DHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 2, 2015

S. Jiang, Ed.
Huawei Technologies Co., Ltd
S. Shen
CNNIC
D. Zhang
Huawei Technologies Co., Ltd
T. Jinmei
Infoblox Inc.

September 29, 2014

Secure DHCPv6
draft-ietf-dhc-sedhcpv6-04

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) enables DHCPv6 servers to pass configuration parameters. It offers configuration flexibility. If not being 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 communications between DHCPv6 clients and DHCPv6 servers. This document provides a DHCPv6 client/server authentication mechanism based on server's public/private key pairs and client's certificates. The DHCPv6 message exchanges are protected by the signature option and the timestamp option newly defined in this document.

Table of Contents

1. Introduction

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6, [RFC3315]) enables DHCPv6 servers to pass configuration parameters and offers configuration flexibility. If not being 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.
- anti-replay protection based on timestamps.

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 communications 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 server's public/private key pairs and client's certificates. The reason for such design and deployment consideration are discussed in Section 7. It also integrates message signatures for the integrity and timestamps for anti-replay. The client authentication on server procedure defined in this document depends on deployed Public Key Infrastructure (PKI, [RFC5280]). However, the deployment of PKI 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 a DHCPv6 server to automatically configure relevant network parameters on clients. In the basic DHCPv6 specification [RFC3315], security of DHCPv6 messages 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 security mechanism defined in this document allows the public key database on the client 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. Overview of Secure DHCPv6 Mechanism with Public Key

This document introduces a Secure DHCPv6 mechanism that uses signatures to secure the DHCPv6 protocol. In order to enable DHCPv6 clients and DHCPv6 servers to perform mutual authentication without previous key deployment, this solution provides a DHCPv6 client/server authentication mechanism based on server's public/private key pairs and client's certificates: the server only accept the client messages that are protected by the client certificate that is signed

by a trusted CA; a client can build up trust relationship with a server for subsequent message exchanges based on leap of faith (LoF) mechanism. This purpose of this design is to simplify the precondition of deploying DHCPv6 authentication and provides limited protection of DHCPv6 message.

In this document, we introduce a public key option (only sent by servers), a certificate option (only sent by clients), a signature option and a timestamp with corresponding verification mechanisms. A DHCPv6 message from a server is attached with a public key option, and carrying a digital signature and a timestamp option. It can be verified by the client. The client processes the payload of the DHCPv6 message only if the validation is successful. Reversely, a DHCPv6 message from a client is attached with a certificate option, and also carrying a digital signature and a timestamp option. It can be verified by the server. The server processes the payload of the DHCPv6 message only if the validation is successful. The end-to-end security protection is bidirection that covers both from DHCPv6 servers to clients and from clients to DHCPv6 servers. Additionally, the optional timestamp mechanism provides anti-replay protection.

By recording the public key that was used by the DHCPv6 server, when the first time it is seen, the DHCPv6 client can make a leap of faith that the server is trustworthy. If no evidence to the contrary surfaces, the client can then validate the server as trustworthy when it subsequently sees the same public key used to sign messages from the same server. In opposite, once the client has determined that it is being attacked, it can either forget that server, or remember that server in a blacklist and drop further packets associated with that server.

On the server DHCPv6 side, upon receiving the client's certificate, the server asserts the validity of the certificate, for example through PKI.

Secure DHCPv6 messages are commonly large. IPv6 fragments [RFC2460] are highly possible. Hence, deployment of Secure DHCPv6 should also consider the issues of IP fragment, PMTU, etc. Also, if there are firewalls between secure DHCPv6 clients and secure DHCPv6 servers, it is RECOMMENDED that the firewalls are configured to pass ICMP Packet Too Big messages [RFC4443].

4.1. New Components

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

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

- The client obtains a certificate from a Certificate Authority that can be used to establish the trustworthiness with the server. Another option is defined to carry the certificate.

- A signature generated using the private key which is used by the receiver to verify the integrity of the DHCPv6 messages and then the identity of the sender.

- A timestamp, to detect replayed packet. 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. The real time provision is out of scope.

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. A recipient MAY support various algorithms simultaneously, and the different senders in a same administrative domain may be allowed to use various algorithms simultaneously.

If the recipient does not support the algorithm used by the sender, it cannot authenticate the message. In the client-to-server case, the server SHOULD reply with an AlgorithmNotSupported status code (defined in Section 5.5). Upon receiving this status code, the client MAY resend the message protected with the mandatory algorithm (defined in Section 5.3).

4.3. Applicability

By default, a secure DHCPv6 enabled client SHOULD start with secure mode by sending secure DHCPv6 messages. If the recipient is secure DHCPv6 enabled server, their communication would be in secure mode. In the scenario where the secure DHCPv6 enabled client and server fail to build up secure communication between them, the secure DHCPv6

enabled client MAY choose to send unsecured DHCPv6 message towards the server according to its local policies.

In the scenario where the recipient is a legacy DHCPv6 server that does not support secure mechanism, the DHCPv6 server (for all of known DHCPv6 implementations) would just omit or disregard unknown options (secure options defined in this document) and still process the known options. The reply message would be unsecured, of course. It is up to the local policy of the client whether to accept the messages. If the client accepts the unsecured messages from the DHCPv6 server, the subsequent exchanges will be in the unsecured mode.

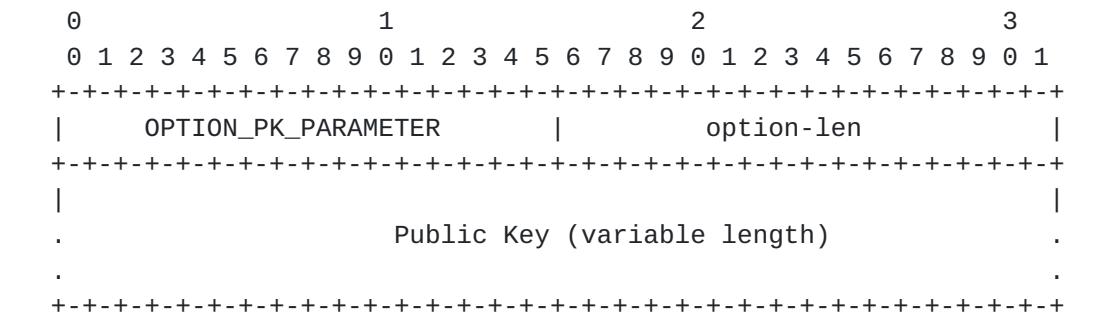
In the scenario where a legacy client sends an unsecured message to a secure DHCPv6 enabled server, there are two possibilities depending on the server policy. If the server's policy requires the authentication, an UnspecFail (value 1, [RFC3315]) error status code, SHOULD be returned. In such case, the client cannot build up the connection with the server. If the server has been configured to support unsecured clients, the server MAY fall back to the unsecured DHCPv6 mode, and reply unsecured messages toward the client; depending on the local policy, the server MAY continue to send the secured reply messages with the consumption of computing resource. The resources allocated for unsecured clients SHOULD be separated and restricted.

5. Extensions for Secure DHCPv6

This section describes the extensions to DHCPv6. Four 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 server. The format of the Public Key option is described as follows:



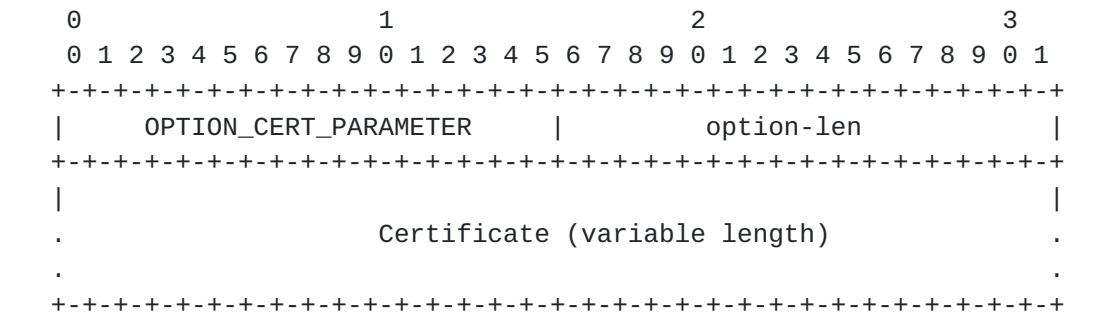
option-code OPTION_PK_PARAMETER (TBA1).

option-len Length of public key in octets.

Public Key A variable-length field containing public key and identify the algorithm with which the key is used (e.g., RSA, DSA, or Diffie-Hellman). The algorithm is identified using the AlgorithmIdentifier structure specified in section 4.1.1.2, [RFC5280]. The object identifiers for the supported algorithms and the methods for encoding the public key materials (public key and parameters) are specified in [RFC3279], [RFC4055], and [RFC4491].

5.2. Certificate Option

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



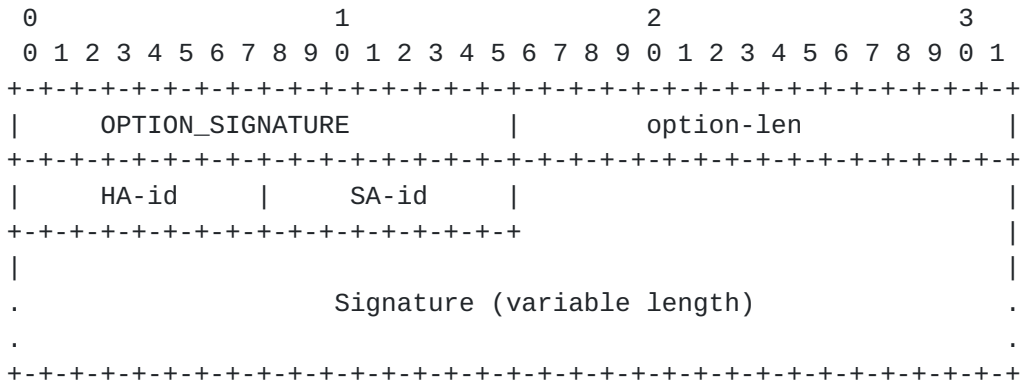
option-code OPTION_CERT_PARAMETER (TBA2).

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 a signature that is signed by the private key to be attached to a DHCPv6 message. The Signature option could be any place within the DHCPv6 message while it is logically created after the entire DHCPv6 header and options, except for the Authentication Option. It protects the entire DHCPv6 header and options, including itself, except for the Authentication Option. The format of the Signature option is described as follows:



option-code OPTION_SIGNATURE (TBA3).

option-len 2 + 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.

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.

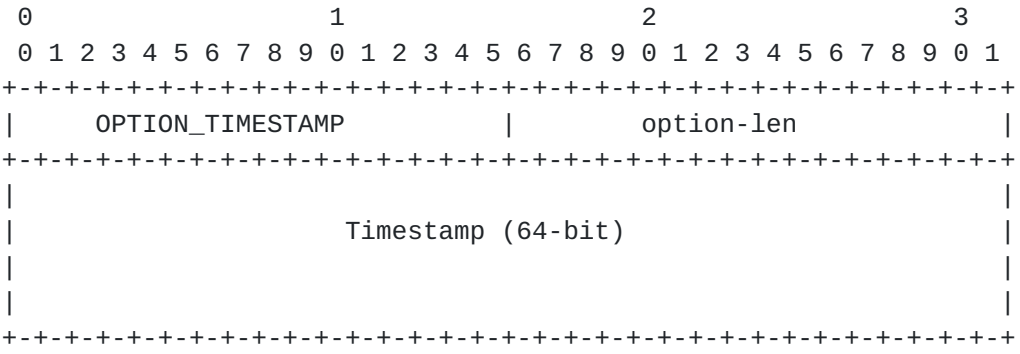
2. All DHCPv6 options including the Signature option (fill the signature field with zeroes) except for the Authentication Option.

The signature field MUST be padded, with all 0, to the next octet boundary if its size is not a 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 the Authentication Option. It allows the Authentication Option be created after signature has been calculated and filled with the valid signature. It is because both options need 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. Timestamp Option

The Timestamp option carries the current time on the sender. It adds the anti-replay protection to the DHCPv6 messages. It is optional.



option-code OPTION_TIMESTAMP (TBA4).

option-len 8, in octets.

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.

5.5. Status Codes

- AlgorithmNotSupported (TBD5): indicates that the DHCPv6 server does not support algorithms that sender used.
- AuthenticationFail (TBD6): indicates that the DHCPv6 client fails authentication check.
- TimestampFail (TBD7): indicates the message from DHCPv6 client fails the timestamp check.
- SignatureFail (TBD8): indicates the message from DHCPv6 client fails the signature check.

6. Processing Rules and Behaviors

This section only covers the scenario where both DHCPv6 client and DHCPv6 server are secure enabled.

6.1. Processing Rules of Sender

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

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

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 Public Key or a Certificate option, which MUST be constructed as explained in Section 5.1 or Section 5.2.

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MUST contain one and only one Signature option, which MUST be constructed as explained in Section 5.3. It protects the message header and all DHCPv6 options except for the Authentication Option.

A Secure DHCPv6 message, except for Relay-forward and Relay-reply messages, MAY contain one and only one Timestamp 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 additional Public Key or Certificate option or Signature Option or

Timestamp Option, aside from those present in the innermost encapsulated messages from the client or server.

If the sender is a DHCPv6 client, in the failure cases, it receives a Reply message with an error status code. The error status code indicates the failure reason on the server side. According to the received status code, the client MAY take follow-up action:

- Upon receiving an AlgorithmNotSupported error status code, the client SHOULD resend the message protected with the mandatory algorithms.

- Upon receiving an AuthenticationFail error status code, the client is not able to build up the secure communication with the recipient. The client MAY switch to other certificate if it has. But it SHOULD NOT retry with the same certificate. It MAY retry with the same certificate following normal retransmission routines defined in [RFC3315].

- Upon receiving a TimestampFail error status code, the client MAY fall back to unsecured mode, or resend the message without a Timestamp option. However, the DHCP server MAY not accept the message without a Timestamp option.

- Upon receiving a SignatureFail error status code, the client MAY resend the message following normal retransmission routines defined in [RFC3315].

6.2. Processing Rules of Recipient

The recipient of a Secure DHCPv6 message could be a DHCPv6 server or a DHCPv6 client. In the failure cases, either DHCPv6 server or client SHOULD NOT process received message, and the server SHOULD reply a correspondent error status code, while the client does nothing. The specific behavior depends on the configured local policy.

When receiving a DHCPv6 message, except for Relay-Forward and Relay-Reply messages, a secure DHCPv6 enabled recipient SHOULD discard any DHCPv6 messages that meet any of the following conditions:

- the Signature option is absent,

- multiple Signature option is presented,

- the Public Key option is absent in the server-to-client message,

- the Certificate option is presented in the server-to-client message,
- the Certificate option is absent in the client-to-server message,
- the Public Key option is presented in the client-to-server message.

In such failure, if the recipient is a DHCPv6 server, the server SHOULD reply an UnspecFail (value 1, [RFC3315]) error status code. If neither of the Signature, Public Key or Certificate options is presented, the sender MAY be a legacy node or in unsecured mode, then, the recipient MAY fall back to the unsecured DHCPv6 mode if its local policy allows.

The recipient SHOULD first check the support of algorithms that sender used. If not pass, the message is dropped. In such failure, if the recipient is a DHCPv6 server, the server SHOULD reply an AlgorithmNotSupported error status code, defined in Section 5.5, back to the client. If all algorithms are supported, the recipient then checks the authority of this sender.

The DHCPv6 server SHOULD validate the client's certificate following the rules defined in [RFC5280]. An implementation may create a local trust certificate record for verified certificates in order to avoid repeated verification procedure in the future. A client certificate that finds a match in the local trust certificate list is treated as verified. A fast search index may be created for this list.

The DHCPv6 client SHOULD validate it by finding a matching 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 message that fails authentication check MUST be dropped. In such failure, the DHCPv6 server SHOULD reply an AuthenticationFail error status code, defined in Section 5.5, back to the client.

The client MAY choose to further process messages from a server for which there is no matched public key. By recording the public key, when the first time it is seen, the client can make a leap of faith (LoF) that the server is trustworthy. If no evidence to the contrary surfaces, the client can then validate the server as trustworthy for subsequent message exchanges. In opposite, once the client has determined that it is being attacked, it can either forget that public key, or remember that public key in a blacklist and drop further packets associated with that public key.

At this point, the recipient has either recognized the authentication of the sender, or decided to drop the message. The recipient MUST now authenticate the sender by verifying the signature and checking timestamp (see details in Section 6.4), if there is a Timestamp option. 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. Depending on server's local policy, the message without a Timestamp option MAY be acceptable or rejected. If the server rejects such a message, a TimestampFail error status code, defined in Section 5.5, should be sent back to the client.

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 (if there is a Timestamp option) 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 unsecured messages. In the case the recipient is DHCPv6 server, the DHCPv6 server SHOULD reply a SignatureFail error status code, defined in Section 5.5, for the signature verification failure; or a TimestampFail error status code, defined in Section 5.5, for the timestamp check failure, back to the client.

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 invalid. The message MUST 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, by definition in this document, relay agents SHOULD NOT add any secure DHCPv6 options.

6.4. Timestamp Check

In order to check the Timestamp option, defined in Section 5.4, 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 roughly 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 timestamp in the last received and accepted DHCPv6 message. This is called TSlast.

A 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 or RDnew < RDlast, 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. Deployment Consideration

This document defines two directions of authentication: authentication based on client's certificate and authentication based on leap-of-faith (LoF) to server's public key.

7.1. Authentication on a client

For clients, DHCPv6 authentication generally means verifying whether the sender of DHCP messages is a legal DHCPv6 server and verifying whether the message has been modified during transmission. Because the client may have to validate the authentication in the condition of without connectivity wider than link-local, authentication with certificates may not always be feasible. So, this document only sticks on Leaf of Faith model, to make sure the client talks to the same previous server.

Message integrity is provided. But there is a chance for the client to incorrectly trust a malicious server at the beginning of the first session with the server (and therefore keep trusting it thereafter). But LoF guarantees the subsequent messages are sent by the same previous server, and therefore narrows the attack scope. This may make sense if the network can be reasonably considered secure and requesting pre-configuration is deemed to be infeasible. A small home network would be an example of such cases.

For environments that are neither controlled nor really trustworthy, such as a network cafe, while LoF model, i.e. silently trusting the server at the first time, would be too insecure. But some middle

ground might be justified, such as requiring human intervention at the point of LoF.

7.2. Authentication on a server

As for authentication on a server, there are several different scenarios to consider, each of which has different applicability issues. If the server allows LoF any malicious user can pretend to be a new legitimate client. While the server can always be considered to have connectivity to validate certificate, it is feasible to check client certificates.

Network administrators may wish to constrain the allocation of addresses to authorized hosts to avoid denial of service attacks in "hostile" environments where the network medium is not physically secured, such as wireless networks or college residence halls. A server may have to selectively serve a specific client or deny specific clients depending on the identity of the client in a controlled environment, like a corporate intranet. But the support from skilled staff or administrator may be required to set up the clients.

8. 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 client knows the public key of servers or the client's certificate can be verified through a trust CA. It prevents DHCPv6 server spoofing. The clients may discard 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, it can make a leap of faith. If the DHCPv6 server that used that public key is in fact legitimate, then all future communication with that DHCPv6 server can be protected by storing the public key. 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.

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 server, whose local policy accepts messages without a Timestamp option, may have to face the risk of replay attacks.

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, how the two communicating 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.

9. IANA Considerations

This document defines three new DHCPv6 [RFC3315] options. The IANA is requested to assign values for these three options from the DHCPv6 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 Timestamp Option (TBA4),described in Section 5.4.

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

Name	Value	RFCs
SHA-1	0x01	this document
SHA-256	0x02	this document
SHA-512	0x03	this document

Signature Algorithm for Secure DHCPv6. The values in this table are 8-bit unsigned integers. The following initial values are assigned for Signature Algorithm for Secure DHCPv6 in this document:

Name	Value	RFCs
RSASSA-PKCS1-v1_5	0x01	this document

IANA is requested to assign the following new DHCPv6 Status Codes, defined in Section 5.5, in the DHCPv6 Parameters registry maintained in http://www.iana.org/assignments/dhcpv6-parameters:

Code	Name	Reference
TBD5	AlgorithmNotSupported	this document
TBD6	AuthenticationFail	this document
TBD7	TimestampFail	this document
TBD8	SignatureFail	this document

10. Acknowledgments

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, Qi Sun, Suresh Krishnan,

Fred Templin and other members of the IETF DHC working groups for their valuable comments.

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

11. Change log [RFC Editor: Please remove]

draft-ietf-dhc-sedhcpv6-04: addressed comments from mail list. Making timestamp an independent and optional option. Reduce the serverside authentication to base on only client's certificate. Reduce the clientside authentication to only Leaf of Faith base on server's public key. 2014-09-26.

draft-ietf-dhc-sedhcpv6-03: addressed comments from WGLC. Added a new section "Deployment Consideration". Corrected the Public Key Field in the Public Key Option. Added consideration for large DHCPv6 message transmission. Added TimestampFail error code. Refined the retransmission rules on clients. 2014-06-18.

draft-ietf-dhc-sedhcpv6-02: addressed comments (applicability statement, redesign the error codes and their logic) from IETF89 DHC WG meeting and volunteer reviewers. 2014-04-14.

draft-ietf-dhc-sedhcpv6-01: addressed comments from IETF88 DHC WG meeting. Moved Dacheng Zhang from acknowledgement to be co-author. 2014-02-14.

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.

12. References

12.1. Normative References

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[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.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

[RFC4491] Leontiev, S. and D. Shefanovski, "Using the GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms with the Internet X.509 Public Key Infrastructure Certificate and CRL Profile", RFC 4491, May 2006.

[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.

12.2. 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

Tatuya Jinmei
Infoblox Inc.
3111 Coronado Drive
Santa Clara, CA
USA

Email: jinmei@wide.ad.jp