Network Working Group
Internet-Draft
Obsoletes: 5203 (if approved)
Intended status: Standards Track
Expires: February 21, 2011

J. Laganier
QUALCOMM Inc.
T. Koponen
HIIT
L. Eggert
Nokia

August 20, 2010

Host Identity Protocol (HIP) Registration Extension
draft-ietf-hip-rfc5203-bis-00

Abstract

This document specifies a registration mechanism for the Host Identity Protocol (HIP) that allows hosts to register with services, such as HIP rendezvous servers or middleboxes.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on February 21, 2011.

Copyright Notice

1. Introduction

This document specifies an extension to the Host Identity Protocol (HIP) [RFC5201]. The extension provides a generic means for a host to register with a service. The service may, for example, be a HIP rendezvous server [RFC5204] or a middlebox [RFC3234].

This document makes no further assumptions about the exact type of service. Likewise, this document does not specify any mechanisms to discover the presence of specific services or means to interact with them after registration. Future documents may describe those operations.

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 RFC 2119 [RFC2119].

2. Terminology

In addition to the terminology defined in the HIP Architecture [RFC4423], the HIP specification [RFC5201], and the HIP Rendezvous Extension [RFC5204], this document defines and uses the following terms:

Requester:
: a HIP node registering with a HIP registrar to request registration for a service.

Registrar:
: a HIP node offering registration for one or more services.

Service:
: a facility that provides requesters with new capabilities or functionalities operating at the HIP layer. Examples include firewalls that support HIP traversal or HIP rendezvous servers.

Registration:
: shared state stored by a requester and a registrar, allowing the requester to benefit from one or more HIP services offered by the registrar. Each registration has an associated finite lifetime. Requesters can extend established registrations through re-registration (i.e., perform a refresh).

Registration Type:
an identifier for a given service in the registration protocol. For example, the rendezvous service is identified by a specific registration type.

3. HIP Registration Extension Overview

This document does not specify the means by which a requester discovers the availability of a service, or how a requester locates a registrar. After a requester has discovered a registrar, it either initiates HIP base exchange or uses an existing HIP association with the registrar. In both cases, registrars use additional parameters, which the remainder of this document defines, to announce their quality and grant or refuse registration. Requesters use corresponding parameters to register with the service. Both the registrar and the requester MAY also include in the messages exchanged additional HIP parameters specific to the registration type implicated. Other documents will define parameters and how they shall be used. The following sections describe the differences between this registration handshake and the standard HIP base exchange [RFC5201].

3.1. Registrar Announcing Its Ability

A host that is capable and willing to act as a registrar SHOULD include a REG_INFO parameter in the R1 packets it sends during all base exchanges. If it is currently unable to provide services due to transient conditions, it SHOULD include an empty REG_INFO, i.e., one with no services listed. If services can be provided later, it SHOULD send UPDATE packets indicating the current set of services available in a new REG_INFO parameter to all hosts it is associated with.

3.2. Requester Requesting Registration

To request registration with a service, a requester constructs and includes a corresponding REG_REQUEST parameter in an I2 or UPDATE packet it sends to the registrar.

If the requester has no HIP association established with the registrar, it SHOULD send the REG_REQUEST at the earliest possibility, i.e., in the I2 packet. This minimizes the number of packets that need to be exchanged with the registrar. A registrar MAY end a HIP association that does not carry a REG_REQUEST by including a NOTIFY with the type REG_REQUIRED in the R2. In this case, no HIP association is created between the hosts. The REG_REQUIRED notification error type is 51.

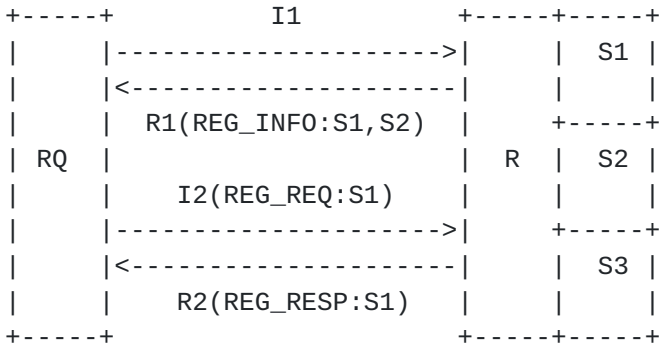
3.3. Registrar Granting or Refusing Service(s) Registration

Once registration has been requested, the registrar is able to authenticate the requester based on the host identity included in I2. It then verifies that the host identity is authorized to register with the requested service(s), based on local policies. The details of this authorization procedure depend on the type of requested service(s) and on the local policies of the registrar, and are therefore not further specified in this document.

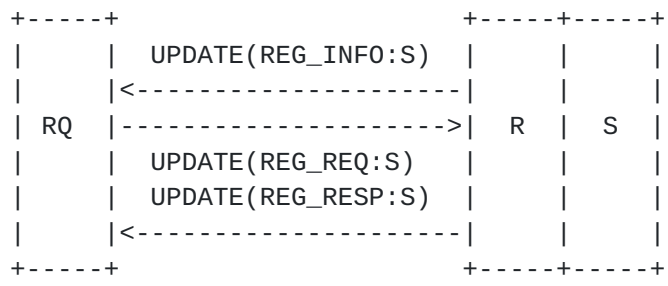
After authorization, the registrar includes a REG_RESPONSE parameter in its response, which contains the service type(s) for which it has authorized registration, and zero or more REG_FAILED parameters containing the service type(s) for which it has not authorized registration or registration has failed for other reasons. This response can be either an R2 or an UPDATE message, respectively, depending on whether the registration was requested during the base exchange, or using an existing association. In particular, REG_FAILED with a failure type of zero indicates the service(s) type(s) that require further credentials for registration.

If the registrar requires further authorization and the requester has additional credentials available, the requester SHOULD try to register again with the service after the HIP association has been established. The precise means of establishing and verifying credentials are beyond the scope of this document and are expected to be defined in other documents.

Successful processing of a REG_RESPONSE parameter creates registration state at the requester. In a similar manner, successful processing of a REG_REQUEST parameter creates registration state at the registrar and possibly at the service. Both the requester and registrar can cancel a registration before it expires, if the services afforded by a registration are no longer needed by the requester, or cannot be provided any longer by the registrar (for instance, because its configuration has changed).



A requester (RQ) registers with a registrar (R) of services (S1) and (S2), with which it has no current HIP association.



A requester (RQ) registers with a registrar (R) of services (S), with which it currently has a HIP association established.

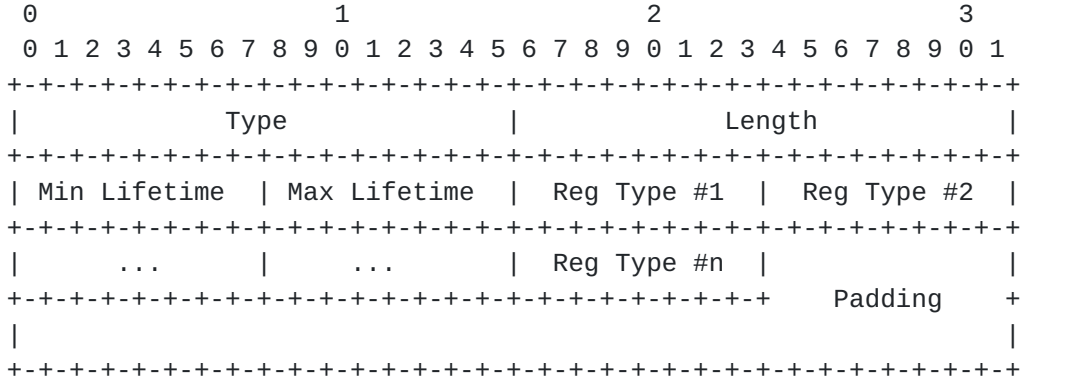
4. Parameter Formats and Processing

This section describes the format and processing of the new parameters introduced by the HIP registration extension.

4.1. Encoding Registration Lifetimes with Exponents

The HIP registration uses an exponential encoding of registration lifetimes. This allows compact encoding of 255 different lifetime values ranging from 4 ms to 178 days into an 8-bit integer field. The lifetime exponent field used throughout this document MUST be interpreted as representing the lifetime value 2^((lifetime - 64)/8) seconds.

4.2. REG_INFO

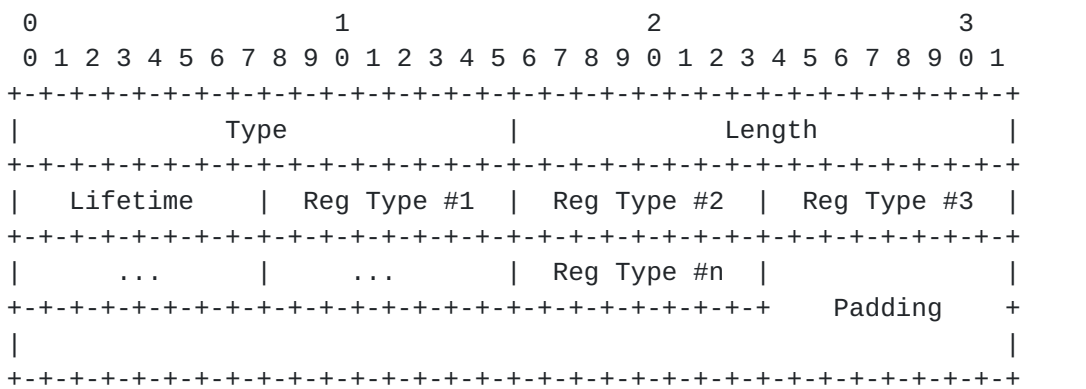


Type	930
Length	Length in octets, excluding Type, Length, and Padding.
Min Lifetime	Minimum registration lifetime.
Max Lifetime	Maximum registration lifetime.
Reg Type	The registration types offered by the registrar.

Other documents will define specific values for registration types. See Section 7 for more information.

Registrars include the parameter in R1 packets in order to announce their registration capabilities. The registrar SHOULD include the parameter in UPDATE packets when its service offering has changed. HIP_SIGNATURE_2 protects the parameter within the R1 packets.

4.3. REG_REQUEST



Type	932
Length	Length in octets, excluding Type, Length, and Padding.
Lifetime	Requested registration lifetime.
Reg Type	The preferred registration types in order of preference.

Other documents will define specific values for registration types.

See Section 7 for more information.

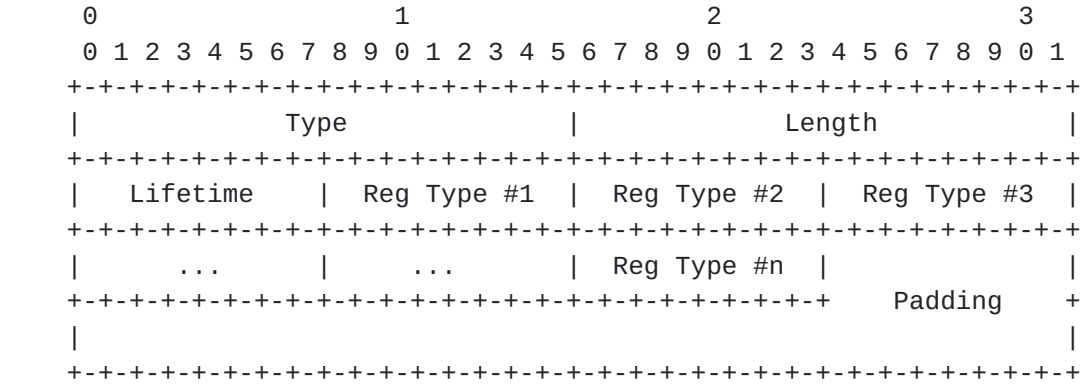
A requester includes the REG_REQUEST parameter in I2 or UPDATE packets to register with a registrar's service(s). If the REG_REQUEST parameter is in an UPDATE packet, the registrar MUST NOT modify the registrations of registration types that are not listed in the parameter. Moreover, the requester MUST NOT include the parameter unless the registrar's R1 packet or latest received UPDATE packet has contained a REG_INFO parameter with the requested registration types.

The requester MUST NOT include more than one REG_REQUEST parameter in its I2 or UPDATE packets, while the registrar MUST be able to process one or more REG_REQUEST parameters in received I2 or UPDATE packets.

When the registrar receives a registration with a lifetime that is either smaller or greater than the minimum or maximum lifetime, respectively, then it SHOULD grant the registration for the minimum or maximum lifetime, respectively.

HIP_SIGNATURE protects the parameter within the I2 and UPDATE packets.

4.4. REG_RESPONSE



Type 934
Length Length in octets, excluding Type, Length, and Padding.
Lifetime Granted registration lifetime.
Reg Type The granted registration types in order of preference.

Other documents will define specific values for registration types. See Section 7 for more information.

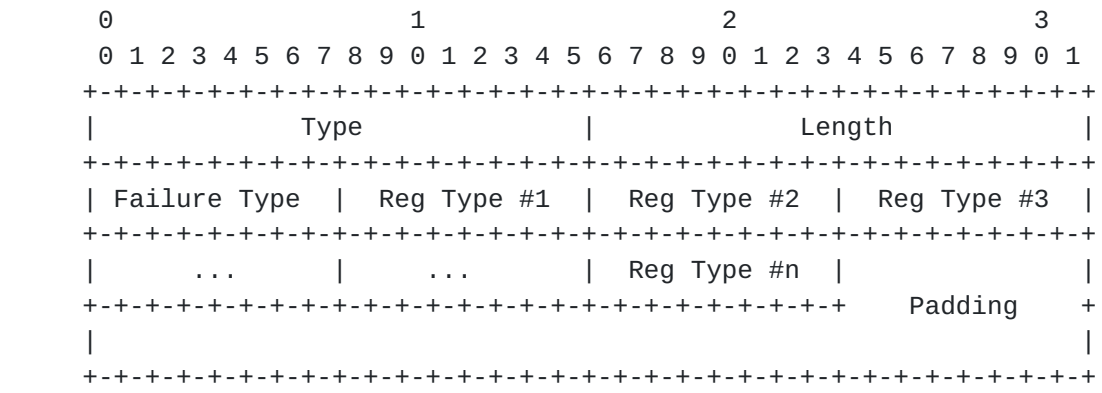
The registrar SHOULD includes an REG_RESPONSE parameter in its R2 or UPDATE packet only if a registration has successfully completed.

The registrar MUST NOT include more than one REG_RESPONSE parameter in its R2 or UPDATE packets, while the requester MUST be able to process one or more REG_RESPONSE parameters in received R2 or UPDATE packets.

The requester MUST be prepared to receive any registration lifetime, including ones beyond the minimum and maximum lifetime indicated in the REG_INFO parameter. It MUST NOT expect that the returned lifetime will be the requested one, even when the requested lifetime falls within the announced minimum and maximum.

HIP_SIGNATURE protects the parameter within the R2 and UPDATE packets.

4.5. REG_FAILED



Type	936
Length	Length in octets, excluding Type, Length, and Padding.
Failure Type	Reason for failure.
Reg Type	The registration types that failed with the specified reason.

Failure Type	Reason
0	Registration requires additional credentials
1	Registration type unavailable
2-200	Unassigned
201-255	Reserved by IANA for private use

Other documents will define specific values for registration types. See Section 7 for more information.

A failure type of zero means a registrar requires additional credentials to authorize a requester to register with the registration types listed in the parameter. A failure type of one means that the requested service type is unavailable at the

registrar. Failure types other than zero (0) and one (1) have not been defined.

The registrar SHOULD include the REG_FAILED parameter in its R2 or UPDATE packet, if registration with the registration types listed has not completed successfully and a requester is asked to try again with additional credentials.

HIP_SIGNATURE protects the parameter within the R2 and UPDATE packets.

5. Establishing and Maintaining Registrations

Establishing and/or maintaining a registration may require additional information not available in the transmitted REG_REQUEST or REG_RESPONSE parameters. Therefore, registration type definitions MAY define dependencies for HIP parameters that are not defined in this document. Their semantics are subject to the specific registration type specifications.

The minimum lifetime both registrars and requesters MUST support is 10 seconds, while they SHOULD support a maximum lifetime of 120 seconds, at least. These values define a baseline for the specification of services based on the registration system. They were chosen to be neither too short nor too long, and to accommodate for existing timeouts of state established in middleboxes (e.g., NATs and firewalls.)

A zero lifetime is reserved for canceling purposes. Requesting a zero lifetime for a registration type is equal to canceling the registration of that type. A requester MAY cancel a registration before it expires by sending a REG_REQ to the registrar with a zero lifetime. A registrar SHOULD respond and grant a registration with a zero lifetime. A registrar (and an attached service) MAY cancel a registration before it expires, at its own discretion. However, if it does so, it SHOULD send a REG_RESPONSE with a zero lifetime to all registered requesters.

6. Security Considerations

This section discusses the threats on the HIP registration protocol, and their implications on the overall security of HIP. In particular, it argues that the extensions described in this document do not introduce additional threats to HIP.

The extensions described in this document rely on the HIP base exchange and do not modify its security characteristics, e.g., digital signatures or HMAC. Hence, the only threat introduced by

these extensions is related to the creation of soft registration state at the registrar.

Registrars act on a voluntary basis and are willing to accept being a responder and then to create HIP associations with a number of previously unknown hosts. Because they have to store HIP association state anyway, adding a certain amount of time-limited HIP registration state should not introduce any serious additional threats, especially because HIP registrars may cancel registrations at any time at their own discretion, e.g., because of resource constraints during an attack.

7. IANA Considerations

This section is to be interpreted according to the Guidelines for Writing an IANA Considerations Section in RFCs [RFC2434].

This document updates the IANA Registry for HIP Parameter Types by assigning new HIP Parameter Types values for the new HIP Parameters defined in this document:

- REG_INFO (defined in Section 4.2)
- REG_REQUEST (defined in Section 4.3)
- REG_RESPONSE (defined in Section 4.4)
- REG_FAILED (defined in Section 4.5)

IANA has allocated the Notify Message Type code 51 for the REG_REQUIRED notification error type in the Notify Message Type registry.

IANA has opened a new registry for registration types. This document does not define registration types but makes the following reservations:

Reg Type	Service
0-200	Unassigned
201-255	Reserved by IANA for private use

Adding a new type requires new IETF specifications.

IANA has opened a new registry for registration failure types. This document makes the following failure type definitions and reservations:

Failure Type	Reason
0	Registration requires additional credentials
1	Registration type unavailable
2-200	Unassigned
201-255	Reserved by IANA for private use

Adding a new type requires new IETF specifications.

8. Acknowledgments

The following people (in alphabetical order) have provided thoughtful and helpful discussions and/or suggestions that have helped to improve this document: Jeffrey Ahrenholz, Miriam Esteban, Mika Kousa, Pekka Nikander, and Hannes Tschofenig.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., Ed., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

9.2. Informative References

[RFC3234] Carpenter, B. and S. Brim, "Middleboxes: Taxonomy and Issues", RFC 3234, February 2002.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.

[RFC5204] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 5204, April 2008.

Authors' Addresses

Julien Laganier
QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121
USA

Phone: +1 858 858 3538
EMail: julienl@qualcomm.com

Teemu Koponen
Helsinki Institute for Information Technology
Advanced Research Unit (ARU)
P.O. Box 9800
Helsinki FIN-02015-HUT
Finland

Phone: +358 9 45 1
EMail: teemu.koponen@iki.fi
URI: http://www.hiit.fi/

Lars Eggert
Nokia Research Center
P.O. Box 407
Nokia Group 00045
Finland

Phone: +358 50 48 24461
EMail: lars.eggert@nokia.com
URI: http://research.nokia.com/people/lars_eggert/