Erik Nordmark
Sun Microsystems

Multi6 Application Referral Issues

<draft-ietf-multi6-app-refer-00.txt>

Status of this Memo

By submitting this Internet-Draft, I certify that any applicable patent or other IPR claims of which I am aware have been disclosed, and any of which I become aware will be disclosed, in accordance with RFC 3668.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet Draft expires July 10, 2005.

Copyright Notice

Abstract

In order to fully solve the scalable multihoming problem there is a need to separate the current IP address functionality into identifiers (which are used to identify e.g., TCP connections) and locators which are used to forward packets in the routing system. Such a separation has an impact on the current use of IP address in the application layer.

This document presents these issues for the purposes of stimulating discussions.

Contents

1. INTRODUCTION

The goal of the IPv6 multihoming work is to allow a site to take advantage of multiple attachments to the global Internet without having a specific entry for the site visible in the global routing table. Specifically, a solution should allow hosts to use multiple attachments in parallel, or to switch between these attachment points dynamically in the case of failures, without an impact on the transport and application layer protocols.

This document assumes a particular approach to solving this which is based on there being multiple locators (recorded in the DNS using AAAA records) for a host, and for a given transport connection, a pair consisting of a local and a peer locator is chosen as part of the identification of the connection. Thus this approach does not introduce any new identifier name space, instead the applications and transports "see" what looks like just an IP address even though some multi6 layer below provides survivability when one of the multiple locators become unreachable. Some more aspects about this approach

is specified in [M6SHIM]. For example,the NOID proposal [NOID] used this model, and NOID defined a term AID, for "Application Identifier", for the locator that is chosen to be part of the identification of the transport connection. The AID term is less than ideal; in this document we use the term "ULID" for upper-layer identifier for lack of a better one, for this concept.

Effectively this approach implies that while a single IP address (the ULID) is presented to the applications at the API, this IP address serves as a handle for the set of IP addresses/locators for the peer.

1.1. Current IP address usage in applications

For the purposes of understanding the impact of multi6 on applications we here attempt to categorize the usage of IP addresses in applications:

- Short-lived local handle. The IP addresses is never retained by the application. The only usage is for the application to pass it from the DNS APIs (e.g., getaddrinfo()) and the API to the protocol stack (e.g., connect() or sendto()).

- Long-lived application associations. The IP address is retained by the application for several instances of communication.

- Callbacks. The application at one end retrieves the IP address of the peer and uses that to later communicate "back" to the peer.

- Referrals. In an application with more than two parties, party B takes the IP address of party A and passes that to party C. After this party C uses the IP address to communicate with A.

- "Identity" comparison. Some applications might retain the IP address, not as a means to initiate communication as in the above cases, but as a means to compare whether a peer is the same as another peer. While this is insecure in general, it might be something which is used e.g., when TLS is used.

  Using an IP address for "identity comparison" is problematic today due to the lack of security, but also because hosts, such as large servers, might be multihomed even in IPv4 resulting in there not being a single IP address that can identify the host. The introduction of site multihoming using different locator prefixes will make this even more prevalent. Thus we need to understand to what extent IP addresses are used today for "identity comparison" in applications; perhaps this is a non-issue.

  The reason "identity comparison" solely using IP addresses is

insecure is because the IP address spoofing is a common element of attacks in today's Internet. Even though there are techniques such as ingress filtering [INGRESS] which can reduce the places from which arbitrary IP address spoofing is possible, they are not deployed everywhere, and can not prevent "local" IP address spoofing such as using a different IP address in the same subnet. And when using some security, such as IPsec or TLS, then it is true that the application will have better certainty about the identity of the peer in many cases, but the identity it has certainty about isn't the IP address, but the identity to which the certificate is bound.

Stated differently, if host A uses IP address X with IKE/IPsec and its certificate, and host B later uses the same IP address with IKE/IPsec and its certificate, it doesn't mean that the IP address refers to the same identity in the two cases.

1.2. Problem Statement

The use of a single locator as a handle for the set of locators for the peer at the API impacts the applications as typified above except for the case of the short-lived handle. This case is simple because in fact the IP address, when not used for anything in the application itself, is just a handle that is carried between the DNS API and the TCP/UDP APIs, hence having it effectively be a handle on the set of locators does not introduce any effect on the applications.

The long-lived local handle, callbacks, and referrals cases can continue to work in limited ways, but such applications will not be able to take advantage of the multiple locators of the peers. This limitation seems rather natural given that the only name space we will have with this approach that would name a node using a single name is a FQDN (even though FQDNs have issues as will be discussed below), and to name a node using IP addresses would require using the entire set of locators. Thus unless we introduce a new 128-bit name space for node identifiers (or "stack names" in [NSRG]) there would need to be some changes to applications with these types of IP address usage.

In this approach, the upper level protocols will operate on ULIDs which are mere locators. Thus as long as a site hasn't renumbered, the ULID can be used to either send packets to the host, or (e.g. if that locator isn't working) it is possible for an application to do a reverse lookup plus forward lookup of the ULID to get the set of locators for the peer.

Once a site has been renumbered, the ULIDs which contain the old

prefix will no longer be useful. Hence applications must try to honor the DNS TTL somehow. But this is a renumbering issue and not an effect of the multihoming support.

Applications, which map the peer's IP address to a domain name, today perform a reverse lookup in the DNS (e.g., using the getnameinfo() API). The approach [M6SHIM] doesn't add or subtract to neither the benefits nor the issues associated with performing such reverse lookups.

Applications which today either retain a peer's IPv6 address for future use, such as connecting back to that peer ("callbacks"), or pass a peer's IPv6 address to a third party ("referrals") will not break with this multihoming support; they will end up retaining and/or passing the ULID instead of an IPv6 address. Since the ULID is a locator things will still work as long as that locator is reachable.

Should the locator which is the ULID not be reachable, the application/host will fail to communicate with the peer. If the DNS is maintained, a reverse plus forward lookup of the ULID can be used to determine the other locators. Whether these lookups can be hidden from the application, or whether the applications need to be modified to make the callbacks and referrals take full advantage of the multihoming is for further study.

Alternatively, the applications can be modified to either pass a domain name, or pass the set of locators, when performing referrals. Such an approach would handle multihoming, but not necessarily site renumbering.

Note that large parts of the issues of multiple locators for a host where in fact introduced as part of the dual-stack transition mechanism for IPv6, where a host ends up with at least one IPv4 address and at least one IPv6 address. Thus the introduction of multihoming using multiple locators is not the first case of introducing these issues to applications, but can be viewed as an opportunity to better architect the general issue.

2. ALTERNATIVES

Given that multihoming will at least introduce multiple addresses/locators for a given host, continuing to use a single address for the long-lived local handle, callbacks, and referrals cases isn't likely to be optimal.

2.1. FQDN instead of IP addresses

Applications where it is possible to use FQDNs (or protocol elements which embed FQDNs such as URIs) should definitely do so because that would hide the multiple addresses from the core of the application.

However, it is far from clear that this is feasible for all applications in all deployments. Reasons why FQDNs are problematic include:

- Client hosts typically do not have a FQDN which is resolvable in the DNS today.

- Even if the bullet above is addressed somehow, the introduction of RFC 3041 temporary addresses for improved privacy adds to the burden. Today there is no mechanism for a host with temporary addresses to create temporary FQDNs that resolve to those addresses, and temporary FQDNs are necessary to preserve the difficulty of correlating these addresses with more permanent identifiers of the host.

- At the server end a FQDN is often used, not as a host name, but as an identifier of a service, where the service might be implemented by multiple hosts. In some cases the individual servers might have their own host name as well (such as www17.example.com) but it isn't clear how common this is.

2.2. Single IP address

As stated above a single address can't in general represent all the locators of a host, and all the locators are needed for the application to take advantage of the multiple paths offered by multihoming. However, in the specific case where the DNS forward and reverse maps for a host are maintained, a single address is sufficient.

In this case it is possible to perform a reverse lookup of the single IP address to find out the FQDN, followed by a forward lookup of the FQDN to find all the AAAA records. This method finds all the locators registered in the DNS but assumes that both the reverse tree

and the forward tree are maintained in some consistent manner. This is not likely to be the case for RFC 3041 temporary addresses [RFC3041], and will be problematic for "home multihoming" where a small site is multihomed using multiple ISPs and each ISP provides the forward and reverse DNS for the IP addresses it provides to the site but there is no place to have a single FQDN which maps to both the IP addresses with matching entries in the reverse tree, unless the reverse tree is delegated to the subscriber.

Hence this is not recommended except as a transitional measure.

2.3. Full set of IP addresses

When FQDNs can not be used, the second most attractive approach to handle the general case of multihoming with multiple locators per host is to make the applications that use long-lived handles, perform callbacks, or referrals, use the full set of IP addresses for their peers and themselves.

This approach requires APIs by which the applications can retrieve the locators used. For example, at the socket API layer, such APIs could be getmylocators(int socket, ...) and getpeerlocators(int socket, ...) which return the set of locators for the local and remote end of the communication represented by a socket.

Together with the existing getsockname() and getpeername() API calls, which return the identifier used by the transport protocol (the ULID), such new APIs would allow the applications to retrieve the transport layer identifiers, whether or not they are locators, and the locator sets.

3. FUTURE EVOLUTION

There are two things which might have an impact on long-lived sessions, callbacks and referrals in the near future. The first in using unreachable locators, for instance centrally-assigned Unique local addresses [ULA], as ULIDs. The second is the possibility of introducing a scheme with a separate 128-bit identifier name space and lookup functions, where [HIP] is the best current example.

Unique local addresses could be treated following approach as just another locator which is known to be unreachable (at least when outside the site to which it is assigned), thus there might be some performance optimizations that can be applied. But the fact that the

identifier is centrally assigned means that it is reasonable to maintain reverse and forward DNS entries for at least the /48 prefix. Depending on issues like RFC 3041 addresses, it may or may not be reasonable to expect reverse and forward DNS entries for the full ULA. But in any case, this doesn't seem to have an impact on how referrals would take place; passing a FQDN or the list of locators would work in any case. However, should the unreachable ULA not be treated as a locator but instead solely as a ULID, then the referrals would need to carry a ULID plus the list of locators.

As currently defined HIP uses a 128-bit hash of a public key (the HIT) at the socket API and in the transport protocols, which has quite different implications than using one of the locators which is the focus of this document.

A question is whether there are some minimum things that can be done that would cause less future obstacles to adopt HIP on a large scale, should that come to pass. The thing that is an obvious issue in terms of this proposal is the assumption that the ULID is one of the locators. As a result of this, the recommendations below suggest that the ULID be carried together with the list of locators, so that should the ULID not be one of the locators the application mechanism will at least not loose track of the ULID.

This clearly doesn't solve the issue of referrals etc for HIP, since a general solution would require some way to lookup a HIT which does not yet exist. But it does at least ensure that the applications would contain both the ULID=HIT and the set of locators.

4. RECOMMENDATIONS

The applications that are not already using FQDNs, and can't be converted to use FQDNs, seem to be best served by carrying a list locators plus the ULID where these applications today carry a single IP address.

Carrying the ULID in addition to the list of locators means that the applications is more likely to be able to move to a possible future world where the ULID is not necessarily one of the locators.

This requires the definition and implementation of new APIs for the applications to retrieve the set of peer and local locators from the protocol stack, as exemplified in the section above with the made up getpeerlocators() and getmylocators() API calls. But it also

requires a mechanism by which the application can pass that information to the protocol stack so that the multi6 protocol layer has all the alternate locators available when establishing communication. Thus we need to study the implications of a setpeerlocators() type of API.

Note that the multihoming approach we are discussing, and as a result these recommendations for applications, does not necessarily handle the all the cases of renumbering any better than we handle them today in the IPv4 Internet. For instance, when renumbering the set of locators for a host will change. If a peer retains the set of locators, as long as at least one of those remain valid after the renumbering (and that locator doesn't experience a temporary routing failure), the peer will be able to contact the host. But should all the locators be renumbered, the stale set of locators retained by the peer will be useless. Hence the ULID plus the set of locators will not serve as some long-term stable "identifier" for a host, but they do allow applications to take full advantage of the multiple paths available when multihoming is used.

Furthermore, note that noting in these recommendations say that applications need to be aware of any semantics of the different locators in a set; the applications merely need to store the set and pass it to peers when doing referrals.

5. ACKNOWLEDGMENTS

This document was originally produced of a MULTI6 design team consisting of (in alphabetical order): Jari Arkko, Iljitsch van Beijnum, Marcelo Bagnulo Braun, Geoff Huston, Erik Nordmark, Margaret Wasserman, and Jukka Ylitalo.

6. REFERENCES

6.1. Normative References
6.2. Informative References

[M6SHIM] Nordmark, E, and M. Bagnulo, Multihoming L3 Shim Approach, draft-ietf-multi6-l3shim-00.txt, Jan 2005.

[NSRG] Lear, E., and R. Droms, "What's In A Name: Thoughts from the NSRG", draft-irtf-nsrg-report-09.txt (work in progress), March 2003.

[RFC3041] T. Narten and Draves, R, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", January 2001.

[NOID] Erik Nordmark, "Multihoming without IP Identifiers", Oct 27, 2003, <draft-nordmark-multi6-noid-01.txt>

[ULA] R. Hinden, and B. Haberman, Centrally Assigned Unique Local IPv6 Unicast Addresses, draft-ietf-ipv6-ula-central-00.txt

[HIP] Pekka Nikander, "End-Host Mobility and Multi-Homing with Host Identity Protocol", 17-April-04, draft-ietf-hip-mm-00.txt

AUTHOR'S ADDRESSES

Erik Nordmark
Sun Microsystems, Inc.
17 Network Circle
Mountain View, CA
USA

phone: +1 650 786 2921
fax: +1 650 786 5896
email: erik.nordmark@sun.com

APPENDIX A: CHANGES SINCE PREVIOUS DRAFT

The following changes have been made since draft-nordmark-multi6dt-refer-00.txt:

- Clarified why identity comparison is insecure in general.
- Point out that issues where in fact already introduced with dual-stack IPv6 transition, where a single host has one IPv4 and one

IPv6 address.

- Clarified that applications should not be concerned with the semantics of the locators in the set; just need to carry the set around instead of carrying a single address around.

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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