Stuart Cheshire
Apple Computer

Document: draft-cheshire-dnsext-nbp-00.txt
Expires 10th March 2002
10th September 2001

Requirements for the Replacement of AppleTalk Name Binding Protocol

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

Status of this Memo

Abstract

An understood, but poorly specified, goal of the current work on Multicast DNS is the ability to retire AppleTalk Name Binding Protocol and replace it with an all-IP solution. This draft outlines the specific properties required of an IP replacement for AppleTalk Name Binding Protocol.

1. Introduction

A common goal of many of the parties working on Multicast DNS [mDNS-EAT] [mDNS-SC] is to provide a viable IP-based replacement for AppleTalk Name Binding Protocol (NBP). Unfortunately, the precise requirements of such an IP-based replacement have been assumed but not written down. Furthermore, it is likely that each person has a different idea of what the unstated assumptions are, leading to miscommunication and misunderstandings when discussing what Multicast DNS should do and how it should work. Finally, there are many who are experts in the area of DNS who know nothing about NBP, and without any knowledge of the unstated goal, many of the discussions on this topic probably look like utter nonsense.

This draft seeks to remedy this problem by clearly stating the requirements for an IP-based replacement for NBP. Replacing NBP is not the only goal of Multicast DNS, and therefore these requirements are not the only design considerations. However, replacing NBP is a major motivation behind the work in Multicast DNS. A Multicast DNS solution that is, amongst other things, a viable replacement for NBP, is much more compelling than one which is not.

In most cases, the requirements presented in this document are simply a restatement of what AppleTalk NBP currently does. However, this document is not restricted to describing only what NBP currently does. In some cases, the requirements for a viable IP-based replacement go beyond NBP. For example, AppleTalk NBP uses Apple Extended ASCII for its character set. It is clear that an IP-based replacement being designed today should use Unicode, probably in the form of UTF-8. AppleTalk NBP has no built-in security provisions; an IP-based replacement cannot have that same error. AppleTalk NBP has a reputation, partially deserved, partially not, for being too 'chatty' on the network. An IP-based replacement should not have this same failing. The intent is to learn from NBP and build a superset of its functionality, not to replicate it precisely with all the same flaws.

The proposals described in "Performing DNS queries via IP Multicast" [mDNS-SC] and "Discovering Named Instances of Abstract Services using DNS" [NIAS], taken together, describe a solution that meets these requirements. This document is written, in part, in response to a request for more background information to support why those proposals are necessary.

2. Requirements

This Section lists the 11 requirements for an IP-based replacement for AppleTalk NBP.

2.1 Name-to-Address Mapping

NBP's primary function is translating names to addresses.

NBP stands for Name Binding Protocol, not Network Browsing Protocol. Many people know NBP only as "that thing that lets you browse the network in the Macintosh Chooser". While browsing is an important facility of NBP, it is secondary to NBP's primary function of translating names to addresses.

Every time a user prints using AppleTalk, the printing software takes the name of the currently selected printer, looks up the current AppleTalk address associated with that named service, and establishes a connection to that service on the network. The user may invoke NBP's browsing capability once when first selecting the desired printer in the Chooser, but then after that, every single time they print anything, it is a simple efficient name-to-address lookup that is being performed, not a full-fledged browsing operation.

Any NBP replacement needs to support, as it's primary function, an efficient name-to-address lookup operation.

2.2 Name Services, not Hardware

The primary named entities in NBP are services, not "hosts", "machines", "devices", or pieces of hardware of any kind. This concept is more subtle than it may seem at first, so it bears some discussion.

The AppleTalk NBP philosophy is that naming a piece of hardware on the network is of little use if you can't communicate with that piece of hardware. To communicate with a piece of hardware, there needs to be a piece of software running on that hardware which sends and receives network packets conforming to some specific protocol. This means that whenever you communicate with a machine, you are really communicating with some piece of software on that machine. Even if you just 'ping' a machine to see if it is responding, it is not really the machine that you are 'pinging', it is the software on that machine that generates ICMP Echo Responses.

Consequently, this means that the only thing worth naming is the software entities with which you can communicate. A user who wants to use a print server or a file server needn't care about what hardware implements those services. There may be a single machine hosting both services, or there may be two separate machines. The end user doesn't need to care.

The one exception to this is network managers, who may want to name physical hardware for the purpose of tracking physical inventory. However, even this can be recast into a service-oriented view of the world by saying that what you're naming is not the hardware, but the ICMP Echo Responder that runs (or is assumed to be running) on every piece of IP hardware.

2.3 Address Services, not Hardware
-or-
Escape the Tyranny of Well Known Ports

The reader may argue that DNS already supports the philosophy of naming services instead of hosts. When we see names like "www.example.com.", "pop.example.com.", "smtp.example.com.", "news.example.com." and "time.example.com.", we do not assume that each of those names refer to a different host. They are clearly intended to be logical service names, and could in fact all refer to the same IP address.

The shortcoming here is that although the names are clearly logical service names, the result today of doing a conventional ("A" Record) DNS lookup for those names gives you only the IP address of the hardware where the service is located. To communicate with the desired service, you also need to know the TCP or UDP port number at which the service can be reached, not just the IP address.

This means that the port number has to be communicated out-of-band, in some other way. One way is for the port number to be a specific well-known constant for any given protocol. This makes it hard to run more than one instance of a service on a single piece of hardware. Another way is for the user to explicitly type in the port number, for example, "www.example.com.:8080" instead of "www.example.com.", but needing to know and type in a port number is as ugly and fragile as needing to know and type in an IP address.

Another aspect of the difficulty of running more than one instance of a service on a single piece of hardware is that it forces application programmers to write their own demultiplexing capability. For example, if an AppleTalk print server offers three print queues, each print queue runs as its own independent service, listening on its own port number (called a socket number in AppleTalk terminology) and each is advertised as a separate independent named NBP entity. When a client looks up the address of that named NBP entity, the reply encodes not only on which net and subnet the service resides, and on which host on that subnet (like an IP address does), but also on which port number (socket number) within that host. In contrast, if an lpr print server offers three print queues, all three print queues are typically reached through the same well-known port number, and then the lpr protocol has to use it own demultiplexing capability (the print queue name) in order to determine which print queue is sought. This makes it especially difficult to run two different pieces of print queue software from different vendors on the same machine, because they cannot both use the same well-known port.

A similar trick is used in HTTP 1.1, where the Host header is used to allow multiple logical http services to run at the same IP address. Again, this works for a single-vendor solution, but if you have an image server, a database program, an http email access gateway, and a regular http server, they can't all run on the same TCP port on the same machine.

Yet another problem of well-known ports is that port numbers are a finite resource. Originally, port numbers 0-1023 were reserved for well-known services, and the remaining 98% of the port space was free for dynamic allocation. Since then, the range of "Registered Ports" has crept upwards until today, ports 0-49151 are reserved, and only 25% of the space remains available for dynamic allocation. Even though 65535 may seem like a lot of available port numbers, with the pace of software development today, if every new protocol gets its own private port number, we will eventually run out. To avoid having to do application-level demultiplexing, protocols like the X Window System wisely use a range of port numbers, and let TCP do the demultiplexing for them. The X Window System uses 64 ports, in the range 6000-6063. If every new protocol were to get its own chunk of 64 ports, we would run out even faster.

Any NBP replacement needs to provide, not just the network number, subnet number, and host number within that subnet (i.e. the IP address) but also the port number within that host where the service is located. Furthermore, since many existing IP services such as lpr *do* already use additional application-layer demultiplexing information such as a print queue name, an NBP replacement needs to support this too by including this information as part of the complete package of addressing information provided to the client to enable it to use the service. The NBP replacement needs to name individual print queues as first-class entities in their own right. It is not sufficient to name a print server, within which separate print queues can be found.

One possible answer here is that an IP-based NBP replacement could use a solution derived from DNS "SRV" records instead of "A" records, since SRV records *do* provide a port number. However, this alone is not a complete solution, because "SRV" records cannot tell you an lpr print queue name.

2.4 Typed Name Space

AppleTalk NBP names are structured names, of the form:

```
   Name : Type @ Zone
```

The Name is the user-visible name of the service.

The Type is an opaque identifier which identifies the type of service. For convenience, the opaque identifier is generally constructed using descriptive ASCII text, but this text has no meaning to the protocol, and care should be taken in inferring too much meaning from it. For example, the NBP Service Type "LaserWriter" means "any service that speaks PostScript over AppleTalk Printer Access Protocol". It does not necessarily mean an Apple-branded "LaserWriter" printer; nor does the service even have to be a printer. A device that archives documents to recordable CDs could advertise itself as a "LaserWriter", meaning that it speaks PostScript over PAP, not necessarily that it prints that PostScript on paper when it gets it. The end-user never directly sees the Service Type. It is implicit in the user's action; e.g. when printing, the printing software knows what protocol(s) it speaks and consequently what Service Type(s) it should be looking for -- the user doesn't have to tell it.

The Zone is an organizational or geographical grouping of named services. Typical AppleTalk Zone Names are things like "Engineering" and "Sales". The equivalent concept in DNS could be a subdomain such as "engineering.company.com." or "sales.company.com."

Each {Type,Zone} pair defines a name space in which service names can be registered. It is not a name conflict to have a printer called "Sales" and a file server called "Sales", because one is "Sales:LaserWriter@Zone" and the other is "Sales:AFPServer@Zone".

Any NBP replacement needs to provide a mechanism that allows names to be grouped into organizational or geographical "zones", and within each "zone", to provide an independent name space for each service type.

2.5 User-Friendly Names

When repeatedly typing in names on command-line systems, it is helpful to have names that are short, all lower-case, with no spaces or other unusual characters.

Since Service Names are intended to be selected from a list, not repeatedly typed in on a keyboard, there is no for them to be restricted so. Users should be able to give their printers names like "Sales", "Marketing", and "3rd Floor Copy Room", not just "printer1.ietf.org." Of course a user is free to restrict their

Service Names to lower-case letters without spaces if they wish, but they should not be forced to do that.

Any NBP replacement needs to support a full range of rich text characters, including upper case, lower case, spaces, accented characters, and so on. The correct solution is likely to be Unicode, probably in the form of UTF-8.

Note that although the characters ':' and '@' are used when writing AppleTalk NBP names, they are simply a notational convenience in written text. In the on-the wire protocol and in the software data structures, NBP Name, Type and Zone strings are all allowed to contain arbitrary characters, including ':' and '@'.

2.6 Zeroconf Operation

AppleTalk NBP is self-configuring. On a network of just two hosts, they communicate peer-to-peer using multicast. On a large managed network, AppleTalk routers automatically perform an aggregation function, allowing name lookups to be performed via unicast to a service running on the router, instead of by flooding the entire network with multicast packets to every host.

Any NBP replacement needs to operate in the absence of external network infrastructure. It should also be able to take advantage of appropriate external network infrastructure, where present, to perform queries via unicast instead of multicast.

2.7 Name Space Management
-or-
Name Conflict Detection

Because an NBP replacement needs to operate in a Zeroconf environment, it cannot be assumed that a central network administrator is managing the network. In a managed network normal administrative controls may apply, but in the Zeroconf case an NBP replacement must make it easy for users to name their devices as they wish, without the inconvenience or expense of having to seek permission or pay some organization like a domain name registry for the privilege. However, this ease of naming and freedom to choose any desired name means that two users may independently decide to run a personal file server on their laptop computers, and (unimaginitively) name it "My Computer". When these two users later attend the next IETF meeting and find themselves part of the same wireless network, there may be problems.

Similarly, every Epson Ethernet printer may ship from the factory with its Service Name set to "Epson Printer". On a typical small home network where there is only one printer this is not a problem, but it could be a problem if two or more such printers are connected to the same network.

Any NBP replacement needs to detect such conflicts, and handle them appropriately. In the case of the laptop computers, which have keyboards, screens, and human users, the software should display a message telling one or both users that they need to select a new name.

In the case of the printers which have no keyboard or screen, the software should automatically select a new unique name, perhaps by appending an integer to the end of the existing name, e.g. "Epson Printer 2".

Because of the potentially transient nature of connectivity on the wireless networks that are becoming more and more common, this name conflict detection needs to be an ongoing process. It is not sufficient to simply verify uniqueness of names for a few seconds during the boot process and then assume that the names will remain unique indefinitely.

2.8 Late Binding

When the user selects their default printer, the software should not store the IP address and port number, but just the name. Then, every time the user prints, the software should look up the name to find the current IP address and port number for that service. This allows a named logical service to be moved from one piece of hardware to another without disrupting the user's ability to print to that named print service.

On a network using DHCP or self-assigned link-local addresses, a device's IP address may change from day to day. By deferring binding of name to address until actual use, this allows the client to get the correct IP address at the time the service is used.

Similarly, with a service using a dynamic port number instead of a fixed well-known port, the service may not get the same port number every time it is started or restarted. By deferring binding of name to port number until actual use, this allows the client to get the correct port number at the time the service is used.

2.9 Simplicity

Any NBP replacement needs to be simple enough that vendors of even the cheapest ink-jet printer can afford to implement it in the device's limited firmware.

2.10 Network Browsing

AppleTalk NBP offers certain limited wildcard functionality. For example, the service name "*" means "any name". This allows a client to perform an NBP lookup such as "*:LaserWriter@MyZone" and receive back in response a list of all the PAP (AppleTalk Printer Access Protocol) printers in the Zone called "MyZone".

Any NBP replacement needs to allow a piece of software, such as a printing client, or a file server client, to enumerate all the named instances of services in a specified zone (domain) which speak its protocol(s).

2.11 Browsing and Registration Guidance

AppleTalk NBP provides certain meta-information to the client.

On a network with multiple AppleTalk Zones, the AppleTalk network infrastructure informs the client of the list of Zones that are available for browsing. It also informs the client of the default Zone, which defines the client's logical "home" location. This is the Zone that is selected by default when the Macintosh Chooser is opened, and is usually the Zone where the user is most likely to find services like printers that are physically nearby, but the user is still free to browse any Zone in the offered list that they wish.

An Epson printer may be preconfigured at the factory with the Service Name "Epson Printer", but they do not know on which network the printer will eventually be installed, so the printer will have to learn this from the network on arrival. On a network with multiple AppleTalk Zones, the AppleTalk network infrastructure informs the client of a single default Zone within which it may register Service Names. In the case of a device with a human user, the AppleTalk network infrastructure may also inform the client of a list of Zones within which the client may register Service Names, and the user may choose to register Service Names in any one of those Zones instead of in the suggested default Zone.

Any NBP replacement needs to provide the following information to the client:

* The suggested zone (domain) in which to register Service Names.
* A list of recommended available zones (domains) in which Service Names may be optionally registered.
* The suggested default zone (domain) for network browsing.
* A list of available zones (domains) which may be browsed.

3. Existing Protocols

The question has been asked, "Isn't SLP the IETF replacement for NBP?"

SLP [RFC 2608] provides extremely rich and flexible facilities in the area of Requirement 10, "Network Browsing". However, SLP provides none of the service naming, automatic name conflict detection, or efficient name-to-address lookup which form the majority of what AppleTalk NBP does.

SLP returns results in the form of URLs. In the absence of DNS, URLs cannot usefully contain DNS names. Discovering a service URL of the form "http://169.254.17.202/" is not particularly informative to the user. Discovering a service URL of the form "http://epson_stylus_900n.local.arpa./" is slightly more informative (though still not very user-friendly), but on a Zeroconf network this presupposes the existence of Multicast DNS to do the name lookup.

SLP provides fine-grained query capabilities, such as the ability to prune a long list of printers to show only those that have blue paper in the top tray, which could be useful on extremely large networks with very many printers, but may be overkill for a small home network with only one or two printers.

It is entirely possible that it may be possible to extend SLP in some trivial way to provide the desired name space definition and management functions as a standard feature of the protocol. If so, I welcome feedback from experts in the area of SLP about how that might be done.

4. IPv6 Considerations

An IP replacement for AppleTalk Name Binding Protocol needs to support IPv6 addresses as well as it supports IPv4 addresses.

5. Security Considerations

AppleTalk Name Binding Protocol has no inherent security mechanism. This would not be acceptable in an IP replacement. It should be possible for a client to verify the authenticity of the information it is receiving. It may also be useful for a server to be able to verify that a client has authority to request that information, and it may be useful to have a way to encrypt the data in transit to protect it against eavesdropping.

A solution based on or derived from DNS may be able to use DNSSEC [RFC 2535] to meet some of these requirements.

6. IANA Considerations

AppleTalk Name Binding Protocol defines a name space for Zones, a name space for service Types, and name spaces for named instances of those services. Each name space uses 32-character ASCII text strings, so the name space for Type names is sufficiently large and sufficiently sparsely used that Apple never bothered with maintaining an official registry of assigned NBP service Type names.

In an IP replacement, the name space of zones (domains) would be managed the same way as domains are currently managed, which is to say through delegation from the root. In addition, if Multicast DNS is successful [mDNS-EAT] [mDNS-SC] there will also be a specially reserved domain available for local use without the overhead of formal delegation.

IANA should probably manage the name space of service type names, to prevent unintended name collisions. However, the name space of textual names is large enough that type names would not be a precious resource, so they could be handed out freely to anyone who needs one, effectively without limit.

The name space of instance names is managed locally at each site.

7. Copyright

8. References

[mDNS-EAT] Esibov, Aboba & Thaler, "Multicast DNS", Internet-Draft (work in progress), draft-ietf-dnsext-mdns-03.txt, July 2001.

[mDNS-SC] S. Cheshire, "Performing DNS queries via IP Multicast", Internet-Draft (work in progress), draft-cheshire-dnsext-multicastdns-00.txt, July 2001.

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

[RFC 2608] Guttman, Perkins, Veizades & Day, "Service Location Protocol, Version 2", RFC 2608, June 1999.

9. Author's Address

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

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