Network Working Group	A. Ebalard
Internet-Draft	EADS
Intended status: Informational	September 30, 2010
Expires: April 3, 2011	

TOC

MIPv6 from IPv4-only networks
draft-ebalard-mext-m6t-02

Abstract

MIPv6 [RFC3775] protocol has been designed to work on IPv6 networks: nothing was initially provisioned in the specification to support movement of Mobile Nodes to IPv4-only networks (with or without NAT) or the communication with IPv4 peers.

DSMIPv6 [RFC5555] is the official solution specified to address those needs. It requires IPv4/NAT-awareness by the MIPv6 module, IKE module and IPsec stack. The global approach selected by DSMIPv6 requires changes to implementations and increases complexity.

This memo presents an alternative approach to support operations of MIPv6 mobile nodes from IPv4-only networks. It does not require changes to MIPv6 modules, IKE module and IPsec stack.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

TOC

TOC

1.1. Initial thoughts

MIPv6 [RFC3775] (Johnson, D., Perkins, C., and J. Arkko, “Mobility Support in IPv6,” June 2004.) protocol has been designed to work on IPv6 networks: nothing was initially provisioned in the specification to support movement of Mobile Nodes to IPv4-only networks (with or without NAT) or the communication with IPv4 peers.
In order to address the need to support those use cases, DSMIPv6 [RFC5555] (Soliman, H., “Mobile IPv6 Support for Dual Stack Hosts and Routers,” June 2009.) has been published: it provides a global solution to support the movement of mobile nodes to IPv4-only networks with and without NAT and their communication with IPv4 peers via the Home Agent. The use of DSMIPv6 inhibits Route Optimization mechanism.
The approach selected by DSMIPv6 is to implement IPv4 and NAT awareness in MIPv6, IPsec and IKE componenst on the MN and its HA. Even if this probably has advantages (efficient on-wire format), this IPv4/NAT-awareness also adds a lot of complexity to the MIPv6 process and in its relationships with IPsec and IKE.
Considering the initial problems at hand:

1. Supporting the use of MIPv6 for MN stuck in IPv4-only networks.

2. Supporting the communication of MIPv6 MN with IPv4 peers.

the following separate approaches are possible:

1. Use of a simple tunnel (over IPv4/UDP) between the MN and its HA

2. Use of NAT64 mechanism in the Home Network. Other different solutions may be more suitable.

This document tries and address the first problem by covering the first approach.

1.2. Rationale

TOC

MIPv6 works fine when IPv6 connectivity is available because its logic remains quite simple. The absence of NAT on IPv6 networks (i.e. flat routing) has made it possible to greatly simplify the protocol and its deployment, compared to MIPv4.
IPsec and IKE have seen limited deployment in IPv4 networks (in favor of TLS) partly due to the difficulty or inability to use them in ubiquitous NAT environments. When IPv6 connectivity is available, IPsec and IKE work and interoperate easily. The interactions with MIPv6 in

order to secure the operation of the protocol are quite simple (see [MIGRATE] (Ebalard, A. and S. Decugis, “PF_KEY Extension as an Interface between Mobile IPv6 and IPsec/IKE,” August 2008.)). Basically, if it is possible to mask the existence of IPv4 and NAT to MIPv6, IPsec and IKE, it may be possible to allow its use from IPv4-only networks without invasive and costly changes in the implementation. This is what m6t protocol aims at.
Additionally, the rationale for this work also results from:

- Considerations on the limited set of flows associated with MIPv6 protocol: when RO is not used, MN's traffic always go via the HA.
- The use of Teredo by the author on IPv4 networks as an alternative solution to DSMIPv6 (due to its unavailability on most platform).
- Considerations on the architectural requirements and drawbacks of Teredo (qualification procedure, reliance on Teredo servers, MTU considerations, ...) in the specific context of a MIPv6 MN.

2. Problems and solutions

TOC

As discussed above, the idea is to provide IPv6 connectivity via a simple tunnel over IPv4 and UDP to the HA. On the MN, the tunnel interface hides the existence of IPv4/NAT and makes it possible to operate MIPv6 in a transparent fashion. On a gateway in front of the HA (or on the HA), this provides the same benefit. Nonetheless, creating, using and maintaining this tunnel both on the MN and the gateway in front of the HA requires some care. Problems and selected solutions are discussed below.

2.1. UDP port to use on the tunnel GW

TOC

For a given deployment, clients of the tunnel gateway (MN or MR) need to access it on a known port. In some environments, it may be beneficial for admins to be able to select that specific port. For that reason, it does not seems worth asking IANA to allocate a dedicated port for that purpose.
Basically, the tunnel mechanism described in this memo is a custom version (MIPv6-oriented) of what STUN [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, “Session Traversal Utilities for NAT (STUN),” October 2008.) or Teredo [RFC4380] (Huitema, C., “Teredo:

Tunneling IPv6 over UDP through Network Address Translations (NATs)," February 2006.) provide. As the tunnel GW is a closed service dedicated to the MN, it may be possible to have the GW listen on STUN or Teredo ports (respectively 3478/udp and 3544/udp).

As filtering may be in place in the IPv4 networks the MN found themselves, it could be possible to use a destination port associated with a protocol which is usually authorized, like 4500/udp, 500/udp or 53/udp.

As the choice of the specific ports to be used for a given MN community highly depends on the habits of the MN and the foreign networks they are usually in, the choice of the port on which the tunnel gateway should listen is left as a local decision. The service of the gateway may be available on different ports.

2.2. MN's CoA on the tunnel interface

TOC

A MN needs an IPv6 address to use as source for the packets it sends to the HA, i.e. to be configured on the MN's tunnel interface. Because this address will be the one seen by the HA for the MN, it needs to be unique among all the MN. Additionally, all the addresses used by the MN for their tunnel connectivity needs to be routed from the HA to the tunnel gateway.

The MN has a unique address available: its HoA. Nonetheless, it is not directly usable over the tunnel because the associated prefix is also the home prefix. Using this would create additional routing complexities on the HA and may create additional attack vectors.

As the addresses used by the MN will never be directly routed over the Internet, it is possible to use a Unique Local prefix for that purpose. Generation of an ULA prefix should be performed as described in [RFC4193] (Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses," October 2005.). The remaining of the address after the 48 bits ULA prefix are set as follows.

The 16 bits directly following the first 48 bits of ULA prefix are available for the administrator to configure additional subnets if needed.

The interface identifier is constructed in the following fashion by a MN:

- the first 16 bits are filled with the MN ID, a value known by the MN. It does not need to be known by the tunnel GW or the HA. It is just required to be unique among the set of MN using the /64 prefix.

- the following 48 bits are filled with random bits by the MN each time it changes its point of attachment in the IPv4 infrastructure. The rationale for this random value is given later in the document.

From a theroretical standpoint, the combination of the subnet ID and the MN ID allows an administrator to support up to 2**32 MN, i.e. those are not a limiting factors.
In the end, the tunneling component on the MN should be provided directly with the /80 prefix.
To summarize, the prefix has the following format:

```
+---------------------+---------+---------+-------------------+
|    /48 ULA prefix   | 16 bits | 16 bits |      48 bits      |
|                     |subnet ID|  MN ID  |    random value   |
+---------------------+---------+---------+-------------------+
```

Note that the addressing scheme based on ULA provided above is only indicative: nothing prevents an administrator to use an internal prefix instead of an ULA prefix if it better suits her needs. The only requirement for the operation of the protocol is the per-MN uniqueness of the prefix among the set of MN.

2.3. Maintaining NAT states

TOC

The MN needs to maintain states in the NAT GW which handles its traffic in order to remain reachable from the HA (i.e. by peers which use the MN-HA tunnel).
The MN basically needs to exchange keep-alive messages with the tunnel GW in order for the states in the NAT GW to be maintained. Those exchanges are required only when no traffic is exchanged between the MN and its HA.
If the MN has not sent and received traffic via the tunnel for 30 seconds (default refresh interval borrowed from Teredo specification), the tunneling component on the MN will send an empty IPv6 packet (Next Header in IPv6 header set to 59) via the tunnel. The IPv6 source address of the packet is set to the tunnel interface address and the destination address is set to fe80::1.
When the tunnel component receives the IPv6 packet over the tunnel, the specific destination address (fe80::1) and the absence of upper layer (next header set to 59) indicate the packet is a keep-alive message. It verifies an active state exists for the source address in the packet and that the IPv4 parameters match the state. If this is the case, it replies with a similar message but with the addresses reversed.
Otherwise, it silently drops the packet.

TOC

2.4. Route management on the tunnel GW

Route management on the tunnel gateway is implementation dependent but reflects the status of states. The tunnel gateway maintains states associating the IPv6 address of a MN with its IPv4 information (IPv4 address and UDP port) to be able to forward IPv6 traffic over UDP from the HA to the peer. States can be either temporary or active.

A temporary state is created when a valid UDP-encapsulated IPv6 packet is received from a MN and the IPv6 address is not already known. Such a state has a short lifetime (some milli-seconds to a few seconds) and either become active or is deleted.

A temporary state becomes active when a packet from the HA to the MN is processed by the tunnel gateway. An active state has a longer lifetime which is extended each time traffic is exchanged with the MN (data or keep-alive messages).

In practice, the protocol does not offer a way for a MN to deregister a state. When the MN registers from a new IPv4-only network, it generates a new IPv6 address and uses it. As no traffic is exchanged after some point via the tunnel for that address, it is garbage collected automatically.

For security reasons discussed later in the document, address-related information (IPv6 address, IPv4 address, UDP port) in states (active or temporary) are never updated. More specifically, if the source address of an IPv6 packet received over UDP matches an existing state but there is a mismatch in IPv4-related information, the packet should be dropped and no modification to existing should occur.

2.5. Multiple tunnel interfaces on a MN

TOC

A MN may want to use multiple tunnel interfaces simultaneously, via the same tunnel gateway. The addressing scheme and the protocol does not prevent that. This is mainly a matter of preference among the different interfaces. Additionally, if the same prefix, subnet ID and MN ID are used for all the tunnel interfaces of the MN (i.e. the same /80 prefix), then some local syncrhonization is required to prevent the 48 simultaneous use of identical 48 bits random values.

2.6. MTU considerations

TOC

Proposed approach is based on UDP tunneling. As for Teredo protocol, implementation of this specification SHOULD NOT set the DF bit of the encapsulating IPv4 header.

Nonetheless, packets exchanged between the MN and its HA via the tunnel may undergo PMTUD both at the IPv6 level (between the tunnel GW and the

HA) and at the IPv4 level (between the MN and the tunnel GW). When implementing the mechanism, PMTUD support should be one of the first things to validate.

Teredo does not provide a mechanism for a Teredo client to control the MTU of its Teredo interface and the one on the relay. It is statically set to 1280, which is a safe bet to counter the possible filtering of ICMP messages in the IPv4 internet.

In the protocol described in this memo, a different approach is selected. The MTU of the tunnel interface is initially set to a default value of 1472 bytes on both sides (on the MN and on its tunnel GW). Reception of ICMP Packet too big messages from IPv4 routers on the path may be used to locally update the MTU of the path.

For use in specific environments, implementations should provide a mean for administrator to set the default MTU to a different value.

2.7. Miscellanous

TOC

This subsection covers additional miscellanous points:

*HA reachability via the tunnel GW: monitoring the reachability of the HA via the tunnel gateway is outside the scope of the protcol but may be implemented if needed. The reachability will usually depend on the UDP ports used for contacting the GW, i.e. on the filtering implemented by the IPv4 network (probably by the NAT GW).

*Teredo automatic sunset equivalent: Teredo provides an automatic sunset mechanism. The protocol does not provide one. This is left has an implementation decision. It should be noted that the use of the tunnel from a fast IPv4-only network (ethernet access) may be more efficient in some case than the use of a native IPv6-enabled connection mean via an slower network (3G).

*Usability from a public address: when a public IPv4 address is available at the MN, other transition mechanisms (e.g. 6to4) may be used to get IPv6 connectivity. From an encapsulation standpoint, 6to4 is more efficient (8 bytes gain per packet). Additionally, unlike 6to4, current approach requires (no matter the characteristics of the available IPv4 address) to implement and use the keep-alive mechanism described above. Nonetheless, compared to 6to4, proposed approach may provide a shorter path to join the HA via the tunnel GW.

*Behavior on IPv6-enabled networks: the mechanism is useful from IPv4-only networks (with or without NAT) but also works when IPv6 and IPv4 are both available. When usable (i.e. IPv4 is available), the mechanism provides an additional IPv6 interface on the system that the MIPv6 module can consider. The activation of the mechanism does not inhibit other existing interfaces.

3. Protocol operations

TOC

This short section describes the operation of the protocol.
The m6t tunnel component on a MN is initially configured with:

- *The IPv4 address (and UDP port) of the tunnel gateway associated with its HA

- *A /80 prefix created as described in previous section.

When a MN gets IPv4 connectivity (IPv4 address and route), it selects a random 48 bits value. It concatenates this values to its configured /80 prefix and installs the address on the tunnel interface. A default route is configured for IPv6 traffic to flow via this tunnel. A UDP socket is created for further emission to the GW IPv4 address (and UDP port).
At that point, if the MIPv6 module decides to use the newly configured address as CoA (e.g. this is the only one available at the moment on the MN), it sends an IPsec-protected Binding Update message to its HA (or an IKE packet for negotiating IPsec SA for protecting the Binding Update message). The packet is routed via the tunnel interface and sent to the tunnel GW via the UDP socket.
The tunnel GW in front (or on) the HA receives the IPv4 packet and processes it after some sanity checks on the addresses (as described in previous section). It creates a temporary state containing the addresses and ports and forwards the packet to the final destination (the HA).
The HA processes the packet and sends a reply (IPsec protected BA, IKE packet, ...) to the MN. The tunnel gateway handles that packet and first verifies if an active state exists for that destination. If it is the case, the packet is forwarded to the remote IPv4 NAT GW using the parameters available in the state. If no active state exists, then, a lookup at the temporary states is performed: if one is found, it is marked active and used.

TOC

4. Interactions with MIPv6

per se, m6t does not have specific interactions with MIPv6 (or IPsec/IKE). When a MN uses m6t, it benefits from additional and transparent IPv6 connectivity to its HA. The decision to use this additional connectivity is left to the MIPv6 module based on its configuration.
On the tunnel gateway, the only assumption made by m6t protocol is the expected reply from the HA to the first packet sent by a MN from a freshly generated m6t address. This is required for validation of temporary state (created by first packet), i.e. switch from temporary to valid. If the first packet is the beginning of an IKE negotiation, reply will create the state. More often, first packet will be an IPsec-protected BU to which the HA will reply with by an IPsec-protected BA.

5. Pros and cons

TOC

This short section provides the pros and cons of the approach.
The approach described in this memo is simple, standalone and transparent both on the MN and the HA. It does not require any changes to the MIPv6, IKE or IPsec code running on both compoments. It is fully compatible with NEMO.
Unlike DSMIPv6, as the approach does not modify MIPv6 implementations, it does not extend MIPv6 protocol to support communications with IPv4 peers. NAT64 may be used at the edge of the Home Network to allow communications with IPv4 peers. Other more suitable solutions may exist. Additionally, DSMIPv6 provides IPv4 address stability to the MN (i.e. an IPv4 HoA). The approach described in this memo does not provide that either. In most environments, IPv4 address stability is usually not needed or useful because nodes are usually provisioned with private IPv4 address and have for that reason limited reachability outside their private domain.

6. Security Considerations

TOC

6.1. Preventing DoS

TOC

TOC

6.1.1. Against the MN

Nothing is done to prevent an active attacker located on the path between the MN and the tunnel GW with the ability to access and modify the traffic. Because such an attacker has access to all the traffic exchanged with the tunnel GW and the HA, there is not much that can be done. A similar threat exist in [RFC5555] (Soliman, H., “Mobile IPv6 Support for Dual Stack Hosts and Routers,” June 2009.)

Nonetheless, if an attacker has read acccess and is able to send traffic in parallel of a MN (e.g. case of a MN connected to an unprotected wireless network), m6t protocol provides protection to the MN. More precisely, the use of a new IPv6 random address by the MN each time it changes its point of attachment creates a new state on the remote m6t tunnel gateway, associated with its mapped IPv4 address and port. Because existing m6t bindings between an IPv6 care-of address and an IPv4 address and UDP port cannot be replaced or modified, it is not possible for the attacker to cause the m6t gateway to redirect the MN's traffic to a different IPv4 address and UDP port.

Note that nothing prevents an implementation of changing its IPv6 address more frequently.

6.1.2. Against the HA

TOC

The access via the tunnel GW is not expected to open additional DoS possibilities against the HA.

6.1.3. Against the GW

TOC

The Gateway is basically a single point of failure for the MN which use it as their connection mean to their HA from IPv4 network. If an attacker manages to perform a DoS against the tunnel GW, this may prevent legitimate peers to access their HA.

In order to prevent that, care is needed when implementing the tunnel GW. Rate limiting and garbage collection can be used for that purpose.

When a packet is received from a new IPv4 address, some initial sanity checks are performed on the packet (IPv6 destination address is HA address, IPv6 address is from the configured ULA prefix, ...).

After that initial validation of the packet, a state is recorded, containing the IPv4 source @, the UDP source port and the IPv6 source address of the packet. Such a state should be associated a very low (few ms) garbage collection timer. The value should be configured based on the environment in which the tunnel GW is deployed: it should basically match the expected processing time of a packet on the HA (first IKE packet, BU, ...) and the RTT between the gateway and the HA.

Creation of such states should also be rate-limited on a per IPv4 address basis (not considering the source port): this would prevent an individual attacker to perform a DoS against the service. It would basically require her to have access to a set of bot. The upper limit to use per IPv4 address is not specified in this memo. It should be configured locally based on the environment. It may be possible for some deployments to expect multiple simultaneous connections originatig from the same NAT GW.

When a packet is received by the tunnel GW from the HA, a lookup is performed to check if a valid state exists for the IPv6 destination address. If one is found, it provides the IPv4 destination address and port of the NAT GW associated with that IPv6 address. If it does not yet exist, then a lookup for the address in the pending temporary states is performed. If one is found, it becomes active. The packet is forwarded to the remote GW on the given IPv4 address and UDP port. Garbage collecting for that state is activated: the purpose is for state to be removed if no traffic is exchanged in both direction for more than the expected NAT GW timeout. The maintenance of the state is left to the MN as described in a previous section of the document.

6.1.4. Against the Infrastructure

TOC

An attacker may want to use the tunnel GW to create amplification attacks against elements of the IPv4 infrastructure. For the reasons explained above, an attacker will not be able to redirect a MN's traffic to a controlled address. But the HA is still expected to reply to some unauthenticated probes. For instance, if an attacker targets it's IKE daemon via the tunnel GW, it will get an answer. If it is able to spoof its source address, the response will be sent to the spoofed source address. This kind of attack is still already possible on the IPv4 infrastructure and the attacker would not get additional gain in using the tunnel GW for such a scenario.

7. Implementation

TOC

A (proof of concept) implementation (client and tunnel gateway) has been developed by the author for Linux. It is available at http://natisbad.org/m6t/ under a GPLv2 license. The implementation interoperates transparently and out of the box with unmodified versions of UMIP (Mobile IPv6 for Linux, see http://umip.org/).

TOC

8. IANA Considerations

This document has no IANA actions.

9. Acknowledgements

TOC

The author would like to thank Julien Laganier, Romain Kuntz and Christophe Regouby for interesting technical discussions and reviews of the draft.
This document was generated using xml2rfc.

10. References

TOC

10.1. Normative References

TOC

[RFC3775]	Johnson, D., Perkins, C., and J. Arkko, “Mobility Support in IPv6,” RFC 3775, June 2004 (TXT).
[RFC4193]	Hinden, R. and B. Haberman, “Unique Local IPv6 Unicast Addresses,” RFC 4193, October 2005 (TXT).
[RFC5555]	Soliman, H., “Mobile IPv6 Support for Dual Stack Hosts and Routers,” RFC 5555, June 2009 (TXT).

10.2. Informative References

TOC

[MIGRATE]	Ebalard, A. and S. Decugis, “PF_KEY Extension as an Interface between Mobile IPv6 and IPsec/IKE,” draft-ebalard-mext-pfkey-enhanced-migrate-00 (work in progress), August 2008 (TXT).
[RFC4380]	Huitema, C., “Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs),” RFC 4380, February 2006 (TXT).
[RFC5389]	Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, “Session Traversal Utilities for NAT (STUN),” RFC 5389, October 2008 (TXT).

Author's Address

TOC

Arnaud Ebalard
EADS Innovation Works
12, rue Pasteur - BP76
Suresnes 92152
France
Email: arno@natisbad.org