DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2014

S. Jiang, Ed.
Huawei Technologies Co., Ltd
S. Shen
CNNIC
D. Zhang
Huawei Technologies Co., Ltd
February 14, 2014

Secure DHCPv6 with Public Key
draft-ietf-dhc-sedhcpv6-01

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) enables DHCPv6 servers to pass configuration parameters. It offers configuration flexibility. If not secured, DHCPv6 is vulnerable to various attacks, particularly spoofing attacks. This document analyzes the security issues of DHCPv6 and specifies a Secure DHCPv6 mechanism for communication between DHCPv6 client and server. This mechanism is based on public/private key pairs. The authority of the sender may depend on either pre-configuration mechanism or Public Key Infrastructure.

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 August 18, 2014.

Copyright Notice

Table of Contents

1. Introduction

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6, [RFC3315]) enables DHCPv6 servers to pass configuration parameters. It offers configuration flexibility. If not secured, DHCPv6 is vulnerable to various attacks, particularly spoofing attacks.

This document analyzes the security issues of DHCPv6 in details. This document provides mechanisms for improving the security of DHCPv6 between client and server:

- the identity of a DHCPv6 message sender, which can be a DHCPv6 server or a client, can be verified by a recipient.

- the integrity of DHCPv6 messages can be checked by the recipient of the message.

Note: this secure mechanism in this document does not protect the relay-relevant options, either added by a relay agent toward a server or added by a server toward a relay agent, are considered less vulnerable, because they are only transported within operator networks. Communication between a server and a relay agent, and communication between relay agents, may be secured through the use of IPsec, as described in section 21.1 in [RFC3315].

The security mechanisms specified in this document is based on self-generated public/private key pairs. It also integrates timestamps for anti-replay. The authentication procedure defined in this document may depend on either deployed Public Key Infrastructure (PKI, [RFC5280]) or pre-configured sender's public key. However, the deployment of PKI or pre-configuration is out of the scope.

Secure DHCPv6 is applicable in environments where physical security on the link is not assured (such as over wireless) and attacks on DHCPv6 are a concern.

2. Requirements Language and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

3. Security Overview of DHCPv6

DHCPv6 is a client/server protocol that provides managed configuration of devices. It enables DHCPv6 server to automatically configure relevant network parameters on clients. In the basic DHCPv6 specification [RFC3315], security of DHCPv6 message can be improved.

The basic DHCPv6 specifications can optionally authenticate the origin of messages and validate the integrity of messages using an authentication option with a symmetric key pair. [RFC3315] relies on pre-established secret keys. For any kind of meaningful security, each DHCPv6 client would need to be configured with its own secret key; [RFC3315] provides no mechanism for doing this.

For the key of the hash function, there are two key management mechanisms. Firstly, the key management is done out of band, usually through some manual process. For example, operators can set up a key database for both servers and clients which the client obtains a key before running DHCPv6.

Manual key distribution runs counter to the goal of minimizing the configuration data needed at each host. [RFC3315] provides an additional mechanism for preventing off-network timing attacks using the Reconfigure message: the Reconfigure Key authentication method. However, this method provides no message integrity or source integrity check. This key is transmitted in plaintext.

In comparison, the public/private key security mechanism allows the keys to be generated by the sender, and allows the public key database on the recipient to be populated opportunistically or manually, depending on the degree of confidence desired in a specific application. PKI security mechanism is simpler in the local key management respect.

4. Secure DHCPv6 Overview

To solve the above mentioned security issues, this document introduces the use of public/private key pair mechanism into DHCPv6, also with timestamp. The authority of the sender may depend on either pre-configuration mechanism or PKI. By combining with the signatures, sender identity can be verified and messages protected.

This document introduces a Secure DHCPv6 mechanism that uses a public /private key pair to secure the DHCPv6 protocol. It has two modes; in both modes, the sender has a public/private key pair. In the first mode, the public key of the sender is pre-shared with the recipient, either opportunistically or through a manual process. In the second mode, the sender has a certificate for its public key, signed by a Certificate Authority that is trusted by the recipient. It is possible for the same public key to be used with different recipients in both modes.

In this document, we introduce a public key option, a certificate option and a signature options with a corresponding verification mechanism. Timestamp is integrated into signature options. A DHCPv6 message (from a server or a client), with either a public key or certificate option, and carrying a digital signature, can be verified by the recipient for both the timestamp and authentication, then process the payload of the DHCPv6 message only if the validation is successful. Because the sender can be a DHCPv6 server or a client, the end-to-end security protection can be from DHCPv6 servers to or clients, or from clients to DHCPv6 servers.

This improves communication security of DHCPv6 messages. The authentication options [RFC3315] may also be used for replay protection.

4.1. New Components

The components of the solution specified in this document are as follows:

- The node generates a public/private key pair. A DHCPv6 option is defined that carries the public key.

  The node may also obtain a certificate from a Certificate Authority that can be used to establish the trustworthiness of the node. A second option is defined to carry the certificate. Because the certificate contains the public key, there is never a need to send both options at the same time.

- A signature generated using the private key that protects the integrity of the DHCPv6 messages and authenticates the identity of the sender.

- A timestamp, to detect and prevent packet replay. The secure DHCPv6 nodes need to meet some accuracy requirements and be synced to global time, while the timestamp checking mechanism allows a configurable time value for clock drift.

4.2. Support for algorithm agility

Hash functions are used to provide message integrity checks. In order to provide a means of addressing problems that may emerge in the future with existing hash algorithms, as recommended in [RFC4270], this document provides a mechanism for negotiating the use of more secure hashes in the future.

In addition to hash algorithm agility, this document also provides a mechanism for signature algorithm agility.

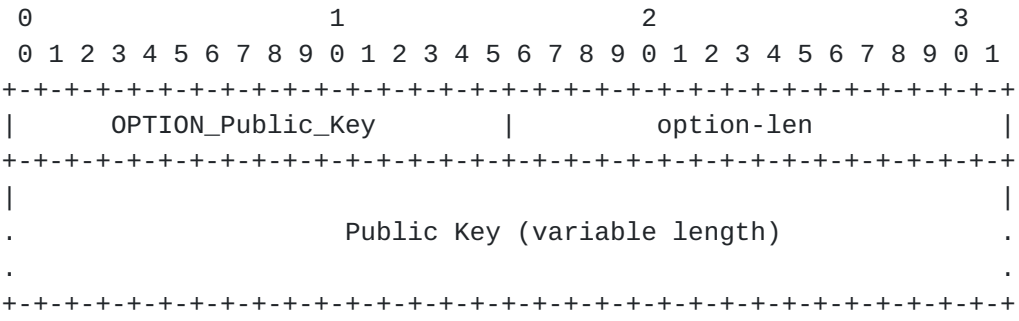
The support for algorithm agility in this document is mainly a unilateral notification mechanism from sender to recipient. If the recipient does not support the algorithm used by the sender, it cannot authenticate the message. In this case, the receiver SHOULD reply with a NotSupportAlgorithm status code (defined in Section 5.4). Upon receiving this status code, the sender MAY resend the message protected with the mandatory algorithms (defined in Section 5.3). Therefore, the senders in a same administrative domain may be allowed to use various algorithms simultaneously.

5. Extensions for Secure DHCPv6

This section extends DHCPv6. Three new options have been defined. The new options MUST be supported in the Secure DHCPv6 message exchange.

5.1. Public Key Option

The Public Key option carries the public key of the sender. The format of the Public Key option is described as follows:



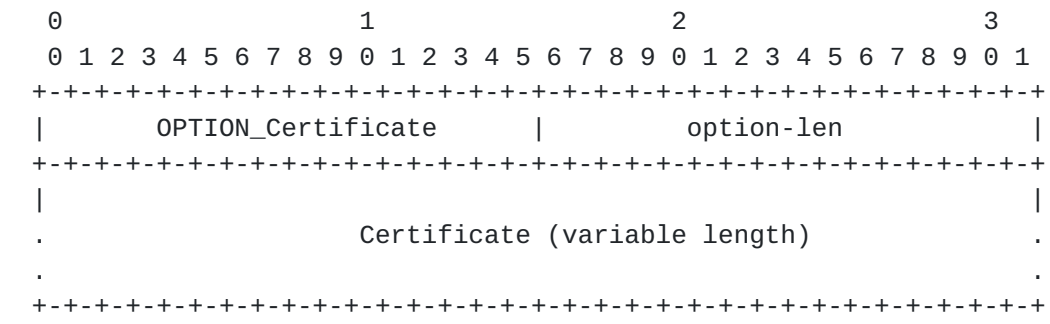
option-code OPTION_PK_PARAMETER (TBA1).

option-len Length of public key in octets.

Public Key A variable-length field containing public key. The key MUST be represented as a lower-case hexadecimal string with the most significant octet of the key first.

5.2. Certificate Option

The Certificate option carries the certificate of the sender. The format of the Certificate option is described as follows:



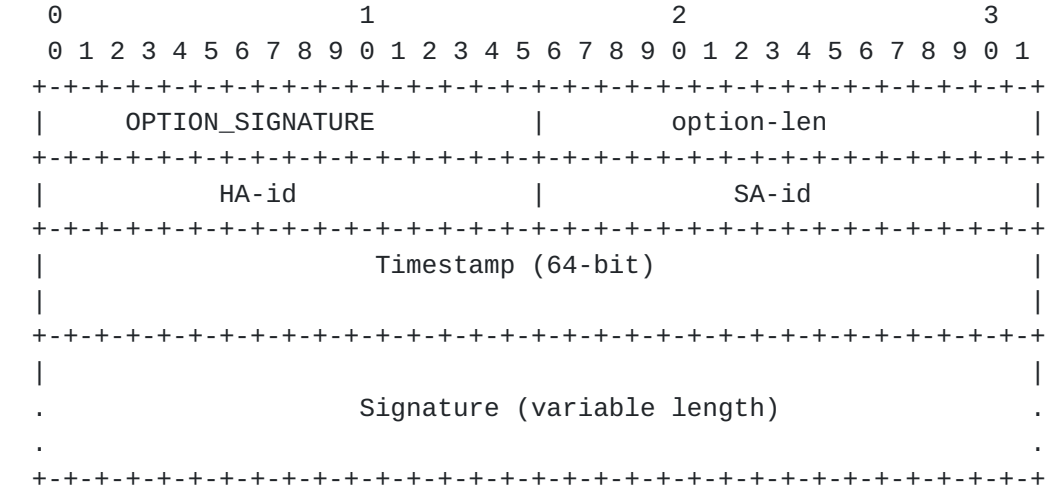
option-code OPTION_CERT_PARAMETER (TBA2).

option-len Length of certificate in octets.

Certificate A variable-length field containing certificate. The encoding of certificate and certificate data MUST be in format as defined in Section 3.6, [RFC5996]. The support of X.509 certificate is mandatory.

5.3. Signature Option

The Signature option allows public key-based signatures to be attached to a DHCPv6 message. The Signature option could be any place within the DHCPv6 message. It protects the entire DHCPv6 header and options, except for the Authentication Option. The format of the Signature option is described as follows:



option-code OPTION_SIGNATURE (TBA3).

option-len 12 + Length of Signature field in octets.

HA-id Hash Algorithm id. The hash algorithm is used for computing the signature result. This design is

adopted in order to provide hash algorithm agility. The value is from the Hash Algorithm for Secure DHCPv6 registry in IANA. The support of SHA-256 is mandatory. A registry of the initial assigned values is defined in Section 8.

SA-id — Signature Algorithm id. The signature algorithm is used for computing the signature result. This design is adopted in order to provide signature algorithm agility. The value is from the Signature Algorithm for Secure DHCPv6 registry in IANA. The support of RSASSA-PKCS1-v1_5 is mandatory. A registry of the initial assigned values is defined in Section 8.

Timestamp — The current time of day (NTP-format timestamp [RFC5905] in UTC (Coordinated Universal Time), a 64-bit unsigned fixed-point number, in seconds relative to 0h on 1 January 1900.). It can reduce the danger of replay attacks.

Signature — A variable-length field containing a digital signature. The signature value is computed with the hash algorithm and the signature algorithm, as described in HA-id and SA-id. The signature constructed by using the sender's private key protects the following sequence of octets:

1. The DHCPv6 message header.

2. All DHCPv6 options including the Signature option (fill the signature field with zeroes) except for the Authentication Option.

The signature filed MUST be padded, with all 0, to the next octet boundary if its size is not an even multiple of 8 bits. The padding length depends on the signature algorithm, which is indicated in the SA-id field.

Note: if both signature and authentication option are presented, signature option does not protect authentication option. It is because both needs to apply hash algorithm to whole message, so there must be a clear order and there could be only one last-created option. In order to avoid update RFC3315 because of changing auth option, the authors chose not include authentication option in the signature.

5.4. Status Codes

- o NotSupportAlgorithm (TBD4): Indicates the recipient does not support algorthims that sender used.

- o NotSupportFaithModel (TBD5): Indicates the recipient does not support the leap of faith model.

- o FaithListExceed (TBD6): Indicates the recipient's list that stores public keys or unverifiable certificates in the leap of faith model currently exceeds.

6. Processing Rules and Behaviors

6.1. Processing Rules of Sender

The sender of a Secure DHCPv6 message could be a DHCPv6 server or a DHCPv6 client.

The node must have a public/private key pair in order to create Secure DHCPv6 messages. The node may have a certificate which is signed by a CA trusted by both sender and recipient.

To support Secure DHCPv6, the Secure DHCPv6 enabled sender MUST construct the DHCPv6 message following the rules defined in [RFC3315].

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain either a the Public Key or Certificate option, which MUST contructed as explained in Section 5.1 or Section 5.2.

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain the Signature option, which MUST be constructed as explained in Section 5.3. It protects the message header and the message payload and all DHCPv6 options except for the Signature option itself and the Authentication Option. Within the Signature option the Timestamp field SHOULD be set to the current time, according to sender's real time clock.

A Relay-forward and relay-reply message MUST NOT contain any Public Key or Certificate option or Signature Option.

Upon receiving a Reply message with a NotSupportAlgorithm status code, the sender MAY resend the message protected with the mandatory algorithms.

Upon receiving a Reply message with a NotSupportFaithModel or FaithListExceed status code, the sender is not able to build up the

connection with the recipient. The sender MAY swith to a verifiable certificate. In the latter case, the sender MAY retry later.

6.2. Processing Rules of Recipient

When receiving a DHCPv6 message, except for Relay-Forward and Relay-Reply messages, a Secure DHCPv6 enabled recipient SHOULD discard the DHCPv6 message if the Signature option is absent, or both the Public Key and Certificate option is absent, or both the Public Key and Certificate option are presented. If all three options are absent, the recipient MAY fall back the unsecure DHCPv6 model.

The recipient SHOULD first check the support of algorthims that sender used. If not, an error NotSupportAlgorithm status code should be sent back to the sender, while the message is dropped siliently. If all algorthims are supported, the recipient then checks the authority of this sender.

If the sender uses certificate, the recipient SHOULD validate the sender's certificate following the rules defined in [RFC5280]. An implementation may create a local trust certificate record for a verified certificate in order to avoid repeated verfication procedure in the future. A sender certificate that finds a match in the local trust certificate list are treated as verified. A fast search index may be created for this list.

If the sender uses a public key, the recipient SHOULD validate it by finding a match public key from the local trust public key list, which is pre-configured or recorded from previous communications. A local trust public key list is a data table maintained by the recipient. It restores public keys from all trustworthy senders. A fast search index may be created for this list.

The recipient may choose to further process the message from a sender for which no authorization information exists, either non-matched public key or certificate cannot be verified. By recording the public key or unverifiable certificate that was used by the sender, when the first time it is seen, the recipient can make a leap of faith that the sender is trustworthy. If no evidence to the contrary surfaces, the recipient can then validate the sender as trustworthy when it subsequently sees the same public key or certificate used to sign messages from the same sender. If the recipient does not support the leap of faith model, it SHOULD reply a message with an error NotSupportFaithModel status code, defined in Section 5.4, back to the sender.

The number of cached public keys or unverifiable certificates MUST be limited in order to protect the DHCPv6 server against resource

exhaustion attacks. If the recipient's list that stores public keys or unverifiable certificates in the leap of faith model exceeds, an error FaithListExceed status code SHOULD be returned to the sender. The resource releasing policy against exceeding situations is out of scope.

At this point, the recipient has either recognized the authorization of the sender, or decided to attempt a leap of faith. The recipient MUST now authenticate the sender by verifying the Signature and checking timestamp. The order of two procedures is left as an implementation decision. It is RECOMMENDED to check timestamp first, because signature verification is much more computationally expensive.

The signature field verification MUST show that the signature has been calculated as specified in Section 5.3. Only the messages that get through both the signature verifications and timestamp check are accepted as secured DHCPv6 messages and continue to be handled for their contained DHCPv6 options as defined in [RFC3315]. Messages that do not pass the above tests MUST be discarded or treated as unsecure messages.

The recipient MAY record the verified public key or certificate for future authentications.

Furthermore, the node that supports the verification of the Secure DHCPv6 messages MAY record the following information:

Minbits The minimum acceptable key length for public keys. An upper limit MAY also be set for the amount of computation needed when verifying packets that use these security associations. The appropriate lengths SHOULD be set according to the signature algorithm and also following prudent cryptographic practice. For example, minimum length 1024 and upper limit 2048 may be used for RSA [RSA].

A Relay-forward or Relay-reply message with any Public Key, Certificate or the Signature option is invilad. The message SHOULD be discarded silently.

6.3. Processing Rules of Relay Agent

To support Secure DHCPv6, relay agents just need to follow the same processing rules defined in [RFC3315]. There is nothing more the relay agents have to do, either verify the messages from client or server, or add any secure DHCPv6 options. Actually, be definition in this document, relay agents MUST NOT add any secure DHCPv6 options.

6.4. Timestamp Check

Recipients SHOULD be configured with an allowed timestamp Delta value, a "fuzz factor" for comparisons, and an allowed clock drift parameter. The recommended default value for the allowed Delta is 300 seconds (5 minutes); for fuzz factor 1 second; and for clock drift, 0.01 second.

Note: the Timestamp mechanism is based on the assumption that communication peers have rough synchronized clocks, with certain allowed clock drift. So, accurate clock is not necessary. If one has a clock too far from the current time, the timestamp mechanism would not work.

To facilitate timestamp checking, each recipient SHOULD store the following information for each sender, from which at least one accepted secure DHCPv6 message is successfully verified (for both timestamp check and signature verification):

- The receive time of the last received and accepted DHCPv6 message. This is called RDlast.

- The time stamp in the last received and accepted DHCPv6 message. This is called TSlast.

An verified (for both timestamp check and signature verification) secure DHCPv6 message initiates the update of the above variables in the recipient's record.

Recipients MUST check the Timestamp field as follows:

- When a message is received from a new peer (i.e., one that is not stored in the cache), the received timestamp, TSnew, is checked, and the message is accepted if the timestamp is recent enough to the reception time of the packet, RDnew:

  -Delta < (RDnew - TSnew) < +Delta

  After the signature verification also successes, the RDnew and TSnew values SHOULD be stored in the cache as RDlast and TSlast.

- When a message is received from a known peer (i.e., one that already has an entry in the cache), the timestamp is checked against the previously received Secure DHCPv6 message:

  TSnew + fuzz > TSlast + (RDnew - RDlast) x (1 - drift) - fuzz

If this inequality does not hold, the recipient SHOULD silently discard the message. If, on the other hand, the inequality holds, the recipient SHOULD process the message.

Moreover, if the above inequality holds and TSnew > TSlast, the recipient SHOULD update RDlast and TSlast after the signature verification also successes. Otherwise, the recipient MUST NOT update RDlast or TSlast.

An implementation MAY use some mechanism such as a timestamp cache to strengthen resistance to replay attacks. When there is a very large number of nodes on the same link, or when a cache filling attack is in progress, it is possible that the cache holding the most recent timestamp per sender will become full. In this case, the node MUST remove some entries from the cache or refuse some new requested entries. The specific policy as to which entries are preferred over others is left as an implementation decision.

7. Security Considerations

This document provides new security features to the DHCPv6 protocol.

Using public key based security mechanism and its verification mechanism in DHCPv6 message exchanging provides the authentication and data integrity protection. Timestamp mechanism provides anti-replay function.

The Secure DHCPv6 mechanism is based on the pre-condition that the recipient knows the public key of senders or the sender's certificate can be verified through a trust CA. It prevents DHCPv6 server spoofing. The clients may decline the DHCPv6 messages from unknown/unverified servers, which may be fake servers; or may prefer DHCPv6 messages from known/verified servers over unsigned messages or messages from unknown/unverified servers. The pre-configuration operation also needs to be protected, which is out of scope. The deployment of PKI is also out of scope.

However, when a DHCPv6 client first encounters a new public key or a new unverifiable certificate, it can make a leap of faith. If the DHCPv6 server that used that public key or unverifiable certificate is in fact legitimate, then all future communication with that DHCPv6 server can be protected by storing the public key or unverifiable certificate. This does not provide complete security, but it limits the opportunity to mount an attack on a specific DHCPv6 client to the first time it communicates with a new DHCPv6 server. The number of cached public keys or unverifiable certificates MUST be limited in order to protect the DHCPv6 server against resource exhaustion attacks.

Downgrade attacks cannot be avoided if nodes are configured to accept both secured and unsecured messages. A future specification may provide a mechanism on how to treat unsecured DHCPv6 messages.

[RFC6273] has analyzed possible threats to the hash algorithms used in SEND. Since the Secure DHCPv6 defined in this document uses the same hash algorithms in similar way to SEND, analysis results could be applied as well: current attacks on hash functions do not constitute any practical threat to the digital signatures used in the signature algorithm in the Secure DHCPv6.

A window of vulnerability for replay attacks exists until the timestamp expires. Secure DHCPv6 nodes are protected against replay attacks as long as they cache the state created by the message containing the timestamp. The cached state allows the node to protect itself against replayed messages. However, once the node flushes the state for whatever reason, an attacker can re-create the state by replaying an old message while the timestamp is still valid. In addition, the effectiveness of timestamps is largely dependent upon the accuracy of synchronization between communicating nodes. However, the two communciating nodes can be synchronized is out of scope of this work.

Attacks against time synchronization protocols such as NTP [RFC5905] may cause Secure DHCPv6 nodes to have an incorrect timestamp value. This can be used to launch replay attacks, even outside the normal window of vulnerability. To protect against these attacks, it is recommended that Secure DHCPv6 nodes keep independently maintained clocks or apply suitable security measures for the time synchronization protocols.

8. IANA Considerations

This document defines three new DHCPv6 [RFC3315] options. The IANA is requested to assign values for these three options from the DHCP Option Codes table of the DHCPv6 Parameters registry maintained in http://www.iana.org/assignments/dhcpv6-parameters. The three options are:

The Public Key Option (TBA1), described in Section 5.1.

The Certificate Option (TBA2), described in Section 5.2.

The Signature Option (TBA3), described in Section 5.3.

The IANA is also requested to add two new registry tables to the DHCPv6 Parameters registry maintained in http://www.iana.org/assignments/dhcpv6-parameters. The two tables are the Hash Algorithm

for Secure DHCPv6 table and the Signature Algorithm for Secure DHCPv6 table.

Initial values for these registries are given below. Future assignments are to be made through Standards Action [RFC5226]. Assignments for each registry consist of a name, a value and a RFC number where the registry is defined.

Hash Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Hash Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
Reserved	0x0000	this document
SHA-1	0x0001	this document
SHA-256	0x0002	this document
SHA-512	0x0003	this document

Signature Algorithm for Secure DHCPv6. The values in this table are 16-bit unsigned integers. The following initial values are assigned for Signature Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
Reserved	0x0000	this document
RSASSA-PKCS1-v1_5	0x0001	this document

IANA is requested to assign the following new DHCPv6 Status Codes, defined in Section 5.4, in the DHCPv6 Parameters registry maintained in http://www.iana.org/assignments/dhcpv6-parameters:

Code	Name	Reference
TBD4	NotSupportAlgorithm	this document
TBD5	NotSupportFaithModel	this document
TBD6	FaithListExceed	this document

9. Acknowledgements

The authors would like to thank Bernie Volz, Ted Lemon, Ralph Droms, Jari Arkko, Sean Turner, Stephen Kent, Thomas Huth, David Schumacher, Francis Dupont, Tomek Mrugalski, Gang Chen and other members of the IETF DHC working groups for their valuable comments.

This document was produced using the xml2rfc tool [RFC2629].

10. Change log [RFC Editor: Please remove]

draft-ietf-dhc-sedhcpv6-00: adopted by DHC WG. 2013-11-19.

draft-jiang-dhc-sedhcpv6-02: removed protection between relay agent and server due to complexity, following the comments from Ted Lemon, Bernie Volz. 2013-10-16.

draft-jiang-dhc-sedhcpv6-01: update according to review comments from Ted Lemon, Bernie Volz, Ralph Droms. Separated Public Key/ Certificate option into two options. Refined many detailed processes. 2013-10-08.

draft-jiang-dhc-sedhcpv6-00: original version, this draft is a replacement of draft-ietf-dhc-secure-dhcpv6, which reached IESG and dead because of consideration regarding to CGA. The authors followed the suggestion from IESG making a general public key based mechanism. 2013-06-29.

11. References

11.1. Normative References

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

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

11.2. Informative References

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC4270] Hoffman, P. and B. Schneier, "Attacks on Cryptographic Hashes in Internet Protocols", RFC 4270, November 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC6273] Kukec, A., Krishnan, S., and S. Jiang, "The Secure Neighbor Discovery (SEND) Hash Threat Analysis", RFC 6273, June 2011.

[RSA] RSA Laboratories, "RSA Encryption Standard, Version 2.1, PKCS 1", November 2002.

Authors' Addresses

Sheng Jiang (editor)
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com

Sean Shen
CNNIC
4, South 4th Street, Zhongguancun
Beijing 100190
P.R. China

Email: shenshuo@cnnic.cn

Dacheng Zhang
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: zhangdacheng@huawei.com