DMM Working Group
Internet-Draft
Intended status: Informational
Expires: August 2, 2017

A. Yegin
Actility
D. Moses
Intel
K. Kweon
J. Lee
J. Park
Samsung
S. Jeon
Sungkyunkwan University

January 29, 2017

On Demand Mobility Management
draft-ietf-dmm-ondemand-mobility-10

Abstract

Applications differ with respect to whether they need IP session continuity and/or IP address reachability. The network providing the same type of service to any mobile host and any application running on the host yields inefficiencies. This document describes a solution for taking the application needs into account in selectively providing IP session continuity and IP address reachability on a per-socket basis.

Table of Contents

1. Introduction

In the context of Mobile IP [RFC5563][RFC6275][RFC5213][RFC5944], following two attributes are defined for the IP service provided to the mobile hosts:

IP session continuity: The ability to maintain an ongoing IP session by keeping the same local end-point IP address throughout the session despite the mobile host changing its point of attachment within the IP network topology. The IP address of the host may change between two independent IP sessions, but that does not jeopardize the IP session continuity. IP session continuity is essential for mobile hosts to maintain ongoing flows without any interruption.

IP address reachability: The ability to maintain the same IP address for an extended period of time. The IP address stays the same across independent IP sessions, and even in the absence of any IP session. The IP address may be published in a long-term registry (e.g., DNS), and it is made available for serving incoming (e.g., TCP) connections. IP address reachability is essential for mobile hosts to use specific/published IP addresses.

Mobile IP is designed to provide both IP session continuity and IP address reachability to mobile hosts. Architectures utilizing these protocols (e.g., 3GPP, 3GPP2, WIMAX) ensure that any mobile host attached to the compliant networks can enjoy these benefits. Any application running on these mobile hosts is subjected to the same treatment with respect to the IP session continuity and IP address reachability.

It should be noted that in reality not every application may need those benefits. IP address reachability is required for applications running as servers (e.g., a web server running on the mobile host). But, a typical client application (e.g., web browser) does not necessarily require IP address reachability. Similarly, IP session continuity is not required for all types of applications either. Applications performing brief communication (e.g., DNS client) can survive without having IP session continuity support.

Achieving IP session continuity and IP address reachability by using Mobile IP incurs some cost. Mobile IP protocol forces the mobile host's IP traffic to traverse a centrally-located router (Home Agent, HA), which incurs additional transmission latency and use of additional network resources, adds to the network CAPEX and OPEX, and decreases the reliability of the network due to the introduction of a single point of failure [RFC7333]. Therefore, IP session continuity and IP address reachability should be be provided only when needed.

Furthermore, when an application needs session continuity, it may be able to satisfy that need by using a solution above the IP layer, such as MPTCP [RFC6824], SIP mobility [RFC3261], or an application-layer mobility solution. Those higher-layer solutions are not subject to the same issues that arise with the use of Mobile IP since they can utilize the most direct data path between the end-points. But, if Mobile IP is being applied to the mobile host, those higher-layer protocols are rendered useless because their operation is inhibited by the Mobile IP. Since Mobile IP ensures that the IP address of the mobile host remains fixed (despite the location and movement of the mobile host), the higher-layer protocols never detect the IP-layer change and never engage in mobility management.

This document proposes a solution for the applications running on the mobile host to indicate whether they need IP session continuity or IP address reachability. The network protocol stack on the mobile host, in conjunction with the network infrastructure, would provide the required type of IP service. It is for the benefit of both the users and the network operators not to engage an extra level of service unless it is absolutely necessary. So it is expected that applications and networks compliant with this specification would utilize this solution to use network resources more efficiently.

2. Notational Conventions

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 [RFC2119].

3. Solution

3.1. Types of IP Addresses

Three types of IP addresses are defined with respect to the mobility management.

- Fixed IP Address

A Fixed IP address is an address with a guarantee to be valid for a very long time, regardless of whether it is being used in any packet to/from the mobile host, or whether or not the mobile host is connected to the network, or whether it moves from one point-of-attachment to another (with a different subnet or IP prefix) while it is connected.

Fixed IP addresses are required by applications that need both IP session continuity and IP address reachability.

- Session-lasting IP Address

A session-lasting IP address is an address with a guarantee to be valid throughout the IP session(s) for which it was requested. It is guaranteed to be valid even after the mobile host had moved from one point-of-attachment to another (with a different subnet or IP prefix).

Session-lasting IP addresses are required by applications that need IP session continuity but do not need IP address reachability.

- Non-persistent IP Address

This type of IP address provides neither IP session continuity nor IP address reachability. The IP address is obtained from the serving IP gateway and it is not maintained across gateway changes. In other words, the IP address may be released and replaced by a new IP address when the IP gateway changes due to the movement of the mobile host.

Applications running as servers at a published IP address require a Fixed IP Address. Long-standing applications (e.g., an SSH session) may also require this type of address. Enterprise applications that connect to an enterprise network via virtual LAN require a Fixed IP Address.

Applications with short-lived transient IP sessions can use Session-lasting IP Addresses. For example: Web browsers.

Applications with very short IP sessions, such as DNS clients and instant messengers, can utilize Non-persistent IP Addresses. Even though they could very well use Fixed or Session-lasting IP Addresses, the transmission latency would be minimized when a Non-persistent IP Addresses are used.

The network creates the desired guarantee (Fixed, Session-lasting or Non-persistent) by either assigning the address prefix (as part of a stateless address generation process), or by assigning an IP address (as part of a stateful IP address generation).

The exact mechanism of prefix or address assignment is outside the scope of this specification.

3.2. Granularity of Selection

The IP address type selection is made on a per-socket granularity. Different parts of the same application may have different needs. For example, control-plane of an application may require a Fixed IP Address in order to stay reachable, whereas data-plane of the same application may be satisfied with a Session-lasting IP Address.

3.3. On Demand Nature

At any point in time, a mobile host may have a combination of IP addresses configured. Zero or more Non-persistent, zero or more Session-lasting, and zero or more Fixed IP addresses may be configured on the IP stack of the host. The combination may be as a result of the host policy, application demand, or a mix of the two.

When an application requires a specific type of IP address and such address is not already configured on the host, the IP stack shall

attempt to configure one. For example, a host may not always have a Session-lasting IP address available. When an application requests one, the IP stack shall make an attempt to configure one by issuing a request to the network (see section Section 3.4 for more details). If the operation fails, the IP stack shall fail the associated socket request. If successful, a Session-lasting IP Address gets configured on the mobile host. If another socket requests a Session-lasting IP address at a later time, the same IP address may be served to that socket as well. When the last socket using the same configured IP address is closed, the IP address may be released or kept for future applications that may be launched and require a Session-lasting IP address.

In some cases it might be preferable for the mobile host to request a new Session-lasting IP address for a new opening of an IP session (even though one was already assigned to the mobile host by the network and might be in use in a different, already active IP session). It is outside the scope of this specification to define criteria for selecting to use available addresses or choose to request new ones. It supports both alternatives (and any combination).

It is outside the scope of this specification to define how the host requests a specific type of address (Fixed, Session-lasting or Non-persistent) and how the network indicates the type of address in its advertisement of IP prefixes or addresses (or in its reply to a request).

The following are matters of policy, which may be dictated by the host itself, the network operator, or the system architecture standard:

- The initial set of IP addresses configured on the host at boot time.

- Permission to grant various types of IP addresses to a requesting application.

- Determination of a default address type when an application does not make any explicit indication, whether it already supports the required API or it is just a legacy application.

3.4. Conveying the Selection

The selection of the address type is conveyed from the applications to the IP stack in order to influence the source address selection algorithm [RFC6724].

The current source address selection algorithm operates on the available set of IP addresses, when selecting an address. According to the proposed solution, if the requested IP address type is not available at the time of the request, the IP stack shall make an attempt to configure one such IP address. The selected IP address shall be compliant with the requested IP address type, whether it is selected among available addresses or dynamically configured. In the absence of a matching type (because it is not available and not configurable on demand), the source address selection algorithm shall return an empty set.

A Socket API-based interface for enabling applications to influence the source address selection algorithm is described in [RFC5014]. That specification defines IPV6_ADDR_PREFERENCES option at the IPPROTO_IPV6 level. That option can be used with setsockopt() and getsockopt() calls to set and get address selection preferences.

Furthermore, that RFC also specifies two flags that relate to IP mobility management: IPV6_PREFER_SRC_HOME and IPV6_PREFER_SRC_COA. These flags are used for influencing the source address selection to prefer either a Home Address or a Care-of Address.

Unfortunately, these flags do not satisfy the aforementioned needs due to the following reasons:

- Current flags indicate a "preference" whereas there is a need for indicating "requirement". Source address selection algorithm does not have to produce an IP address compliant with the "preference" , but it has to produce an IP address compliant with the "requirement".

- Current flags influence the selection made among available IP addresses. The new flags force the IP stack to configure a compliant IP address if none is available at the time of the request.

- The Home vs. Care-of Address distinction is not sufficient to capture the three different types of IP addresses described in Section 2.1.

The following new flags are defined in this document and they shall be used with Socket API in compliance with [RFC5014]:

IPV6_REQUIRE_FIXED_IP /* Require a Fixed IP address as source */

IPV6_REQUIRE_SESSION_LASTING_IP /* Require a Session-lasting IP address as source */

IPV6_REQUIRE_NON_PERSISTENT_IP /* Require a Non-persistent IP address as source */

Only one of these flags may be set on the same socket. If an application attempts to set more than one flag, the most recent setting will be the one in effect.

When any of these new flags is used, the IPV6_PREFER_SRC_HOME and IPV6_PREFER_SRC_COA flags, if used, shall be ignored.

These new flags are used with setsockopt()/getsockopt(), getaddrinfo(), and inet6_is_srcaddr() functions [RFC5014]. Similar to the setsockopt()/getsockopt() calls, the getaddrinfo() call shall also trigger configuration of the required IP address type, if one is not already available. When the new flags are used with getaddrinfo() and the triggered configuration fails, the getaddrinfo() call shall ignore that failure (i.e., not return an error code to indicate that failure). Only the setsockopt() shall return an error when configuration of the requested IP address type fails.

When the IP stack is required to use a source IP address of a specified type, it can perform one of the following: It can use an existing address (if it has one), or it can create a new one from an existing prefix of the desired type. If the host does not already have an IPv6 prefix of the specific type, it can request one from the network.

Using an existing address from an existing prefix is faster but might yield a less optimal route (if a hand-off event occurred since its configuration), on the other hand, acquiring a new IP prefix from the network may take some time (due to signaling exchange with the network) and may fail due to network policies.

An additional new flag - ON_NET flag - enables the application to direct the IP stack whether to use a preconfigured source IP address (if exists) or to request a new IPv6 prefix from the current serving network and configure a new IP address:

IPV6_REQUIRE_SRC_ON_NET /* Set IP stack address allocation behavior */

If set, the IP stack will request a new IPv6 prefix of the desired type from the current serving network and configure a new source IP address. If reset, the IP stack will use a preconfigured one if exists. If there is no preconfigured IP address of the desired type, the IP stack will request a IPv6 prefix from the current serving network (regardless of whether this flag is set or not).

The ON_NET flag must be used together with one of the 3 flags defined above. If ON_NET flag is used without any of these flags, it must be

ignored. If the ON_NET flag is not used, the IP stack is free to either use an existing IP address (if preconfigured) or access the network to configure a new one (the decision is left to implementation).

The following new error codes are also defined in the document and will be used in the Socket API in compliance with [RFC5014].

EAI_REQUIREDIPNOTSUPPORTED /* The network does not support the ability to request that specific IP address type */

EAI_REQUIREDIPFAILED /* The network could not assign that specific IP address type */

4. Usage example

The following example shows the code for creating a Stream socket (TCP) with a Session-Lasting source IP address:

```
#include <sys/socket.h>
#include <netinnet/in.h>

int              s ;                // Socket id
sockaddr_in6     serverAddress ;    // server info for connect()
uint32_t flags = IPV6_REQUIRE_SESSION_LASTING_IP ;
                                    // For requesting a Session-Lasting
                                    // source IP address

  // Create an IPv6 TCP socket
s = socket(AF_INET6, SOCK_STREAM, 0) ;
if (s!=0) {
      // Handle socket creation error
      // ...
} // if socket creation failed
else {

      // Socket creation is successful
      // The application cannot connect yet, since it wants to use a
      // Session-Lasting source IP address It needs to request the
      // Session-Lasting source IP before connecting
    if (setsockopt(s,
                   IPPROTO_IPV6,
                   IPV6_ADDR_PREFERENCE,
                   (void *) flags,
                   sizeof(flags)) == 0){

          // setting session continuity to Session Lasting is successful
```

```
          // The application can connect to the server

          // Set the desired server's port# and IP address
        serverAddress.sin6_port = serverPort ;
        serverAddress.sin6_addr = serverIpAddress ;

          // Connect to the server
        if (connect(s, &serverAddress, sizeof(serverAddress))==0) {
            // connect successful (3-way handshake has been completed
            // with Session-Lasting source address.
            // Continue application functionality
            // ...
        }  // if connect() is successful
        else {
            // connect failed
            // ...
            // Application code that handles connect failure and closes
            // the socket
            // ...
        } // if connect() failed

    }  // if the request of a Session-Lasting source address was successful
    else {
          // application code that does not use Session-lasting IP address
          // The application may either connect without the desired
          // Session-lasting service, or close the socket
          //...
    } // if the socket was successfully created but a Session-Lasting source
      // address was not provided
}  // if socket was created successfully

  // The rest of the application's code
  // ..
```

5. Backwards Compatibility Considerations

Backwards compatibility support is required by the following 3 types of entities:

- The Applications on the mobile host

- The IP stack in the mobile host

- The network infrastructure

5.1. Applications

Legacy applications that do not support the new flags will use the legacy API to the IP stack and will not enjoy On-Demand Mobility feature.

Applications using the new flags must be aware that they may be executed in environments that do not support the On-Demand Mobility feature. Such environments may include legacy IP stack in the mobile host, legacy network infrastructure, or both. In either case, the API will return an error code and the invoking applications must respond with using legacy calls without the On-Demand Mobility feature.

5.2. IP Stack in the Mobile Host

New IP stacks must continue to support all legacy operations. If an application does not use On-Demand Mobility feature, the IP stack must respond in a legacy manner.

If the network infrastructure supports On-Demand Mobility feature, the IP stack should follow the application request: If the application requests a specific address type, the stack should forward this request to the network. If the application does not request an address type, the IP stack must not request an address type and leave it to the network's default behavior to choose the type of the allocated IP prefix. If an IP prefix was already allocated to the host, the IP stack uses it and may not request a new one from the network.

5.3. Network Infrastructure

The network infrastructure may or may not support the On-Demand Mobility feature. How the IP stack on the host and the network infrastructure behave in case of a compatibility issue is outside the scope of this API specification.

6. Summary of New Definitions

The following list summarizes the new constants definitions discussed in this memo:

<netdb.h>	IPV6_REQUIRE_FIXED_IP
<netdb.h>	IPV6_REQUIRE_SESSION_LASTING_IP
<netdb.h>	IPV6_REQUIRE_NON_PERSISTENT_IP
<netdb.h>	IPV6_REQUIRE_SRC_ON_NET
<netdb.h>	EAI_REQUIREDIPNOTSUPPORTED
<netdb.h>	EAI_REQUIREDIPFAILED
<netinet/in.h>	IPV6_REQUIRE_FIXED_IP
<netinet/in.h>	IPV6_REQUIRE_SESSION_LASTING_IP
<netinet/in.h>	IPV6_REQUIRE_NON_PERSISTENT_IP
<netinet/in.h>	IPV6_REQUIRE_SRC_ON_NET
<netinet/in.h>	EAI_REQUIREDIPNOTSUPPORTED
<netinet/in.h>	EAI_REQUIREDIPFAILED

7. Security Considerations

The setting of certain IP address type on a given socket may be restricted to privileged applications. For example, a Fixed IP Address may be provided as a premium service and only certain applications may be allowed to use them. Setting and enforcement of such privileges are outside the scope of this document.

8. IANA Considerations

This document has no IANA considerations.

9. Contributors

This document was merged with [I-D.sijeon-dmm-use-cases-api-source]. We would like to acknowledge the contribution of the following people to that document as well:

Sergio Figueiredo
Altran Research, France
Email: sergio.figueiredo@altran.com

Younghan Kim
Soongsil University, Korea
Email: younghak@ssu.ac.kr

John Kaippallimalil
Huawei, USA
Email: john.kaippallimalil@huawei.com

10. Acknowledgements

We would like to thank Alexandru Petrescu, Jouni Korhonen, Sri Gundavelli, Dave Dolson and Lorenzo Colitti for their valuable comments and suggestions on this work.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5014] Nordmark, E., Chakrabarti, S., and J. Laganier, "IPv6 Socket API for Source Address Selection", RFC 5014, DOI 10.17487/RFC5014, September 2007, <http://www.rfc-editor.org/info/rfc5014>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

11.2. Informative References

[I-D.sijeon-dmm-use-cases-api-source] Jeon, S., Figueiredo, S., Kim, Y., and J. Kaippallimalil, "Use Cases and API Extension for Source IP Address Selection", draft-sijeon-dmm-use-cases-api-source-05 (work in progress), October 2016.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC5563] Leung, K., Dommety, G., Yegani, P., and K. Chowdhury, "WiMAX Forum / 3GPP2 Proxy Mobile IPv4", RFC 5563, DOI 10.17487/RFC5563, February 2010, <http://www.rfc-editor.org/info/rfc5563>.

[RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <http://www.rfc-editor.org/info/rfc5944>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <http://www.rfc-editor.org/info/rfc6824>.

[RFC7333] Chan, H., Ed., Liu, D., Seite, P., Yokota, H., and J. Korhonen, "Requirements for Distributed Mobility Management", RFC 7333, DOI 10.17487/RFC7333, August 2014, <http://www.rfc-editor.org/info/rfc7333>.

Authors' Addresses

Alper Yegin
Actility
Istanbul
Turkey

Email: alper.yegin@actility.com

Danny Moses
Intel Corporation
Petah Tikva
Israel

Email: danny.moses@intel.com

Kisuk Kweon
Samsung
Suwon
South Korea

Email: kisuk.kweon@samsung.com

Jinsung Lee
Samsung
Suwon
South Korea

Email: js81.lee@samsung.com

Jungshin Park
Samsung
Suwon
South Korea

Email: shin02.park@samsung.com

Seil Jeon
Sungkyunkwan University
Suwon
South Korea

Email: seiljeon@skku.edu