on reception.

17.11. RCODE (Response Code)

In both multicast query and multicast response messages, the Response Code MUST be zero on transmission. Multicast DNS messages received with non-zero Response Codes MUST be silently ignored.

18. IPv6 Considerations

An IPv4-only host and an IPv6-only host behave as "ships that pass in the night". Even if they are on the same Ethernet, neither is aware of the other's traffic. For this reason, each physical link may have *two* unrelated "local.arpa." zones, one for IPv4 and one for IPv6. Since for practical purposes, a group of IPv4-only hosts and a group of IPv6-only hosts on the same Ethernet act as if they were on two entirely separate Ethernet segments, it is unsurprising that their use of the "local.arpa." zone should occur exactly as it would if they really were on two entirely separate Ethernet segments.

A dual-stack (v4/v6) host can participate in both "local.arpa." zones, and should register its name(s) and perform its lookups both using IPv4 and IPv6. This enables it to reach, and be reached by, both IPv4-only and IPv6-only hosts.

There has been discussion of the proposal that in the IPv6 case, the all-DNS multicast address should not be a single address, but instead a range of addresses selected using a hash function of the name being looked for. There are some issues with this:

1. The hash function must work correctly with both normal (case-insensitive) DNS labels and binary labels [RFC 2673].

2. This may prevent more than one question being put into a single packet, since the different questions may hash to different multicast addresses.

3. This impedes the ability to use a single multicast reply packet to answer the client and simultaneously facilitate ongoing conflict monitoring, because every client would have to listen on every multicast address in the range (or rapidly join and leave multicast groups on demand for each request) in order to receive the reply.

4. This limits the ability to gain certain useful functionality out

of old resolver software by configuring it with a single All-DNS multicast address to which it can send its queries.

19. Security Considerations

DNSSEC [RFC 2535] should be used where the authenticity of information is important.

When DNS queries for names outside the "local.arpa." domain are sent to the all-DNS multicast address (during of network outages which disrupt communication with the greater Internet) it is *especially* important to use DNSSEC, because the user may have the impression that he or she is communicating with some authentic host, when in fact he or she is really communicating with some local host that is merely masquerading as that name. This is less critical for names within the "local.arpa." domain, because within this domain the user can be aware that names have only local significance and no global authority is implied.

Most computer users neglect to type the trailing dot at the end of a fully qualified domain name, making it a relative domain name (e.g. "www.example.com"). In the event of network outage, attempts to positively resolve the name as entered will fail, resulting in application of the search list, including "local.arpa.", if present. A malicious host could masquerade as "www.example.com" by answering the resulting Multicast DNS request for "www.example.com.local.arpa." To avoid this, a host MUST NOT append the search domain "local.arpa.", if present, to any relative (partially qualified) domain name containing two or more labels. Appending "local.arpa." to single-label relative domain names is acceptable, since the user should have no expectation that a single-label domain name will resolve as-is.

[Lots more work to be done here!]

20. IANA Considerations

The IANA has allocated the IPv4 link-local multicast address 224.0.0.251 for the use described in this document.

We'd like the IANA to designate the DNS domain "local.arpa." a "Multicast Domain" with special semantics, namely that "local.arpa." and its subdomains are link-local, and names within this domain are meaningful only on the link where they originate, much as IPv4 addresses in the 169.254/16 prefix are link-local and meaningful only on the link where they originate. Likewise we'd like the IANA to designate the DNS domain "254.169.in-addr.arpa." to be similarly link-local and non-delegated.

No other IANA services are required by this document.

21. Copyright

22. References

[IEEE802] IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture. Institute of Electrical and Electronic Engineers, IEEE Standard 802, 1990.

[NIAS] S. Cheshire, "Discovering Named Instances of Abstract Services using DNS", Internet-Draft (work in progress), draft-cheshire-dnsext-nias-00.txt, July 2001.

[RFC 2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC 2462] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.

[RFC 2535] D. Eastlake, "Domain Name System Security Extensions", RFC 2535, March 1999.

[RFC 2671] P. Vixie, "Extension mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC 2673] M. Crawford, "Binary Labels in the Domain Name System", RFC 2673, August 1999.

[v4LL] S. Cheshire and B. Aboba, "Dynamic Configuration of IPv4 Link-Local Addresses", Internet-Draft (work in progress), draft-ietf-zeroconf-ipv4-linklocal-03.txt, June 2001.

[ZC] M. Hattig, "Zeroconf Requirements", Internet-Draft (work in progress), draft-ietf-zeroconf-reqts-08.txt, May 2001.

23. Author's Address

Stuart Cheshire
Apple Computer, Inc.
1 Infinite Loop
Cupertino
California 95014
USA

Phone: +1 408 974 3207
EMail: rfc@stuartcheshire.org