Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 20, 2014

S. Jiang
Huawei Technologies Co., Ltd
B. Carpenter
Univ. of Auckland
B. Liu
Y. Yin
Huawei Technologies Co., Ltd
October 17, 2013

Configuration Negotiation Protocol for Network Devices
draft-jiang-config-negotiation-protocol-00

Abstract

This document defines a new protocol that enables intelligent devices to dynamically negotiate their configuration with counterpart devices. This document only defines a general protocol as a negotiation platform while the negotiation objectives for specific scenarios are out of scope.

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 April 20, 2014.

Table of Contents

1. Introduction

The success of Internet has made the IP-based networks bigger and more complicated. The large-scaled ISP networks have become more and more for human based management. Also the operation cost is growing quickly. Consequently, there are therefore increased requirements for autonomy in the networks. In order to fulfil autonomy, devices that are more intelligent need to be able to negotiate directly with each other. [I-D.jiang-config-negotiation-ps] describes the requirements and application scenarios for network devices negotiation. It also describes a behavior model of a generic negotiation protocol. The design of Configuration Negotiation Protocol (CNP) in this document is mainly based on this behavior model.

Although many negotiations may happen between distributed horizontal peers, the main target scenarios are still hierarchy networks, which is the major structure of current large-scaled ISP networks. Thus, where necessary, we assume that each network element has a hierarchical superior. Of course, the protocol itself is capable of being used in a small and/or flat network structure, too.

This document defines a generic negotiation protocol, named Configuration Negotiation Protocol (CNP), that can be used to control decision process among distributed devices or between networks. The newly defined CNP in this document adapts a tight certificate-based mechanism, which needs the network operator runs a Public Key Infrastracture (PKI, [RFC5280]) system. The document introduce a new discovery mechanism, which is based on neighbor learning process and negotiation objective oriented.

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.

- Negotiation Objective: specific negotiation content, which needs to be decided in coordination with another network device. It is naturally based on a specific service or function or action.

- Negotiation Counterpart: a peer device with which the requesting device negotiates a specific negotiation objective.

- Device Identifier: a public key, which identifies the device in CNP messages. It is assumed that its associated private key is maintained in the device only.

- Device Certificate: A certificate for a single device, also the identitier of the device, further described in Section 5.

- Device Certificate Tag: a tag, which is bound to the device identitier. It is used to present Device Certificate in short form.

3. CNP Protocol Overview

The Configuration Negotiation protocol is designed to be a generic platform, which is independent from the negotiation contents. It only takes care of the general intercommunication between negotiation counterparts. The negotiation contents vary, giving the various negotiation objectives and the different pairs of negotiating counterparts.

The CNP has been designed based on simple initiator/responder model, while multiple-party negotiations could be completed by indirect

steps.

3.1. IP Version Independent

To be a generic platform, CNP should be IP version independent. In other words, it should be able to run over IPv6 and IPv4. Its messages and general options are neutral with respect to the IP version.

However, some functions, such as broadcasting on a link, may have to be IP version dependent. For these parts, the document defines support for both IP versions separately.

3.2. Objective Oriented Discovery Mechanism

Typically, one network device has multiple functions. It may be involved in different negotiation processes according to different negotiation objectives. Therefore, the traditional topology-oriented device discovery mechanisms are not sufficient for the CNP. A new

discovery mechanims is needed to find negotiation counterparts based on a specific negotiation objective. At a result, an objective-based discovery mechanism has been designed, in this document.

For every new negotiation objective, the negotiation requesting device needs to start a new discovery process in order to find the proper negotiation counterpart. Because a listening CNP-enabled device has to know the requested negotiation objective to decide whether it is a proper negotiation counterpart and make a response, the discovery process needs to be tightly coupled with the request process. Therefore, in this document, the discovery process is merged into the request process. There is no need for an independent discovery message and process.

3.3. Neighbor Diverting Discovery Mechanism

We now discuss the general flow of Request, Negotiation, and Negotiation-Ending messages, and Accept, Decline and Divert options. Details are given later.

Every Request message is sent to the ALL_CNP_NEIGHBOR mutlicast address Section 4.

If the neighbor device is a proper negotiation counterpart, it MAY respond with a Negotiation message to start a negotiation process, or with a Negotiation-Ending message in the case of a clear Accept or Decline.

If the neigbor device is not a proper negotiation counterpart for the objective given in the Request message, but knows a proper negotiation counterpart, for example because it negotiated the same objective with that negotiation counterpart before, it SHOULD respond with a Negotiation-Ending message with a Divert option pointed to the proper negotiation counterpart. If the neigbor device is not a proper negotiation counterpart, but does not know a proper negotiation counterpart, it SHOULD respond with a Negotiation-Ending message with a Divert option pointed to its hierachical upstream device.

After a CNP device successfully negotiated a specific objective with a negotiation counterpart, it SHOULD record this negotiation counterpart with this objective type locally. This record may be used for future negotiation or to pass to its neighbor as a Divert option. This learning mechanism should be able to support most network establishment scenarios.

3.4. Certificate-based Security Mechanism

A certification based security mechanism provides security properties for CNP:

- the identity of a CNP message sender can be verified by a recipient.
- the integrity of CNP message can be checked by the recipient of the message.
- anti-replay protection on the CNP message recipient.

The authority of the CNP message sender depends on a Public Key Infrastructure system, which should normally be run by the network operator.

A Request message MUST carry a Certificate option, defined in Section 8.6. The first Negotiation Message, responsing to a Request message, SHOULD also carry a Certificate option.

Every messages MUST carry a signature option, defined in Section 8.7.

For now, the authors do not think packet size is problem. In this CNP specification, there SHOULD NOT be multiple certificates in a single message. The current most used public keys are 1024/2048 bits, some may reach 4096. With overhead included, a single certificate is less than 500 bytes. Messages should be far shorter than the normal packet MTU.

3.4.1. 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], a mechanism for negotiating the use of more secure hashes in the future is provided.

In addition to hash algorithm agility, a mechanism for signature algorithm agility is also provided.

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. Senders in a single administrative domain are not required to upgrade to a new algorithm simultaneously.

So far, the algorithm agility is supported by one-way notification, rather than negotiation mode. As defined in Section 8.7, the sender notifies the recipient what hash/signature algorithms it used. If

the responder doesn't know a new algorithm used by the sender, the negotiation request would fail. In order to establish a negotiation session, the sender may fall back to an older, less preferred algorithm.

3.4.2. Message validation on reception

When receiving a CNP message, a recipient SHOULD discard the CNP

message if the Signature option is absent, or the Certificate option is in a Request Message.

For the Request message and the Response message with a Certification Option, the recipient SHOULD first check the authority of this sender following the rules defined in [RFC5280]. After successful authority validation, an implementation MUST add the sender's certification into the local trust certificate record indexed by the associated Device Certifice Tag, defined in Section 5.

The recipient MUST now authenticate the sender by verifying the Signature and checking timestamp, as specified in Section 3.4.3. 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 8.7. The public key used for signature validation is obtained from the certificate either carried by the message or found from a local trust certificate record by searching the message-carried Device Certicate Tag.

Only the messages that get through both the signature verifications and timestamp check are accepted and continue to be handled for their contained CNP options. Messages that do not pass the above tests MUST be discarded or treated as unsecure messages.

3.4.3. TimeStamp checking

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.

To facilitate timestamp checking, each recipient SHOULD store the following information for each sender:

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

- The time stamp in the last received and accepted CNP message.

This is called TSlast.

An accepted CNP message is any successfully verified (for both timestamp check and signature verification) CNP message from the given peer. It initiates the update of the above variables. Recipients MUST then 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

  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 CNP 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. 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.

3.5. Negotiation Procedures

A negotiation initiator sends a negotiation request to discovered negotiation counterpart devices, which may be different according to different negotiation objectives. It may request relevant information from the negotiation counterpart so that it can decide its local configuration to give the most coordinated performance. It

may request the he negotiation counterpart to make a matching configuration in order to set up a successful communication with it. It may request certain simulation/forecast result by sending some dry run conditions.

Beyond the traditional yes/no answer, the negotiation counterpart should be able to reply with suggestion of change condition in the negative scenario. This is going to start a bi-direction negotiation towards reaching a compromise between the two network devices.

The negotiation procedures is ended when one of the negotiation peer sends a Negotiation Ending message, which contains accept or decline option and do not need response from negotiation peer any more.

4. CNP Constants

- ALL_CNP_NEIGHBOR (TBD1)

  A link-local scope multicast address used by a CNP-enabled router to discover CNP-enabled neighbor (i.e., on-link) devices . All routers that support CNP are members of this multicast group.

  * IPv6 mutlicast address: TBD1

  * IPv4 multicast address: TBD2

- CNP Listen Port (TBD3)

  A UDP port that every CNP-enabled network device always listens to.

5. Device Identifier and Certificate Tag

A CNP-enabled Device should generated a public/private key pair. The device then uses the public key as its identifier, which is cryptographic in nature. is used to identify different devices. It is a CNP unique identifier for a CNP participant.

It then gets a certificate for this public key, signed by a Certificate Authority that is trusted by other network devices. The Certificate Authority SHOULD be managed by the network administrator, to avoid needing to trust a third party. The signed certificate would be used for authentication of the messasge sender.

A 128-bit Device Certifcate Tag, which is generated by taking a cryptographic hash over the device certificate, is a short

presentation for CNP messages. It is the index key to find the device certificate in a recepient's local trust certificate record.

The tag value is by taking SHA-1 hash algorithm over the correspondent device certificate and taking the leftmost 128 bits of the hash result.

6. Session Identifier

A 24-bit opaque value used to distinguish multiple sessions between the same two devices. A new session ID SHOULD be generated for every new Request message. All followup messages in the same negotiation procedure, which is initiated by the request message, SHOULD carry the same session identifier.

The session identifier SHOULD have low collision rate locally. It is RECOMMENDED to be generated by a pseudo random algorithm.

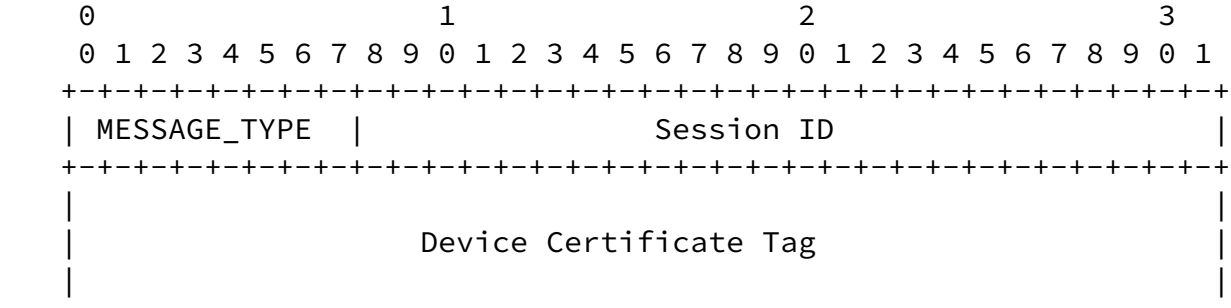
7. CNP Messages

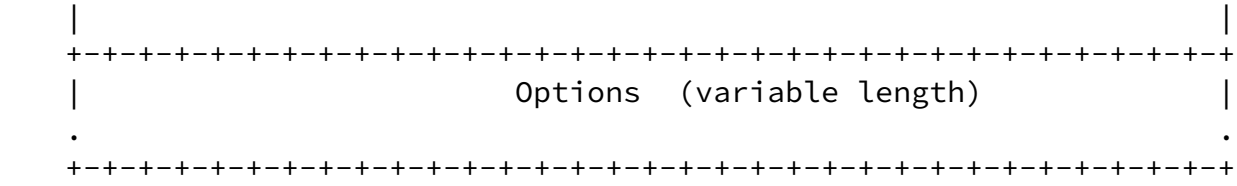
This document defines the following CNP message format and types. Message types not listed here are reserved for future use. The numeric encoding for each message type is shown in parentheses.

7.1. CNP Messsage Format

All CNP messages share an identical fixed format header and a vaiable format area for options. Every Message carreis the Device Certificate Tag of its sender and a session ID. Options are presented serially in the options field, with no padding between the options. Options are byte-aligned.

The following diagram illustrates the format of CNP messages:





MESSAGE_TYPE Identifies the CNP message type. 8-bit.

Session ID Identifies this negotiation session, as defined in

Section 6. 24-bit.

Device Certificate Tag
Present the Device Certificate, which identifies the negotiation deviceas, as defined in Section 5. The Device Certificate Tag is 128 bit, also defined in Section 5. It is used as index key to find the device certificate.

Options CNP Options carried in this message. Options are definded in Section 8.

7.2. Request Message

REQUEST (1) A negotiation requesting node sends a REQUEST message to initiate a negotiation.

If the requesting node does not know any negotiation counterpart, it sends the REQUEST messages to the link-local ALL_CNP_NEIGHBOR multicast address.

If the requesting node reopen to a known negotiation counterpart, it sends the REQUEST message to the unicast address of the negotiation counterparts directly.

7.3. Negotiation Message

NEGOTIATION (2)A negotiation counterpart sends an NEGOTIATION message in response to a REQUEST message or a

Negotiation message in a negotiation process which may need multiple steps.

7.4. Negotiation-ending Message

NEGOTIATION-ENDING (3)
A negotiation counterpart sends an NEGOTIATION-EDNING message to close the negotiation. It MUST contain one, but only one of accept/decline/divert option, defined in Section 8. It could be sent either by the requesting node or the responding node.

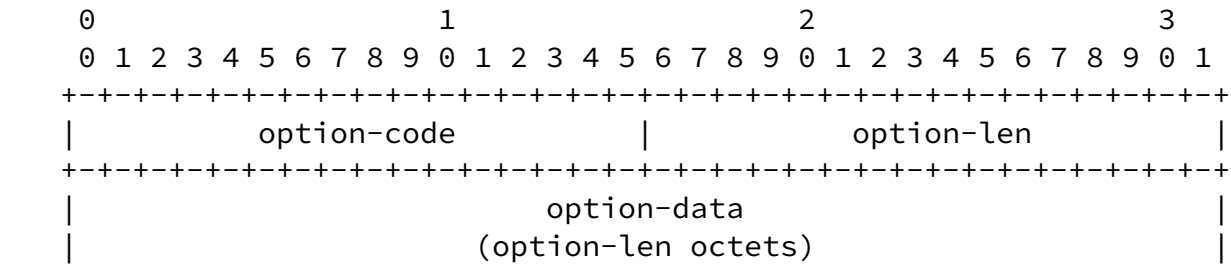
7.5. Confirm-waiting Message

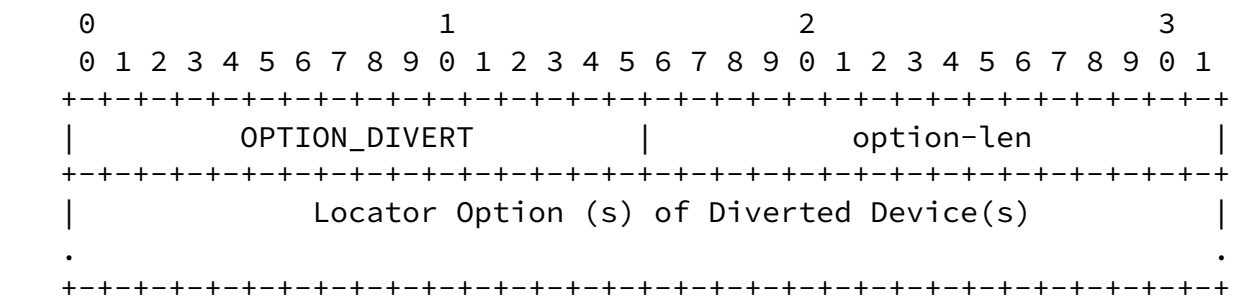
CONFIRM-WAITING (4)
A responding node sends a CONFIRM-WAITING message to indicate the requesting node to wait for the further negotiation response. It may because that the local process need more time or the negotiation is depending on another triggered negotiation. This message MUST NOT include any other options than the WAITING option defined in Section 8.5.

8. CNP General Options

This section defines the CNP general option for the negotiation protocol signalling. Option type 10~64 is reserved for CNP general options defined in the future.

8.1. Format of CNP Options





Option-code An unsigned integer identifying the specific option type carried in this option.

Option-len An unsigned integer giving the length of the option-data field in this option in octets.

Option-data The data for the option; the format of this data depends on the definition of the option.

CNP options are scoped by using encapsulation.

8.2. Divert Option

The divert option is used to redirect a CNP request to another node, which may be more proper for the intended negotiation. It may redirect to an entity that is known as a specific negotiation counterpart or a default gateway or a hierarchically upstream devices. The divert option MUST be only encapsulated in Negotiation-ending messages.

Option-code OPTION_DIVERT (1).

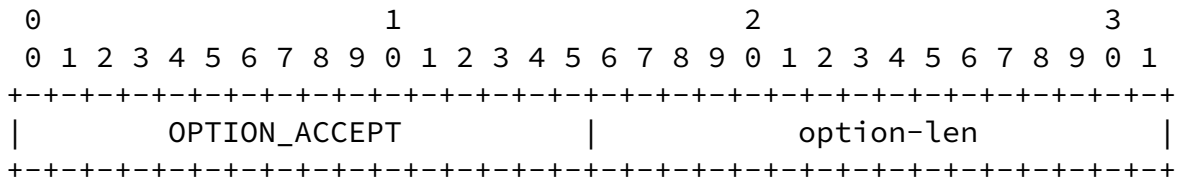
Option-len The length of diverted desitination sub option(s) in octets.

Locator Option (s) of Diverted Device
Locator Option(s),defined in Section 8.8,that point to diverted destination device(s).

8.3. Accept Option

The accept option is used to indicate the negotiation counterpart that the proposed negotiation content is accepted.

The accept option MUST be only encapsulated in Negotiation-ending messages.



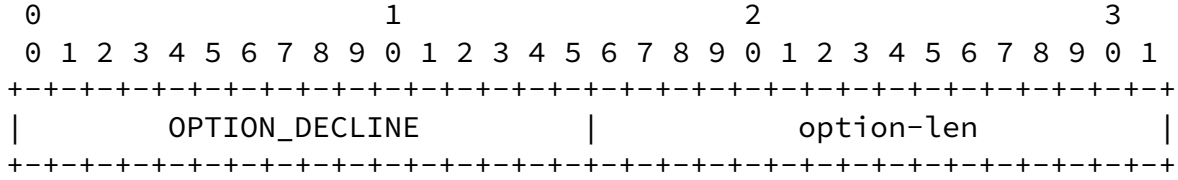
Option-code OPTION_ACCEPT (2).

Option-len 0.

8.4. Decline Option

The decline option is used to indicate the negotiation counterpart the proposed negotiation content is declined and end the negotiation process.

The decline option MUST be only encapsulated in Negotiation-ending messages.



Option-code OPTION_DECLINE (3).

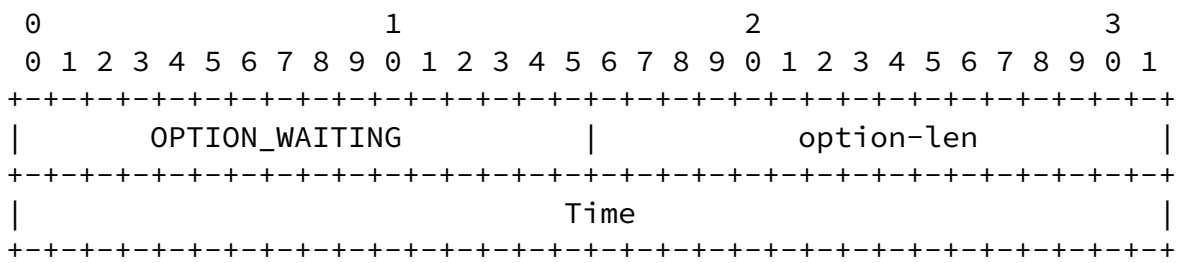
Option-len 0.

Notes: there are scenarios that a negotiation counterpart wants to decline the proposed negotiation content and continue negotiation process. For these scenarios, the negotiation counterpart SHOULD use Response message, with either an objective option that contains at least one all-1 or a specific decline objective option that provides further condition for convergence.

8.5. Waiting Time Option

The waiting time option is used to indicate the negotiation counterpart to wait for the further negotiation response, since the processing might need more time than usual or it might depend on another triggered negotiation.

The waiting time option MUST be only encapsulated in Confirm-waiting messages.



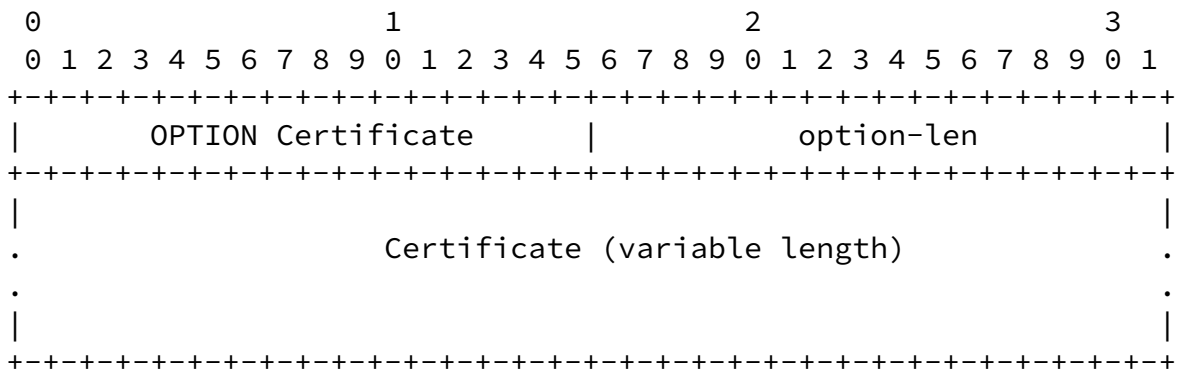
Option-code OPTION_WAITING (4).

Option-len 4, in octets.

Time The time is counted in millisecond as a unit.

8.6. Certificate Option

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



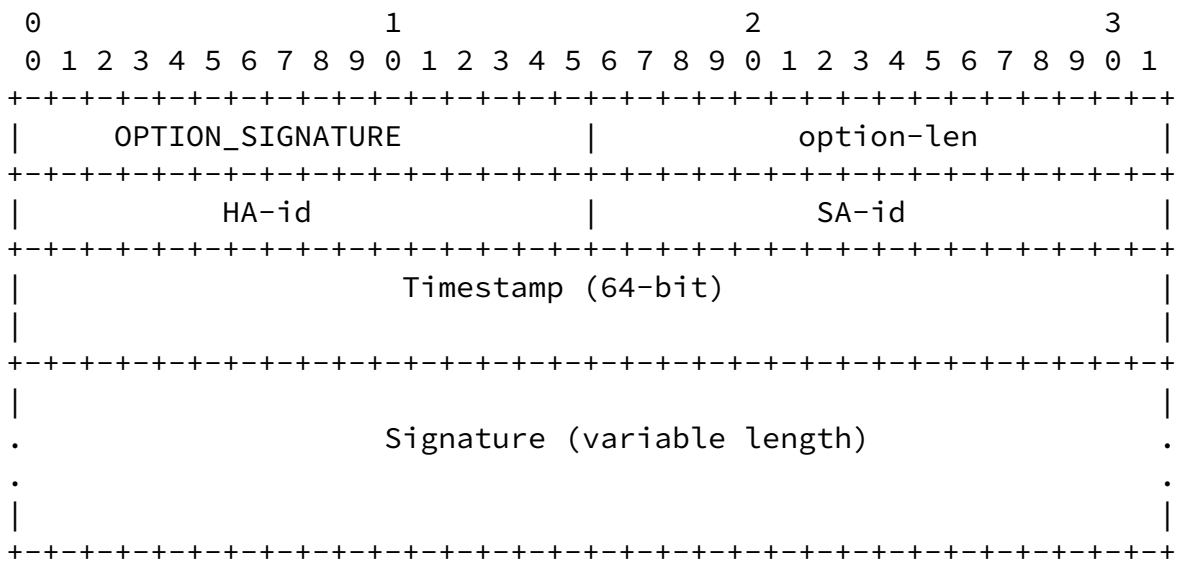
Option-code OPTION_CERT_PARAMETER (5)

Option-len Length of certificate in octets

Public key A variable-length field containing certificate

8.7. Signature Option

The Signature option allows public key-based signatures to be attached to a CNP message. The Signature option could be any place within the CNP message. It protects the entire CNP header and options. A TimeStamp has been integrated in the Signature Option for anti-replay protection. The format of the Signature option is described as follows:



Option-code	OPTION_SIGNATURE (6)
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 CNP registry in IANA. The initial values are assigned for SHA-1 is 0x0001.
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 CNP registry in IANA. The initial values are assigned for RSASSA-PKCS1-v1_5 is 0x0001.
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 CNP message header. 2. All CNP options including the Signature option (fill the signature field with zeroes). 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.

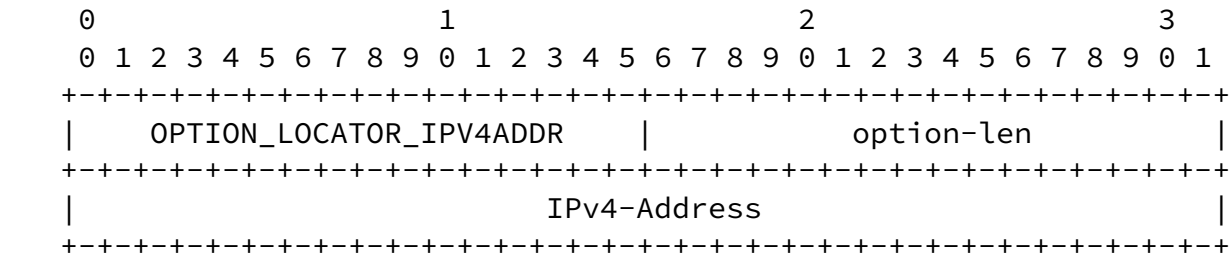
8.8. Locator Options

These locator options are used to present device's or interface's

reachability informations. They are Locator IPv4 Address Option,

Locator IPv6 Address Option and Locator FQDN (Fully Qualified Domain Name) Option.

8.8.1. Locator IPv4 address option

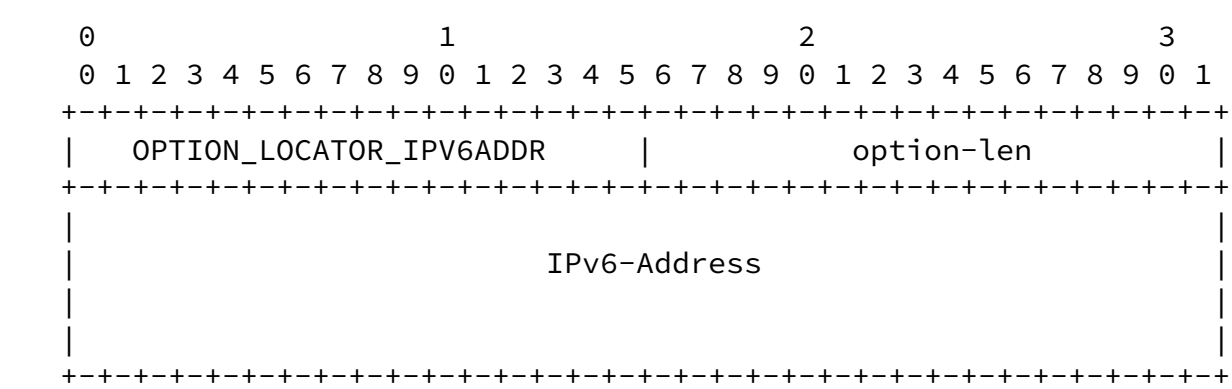


Option-code OPTION_LOCATOR_IPV4ADDR (7)

Option-len 4, in octets.

IPv4-Address The IPv4 address locator of the device/interface.

8.8.2. Locator IPv6 address option



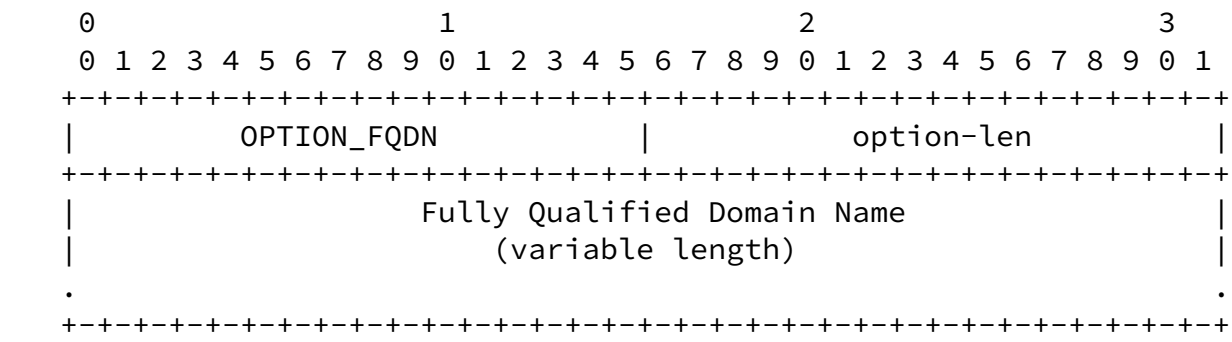
Option-code OPTION_LOCATOR_IPV6ADDR (8).

Option-len 16, in octets.

IPv6-Address The IPv6 address locator of the device/interface.

Note: link-local IPv6 address SHOULD be avoided when this option is used in the Divert option. It may create connect problem.

8.8.3. Locator FQDN option



Option-code OPTION_FQDN (9).

Option-len Lenght of Fully Qualified Domain Name in octets.

Domain-Name The Fully Qualified Domain Name of the entity.

9. Objective Options and Considerations

The Objective options contains negotiation objectives, which are various according to different functions/services. They MUST be carried by Request or Negotiation Messages only. Objective options SHOULD be signed the option type from 65 in the CNP option table.

For most scenarios, there SHOULD be initial values in the negotiation requests. Consequently, the Objective options SHOULD always be completely presented in a Request message. If there is no initial value, all 1 SHOULD be filled in.

9.1. Organizing of CNP Option

Naturally, a negotiation objective, which is based on a specific service or function or action, SHOULD be organized as a CNP option. It is NOT RECOMMENDED to organize multiple negotiation objectives into a single option.

A negotiation objective may have multiple parameters. Parameters can be categorized into two class: the obligatory are presented as fixed fields; and the optional are presented in TLV sub-options. It is NOT RECOMMENDED to split parameters in a same objective into multiple options, unless they have different response periods. The exception scenario may also be described by split objectives.

9.2. Vendor Specific Options

Option codes 128~159 have been reserved for vendor specific options. Multiple option codes have been assigned because a single vendor may use multiple options simultaneously. These vendor specific options are highly likely to have different meanings when used by different vendors.

9.3. Experimental Options

Option code 176~191 have been reserved for experimental options. Multiple option codes have been assigned because a single experiment may use multiple options simultaneously.

10. Items for Future Work

There are a few open design questions that are worthy of more work in the near future, as listed below:

- UDP vs TCP: For now, this specification has chosen UDP as message transport mechanism. However, this is not closed yet. UDP is good for short conversation, fitting the divert scenarios well. However, it may have issues with large packets. TCP is good for stable and long sessions, with a little bit of time comsumption during the session establishment stage.

- Message encryption: should CNP messages be encrypted as well as signed, to protect against internal eavesdropping within the

network?

- TLS vs built-in security mechanism. For now, this specifcation has chosen a PKI based build-in security mechanism. However, TLS may be chosen as security infrastructure for simplification reasons.

- Use case. A use case may help readers to understand the applicability of this specification. However, the authors have not yet decided whether to have a separate document or have it in this document.

- Rules about how data items are defined in a negotiation objective. Maybe a formal information model is needed.

11. Security Considerations

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

Since CNP is intended to be deployed in a single administrative domain recommended to operate its own CA, there is no need for a trusted third party.

12. IANA Considerations

Section 4 defines the following mtwpulticast addresses, which have been assigned by IANA for use by CNP:

ALL_CNP_NEIGHBOR mutlicast address (IPv6): (TBD1)

ALL_CNP_NEIGHBOR mutlicast address (IPv4): (TBD2)

Section 4 defines the following UDP port, which have been assigned by IANA for use by CNP:

CNP Listen Port: (TBD3)

This document defined a new Configuration Negotiation Protocol. The

IANA is requested to create a new CNP registry. The IANA is also requested to add two new registry tables to the newly-created CNP registry. The two tables are the CNP Messages table and CNP Options table.

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

CNP Messages table. The values in this table are 16-bit unsigned integers. The following initial values are assigned in Section 7 in this document:

Type	Name	RFCs
0	Reserved	this document
1	Request Message	this document
2	Negotiation Message	this document
3	Negotiation-end Message	this document
4	Confirm-waiting Message	this document

CNP Options table. The values in this table are 16-bit unsigned integers. The following initial values are assigned in Section 8 and Section 9 in this document:

Type	Name	RFCs

0	Reserved	this document
1	Divert Option	this document
2	Accept Option	this document
3	Decline Option	this document
4	Waiting Time Option	this document
5	Certificate Option	this document
6	Sigature Option	this document
7	Device IPv4 Address Option	this document
8	Device IPv6 Address Option	this document
9	Device FQDN Option	this document
10~64	Reserved for future CNP General Options	this document

128~159	Vendor Specific Options	this document
176~191	Experimental Options	this document

The IANA is also requested to create two new registry tables to the CNP Parameters registry. The two tables are the Hash Algorithm for CNP table and the Signature Algorithm for CNP table.

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

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

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

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

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

13. Acknowledgements

Valuable comments were received from Zhenbin Li and Dacheng Zhang, and other participants in the xxx working group.

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

14. Change log [RFC Editor: Please remove]

draft-jiang-config-negotiation-protocol-00: original version, 2013-10-19.

15. References

15.1. Normative References

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

[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.

15.2. Informative References

[I-D.jiang-config-negotiation-ps] Jiang, S., Yin, Y., and B. Carpenter, "Network Configuration Negotiation Problem Statement and Requirements", draft-jiang-config-negotiation-ps-01 (work in progress), October 2013.

[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.

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

Authors' Addresses

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

Email: jiangsheng@huawei.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com

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

Email: leo.liubing@huawei.com

Yuanbin Yin
Huawei Technologies Co., Ltd
Q15, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: yinyuanbin@huawei.com