INTERNATIONAL STANDARD

ISO/IEC 14908-4

First edition
2012-02-15

Information technology — Control network protocol —

Part 4: **IP communication**

Technologies de l'information — Protocole de réseau de contrôle —

Partie 4: Communication IP

Reference number
ISO/IEC 14908-4:2012(E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 14908-4 was prepared by CEN/TC 247 and was adopted, under a special “fast-track procedure”, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by the national bodies of ISO and IEC.

ISO/IEC 14908 consists of the following parts, under the general title *Information technology — Control network protocol*:

— *Part 1: Protocol stack*

— *Part 2: Twisted pair communication*

— *Part 3: Power line channel specification*

Part 4: IP communication

Introduction

This International Standard has been prepared to provide mechanisms through which various vendors of local area control networks may exchange information in a standardised way. It defines communication capabilities.

This International Standard is used by all involved in design, manufacture, engineering, installation and commissioning activities.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this International Standard may involve the use of patents held by Echelon Corporation.

The ISO and IEC take no position concerning the evidence, validity and scope of this patent right. The holder of this putative patent right has assured the ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of the putative patent rights is registered with the ISO and IEC. Information may be obtained from:

> Echelon Corporation, 4015 Meridian Avenue, San Jose, CA 94304, USA, phone +1-408-938-5234, fax: +1-408-790-3800 http://www.echelon.com.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

INFORMATION TECHNOLOGY – CONTROL NETWORK PROTOCOL –

Part 4: IP communication

1 Scope

This International Standard specifies the transporting of the Control Network Protocol (CNP) packets for commercial local area control networks over Internet Protocol (IP) networks using a tunnelling mechanism wherein the CNP packets are encapsulated within IP packets. It applies to both CNP nodes and CNP routers.

The purpose of this International Standard is to insure interoperability between various CNP devices that wish to use IP networks to communicate using the CNP protocol.

The main body of this International Standard is independent of the CNP protocol being transported over the IP network. The reader is directed to Annex A and Annex B for the normative and informative, respectively, aspects of this specification that are specific to ISO/IEC 14908-1.

Figure 1 shows a possible configuration of such CNP devices and networks connected to an IP network.

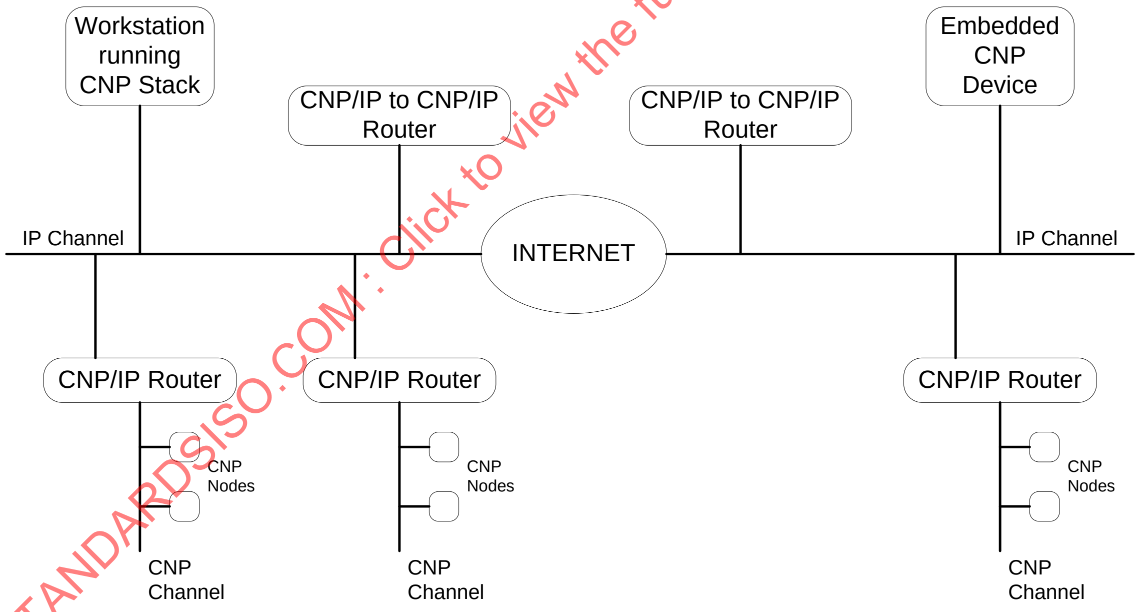

Figure 1 — Typical CNP/IP application

Figure 1 depicts two types of CNP devices: CNP nodes and CNP routers. It should be noted that the routers shown can route packets between typical CNP channels (such as twisted pair or power line) and an IP channel or it can route CNP packets between two IP channels. In this International Standard the IP channel will be defined in such a way to allow it to be used like any other CNP channel.

In the above diagram the IP network can be considered to be one or more IP channels. This International Standard covers only how CNP packets are transported over IP channels. It does not cover how CNP packets are routed between standard CNP channels and IP channels. This specification is not intended to cover the lower layers (physical, MAC and link layers) of either standard CNP or IP channels.

2 Normative references

None.

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1
tunneling
encapsulation of one protocol's packet within the payload of another protocol's packets

3.1.2
channel
common communications transport mechanism that a specific collection of CNP devices share and communicate over without the use of a router

NOTE 1 Channels are used to transport CNP packets below the link layer of the CNP protocol stack.

NOTE 2 Typically this refers to some type of physical media such as power line, RF, or twisted pair, but in the case of IP networks this channel is not physical, but a protocol tunnel.

3.1.3
CNP device
device that uses the CNP protocol to communicate with other CNP devices

NOTE Specifically a CNP/IP device is a CNP device that communicates with other CNP devices over an IP channel.

3.1.4
CNP router
special type of CNP device that routes CNP protocol packets between two or more channels

NOTE Specifically a CNP/IP router is a CNP router in which at least one of the channels it routes packets over is an IP channel.

3.1.5
CNP node
special type of CNP device that can send or receive CNP protocol packets, but does not route them between channels

NOTE 1 Specifically a CNP/IP node is a CNP node in which at least one of the channels it sends and receives packets over is an IP channel.

NOTE 2 All CNP devices are either routers, nodes or both.

3.1.6
CNP group
collection of CNP devices that share a common multicast address

3.1.7
node ID
logical network address that differentiates nodes within the same subnet or domain

3.1.8
Must Be Zero (MBZ)
reserved field that may be used in the following versions of the protocol

NOTE Such fields shall be sent as zero and ignored by the receiver in implementations conforming to the current version of the specification.

3.2 Abbreviations

CTP	Channel Timeout Period
CNP	Control Network Protocol
LFS	Last Forwarded Sequence
MBZ	Must Be Zero
NTP	Network Time Protocol
PSN	Packet Sequence Number
SA/DA	Source Address / Destination Address
SID	Session Identifier
SNTP	Simple Network Time Protocol
UDP	User Datagram Protocol

4 Requirements

The following is a set of general requirements for the transporting of CNP packets over IP channels:

- be as efficient as possible to allow quasi real-time operation;
- be independent of the application level interface used to receive the packets. For example the tunnelling protocol should not rely on the existence of a socket interface or how that interface may be used;
- insure that CNP packet ordering is preserved;
- insure that CNP packets that are “stale” (outside the maximum timeout characteristics of the IP channel) are not forwarded;
- detect packets that get duplicated in the IP network;
- support IP routing devices that prioritise IP packets;
- optional security measures to prevent malicious users from tampering with devices;
- scalable;
- allow status information to be extracted from CNP/IP devices;
- support the exchange of configuration information between CNP/IP devices and configuration servers.

5 CNP/IP device specification

5.1 IP Related device specifications

A CNP/IP device shall behave like any standard IP host capable of exchanging IP packets with any other IP host either on the same IP subnet or anywhere else in the Internet cloud. A CNP/IP device shall have a single unicast IP address and may be capable belonging to as many as 32 multi-cast groups. It is optional that a CNP/IP device support multi-casting. This document does not address the routing of IP packets between subnets or through the Internet. The CNP/IP devices shall be compatible with whatever standard mechanisms (IP routers, switches etc.) are required to perform the IP routing functions.

5.2 CNP related device specifications

5.2.1 Packet formats

The general format of CNP packets which are tunnelled over the IP channel are those packets that are received from or sent to the Link layer (layer 2) of the CNP protocol stack. Refer to Annex A for a precise specification of the packet formats corresponding to the CNP protocol.

5.2.2 Addressing schemes

Different CNP protocols generally use different addressing schemes to exchange packets. Although it is generally not necessary to understand the contents of a CNP packet or its addresses in order to tunnel CNP packets over IP, some aspects of the CNP addressing scheme are reflected in the process of configuration. This is especially true when it comes to setting up the IP channels that are used for tunnelling. Since CNP protocols use different addressing schemes the terminology used in the main body of this specification for describing addresses are meant to be general and rich enough to describe the superset of addressing schemes used in all CNP protocols. The following CNP addressing terms are used in this specification.

— Unique ID. This refers to an ID that is globally unique to all devices within a specific protocol. Unique ID's are generally fixed in nature in that they never change through the life of a device.

— Domain. This is the highest level of a three level hierarchical addressing scheme. Domain ID's should be unique within a particular network. This means that in a particular network where Domains are used if two devices have the same Domain ID they belong to the same Domain. Domain ID's are generally logical in nature and can be changed and configured.

— Subnet. This is the middle level of a three level hierarchical addressing scheme. Subnet ID's should be unique within a particular domain. This means that in a particular network where subnet ID's are used if two devices have the same Domain ID and the same Subnet ID then they belong to the same Subnet. Some CNP's do not use Domains in which case the Subnet may be the highest level of address for a device. Subnet ID's are generally logical in nature and can be changed and configured.

— Node. This is the lowest level of any hierarchical addressing scheme. Node ID's should be unique within a particular Subnet. No two devices within the same subset should have the same Node ID. Node ID's are generally logical in nature and can be changed and configured.

— Group. Groups are an orthogonal addressing scheme to the hierarchical Domain/Subnet/Node triplet just described. Groups are used to allow multi-casting of messages. Some CNP's may not support group addresses and even those that do will have different rules for how they relate to the other addressing schemes. These considerations are not relevant to this specification.

The definitions above are fairly general and are provided as a guideline for how to map the CNP protocol to these terms. In general how the various addressing schemes work within a CNP protocol are not relevant to this specification. It is only necessary to know what the various addressing terms refer to.

Of special note is how these addresses are used for routing within the CNP protocol. Therefore a table is given in the appendix that specifies how the appropriate addresses used in that protocol map to the terms given above.

6 IP channel

6.1 Specification

IP channels are not like typical CNP channels that currently exist. Typical CNP channels are physical busses by nature. This implies that all devices on the channel will by default receive all packets transmitted on that channel. In addition when a new device is added to the channel it is not necessary that other devices on the channel become aware of it before they can exchange packets. To transmit a packet on a channel it is only necessary that a device be capable of physically transmitting the packet on the channel, nothing more. If a device is simply physically connected to a channel it is capable of exchanging packets with other devices on the channel.

By contrast an IP channel is not physical, but logical in nature. There are a number of different physical media that can support IP communications and any of them should be capable of supporting a CNP channel. Because we are dealing with a logical channel it is necessary to “construct” the channel by informing each device on the channel of the existence of the other devices on that channel. In other words before a device can transmit a packet to some other device on an IP channel it shall be made aware of how to specifically send a packet to that device, i.e. its IP address.

Another significant difference between physical and logical channels is that in the case of typical physical channels it is possible to calculate fixed upper bounds on the length of time it will take a packet to traverse from one device to another once the packet is transmitted on the channel. This is not always possible for IP networks. The deviation of packet delivery times between CNP devices on an IP channel are much higher than those experienced with typical CNP channels.

As depicted in Figure 1 the IP channel is used as an intermediary transport mechanism for the CNP packets by a variety of CNP/IP devices. When a CNP packet is transported on an IP channel, an IP message encapsulating the CNP packets is sent to other CNP/IP devices on that IP channel. On reception of one of the IP messages by a CNP/IP device the CNP packets are extracted and processed. A single IP message may contain more than one CNP packet. Therefore the IP messages shall be formatted in such a way to allow the extraction of the individual CNP packets. This is referred to a packet “bunching”. CNP/IP devices shall support the reception of bunched packets. Likewise the bunching shall be done in such a fashion that each CNP packet contained within a bunched IP message is complete, i.e. CNP packets should not cross IP message boundaries as a result of bunching. It is also a requirement that intermediate IP devices be capable of unbundling bunched CNP packets and bunching them in a different manner before forwarding them.

The IP channel is specified by the list of unicast IP addresses, exactly one for each CNP/IP device. There is no maximum to the number of CNP/IP devices on a single IP channel.

If every CNP/IP device on an IP channel contained a list of unicast IP addresses for every other CNP/IP device on that IP channel, this is all that would be required to enable the tunnelling of CNP packets. In the most brute force approach, for each CNP packet to be forwarded on the IP channel a separate unicast IP message could be sent to each CNP/IP device in the channel. This does not scale very well so the following techniques will be used to reduce the IP traffic:

— IP multi-cast groups;

— selective forwarding.

IP multi-cast groups allow a single IP message to be sent to more than one CNP/IP device. Therefore a complete definition of a CNP/IP channel should contain not only the unicast IP addresses of all the CNP/IP devices on the channel but also the IP multi-cast groups to which they belong. Each CNP/IP device can belong to up to 32 multi-cast addresses.

Selective forwarding refers to examining the contents of the CNP packet before forwarding it to determine if it should be sent to a particular CNP/IP device. In order to do this additional CNP specific information shall be known about each potential destination. If the CNP/IP device is a router then the information necessary to perform selective forwarding is the routing tables of the CNP/IP router. If the device is simply a node then the domain, subnet, node id, unique id, and CNP groups that the node belongs to should be known. Therefore all this information is also part of a complete IP channel definition. In short a complete IP channel definition contains all known information that may be relevant to the forwarding of packets to the other CNP/IP devices in the IP channel. It is the universe of all relevant knowledge about the IP channel.

It is important that whatever forwarding scheme is used by a CNP/IP device the following conditions are always true:

a) CNP protocol packets are always received by all CNP/IP devices on the IP channel that need to receive them regardless of whether they are routers or nodes. If there is any ambiguity or uncertainty concerning which CNP/IP devices should receive a CNP packet then that packet may or may not be discarded depending upon specific implementation considerations of the device. The device may either forward the packet to all devices on the channel or it may simply discard it and not forward it to any;

b) a specific CNP packet should never be transmitted twice to the same CNP/IP device unless it is because of some retry mechanism above the link layer of the CNP protocol stack. Due to the nature of IP networks it may happen that a CNP/IP device may receive duplicate IP messages, but this should never be the result of the message being transmitted more than once from another CNP/IP device.

In addition selective forwarding can be performed on multi-cast groups if the groups were formed based upon some criteria. For example multi-cast group 'A' may contain all CNP/IP devices belonging to domain ID 'W'. If a CNP packet is destined for domain 'W' then it would be sufficient to forward it only to multi-cast group 'A'. In order to perform the selective forwarding on multi-cast addresses it is necessary to know if these groups were formed based upon some criteria.

In recognition of the fact that the complete IP channel definition can be unwieldy to use and maintain it is not a requirement that a CNP/IP device use it to forward packets. An alternative data structure called the "send list" can be maintained within each CNP/IP device. The send list may contain both unicast and multi-cast addresses and is subject to the same conditions given above. It can be created and loaded into the CNP/IP device with third party configuration tools that are better suited to creating multi-cast groups based upon some criteria. The send list represents the minimum amount of information required to allow proper forwarding of CNP packets and is structured to simplify the forwarding process such that the CNP/IP device need only forward packets to every address (unicast or multi-cast) in the send list. In order to allow a CNP/IP device to blindly forward packets to each address in the list the following conditions shall be true:

i) CNP protocol packets shall be received by all CNP/IP devices that need to receive them regardless of whether they are routers or nodes (same as above);

ii) a specific CNP packet is never transmitted twice to the same CNP/IP device (same as above);

iii) if device A is a destination in device B's send list then device B should be a destination in device A's send list. This is necessary to support the acknowledged service of the CNP protocol.

It should be possible to perform simple forms of selective forwarding using the send list by associating characteristics with the multi-cast entries in the list.

In general it is important to note that the IP channel definition represents complete global information about the IP channel while the send list is derived and may result from an intelligent grouping of devices based upon some characteristic. The send list's main purpose is to allow fairly efficient operation of CNP/IP devices without requiring them to do extensive processing of the complete channel definition list. It is also important to note that the send list is a configured property of a CNP/IP device meaning that it is controlled and input to a device through some explicit configuration process. Although the send list is a configured property it does not preclude a CNP/IP device from doing self-configuration and calculating its own send list.

In order to have tight controls over the behaviour of CNP/IP devices and how they forward packets it should be possible to configure a CNP/IP device to use an explicit send list and ignore any IP channel configuration information it may have.

6.2 IP transport mechanisms

6.2.1 General

IP is a Network level protocol as shown in Figure 2. It is designed to operate over a wide range of physical media and link layer protocols. As such this document does not specify anything about the link or physical layers of the IP stack.

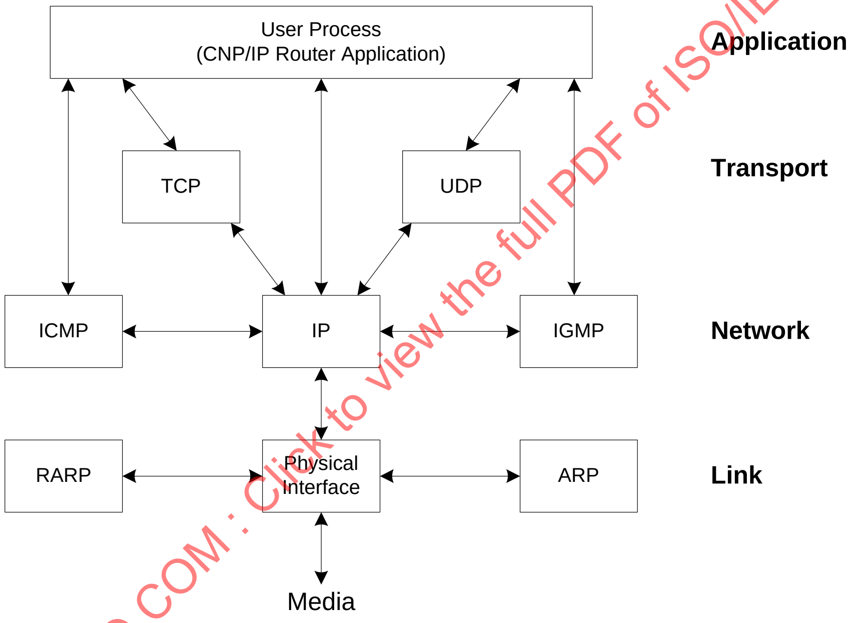

Figure 2 — IP protocol stack

As depicted in Figure 2 the three most common mechanisms used to transport IP packets are the following:

— raw IP;

— TCP;

— UDP.

TCP (refer to RFC 793) and UDP (refer to RFC 768) are transport protocols built on top of IP (refer to RFC 791). Given the increased efficiencies of UDP regarding the transport of CNP data messages and its support of multi-cast addressing, it will be used to communicate between CNP/IP devices. All CNP/IP devices shall support UDP. TCP has some advantages for use in the configuration process and may be supported for certain types of messages in addition to UDP. TCP support in CNP/IP devices is optional.

To address the sequencing issue there will be a sequence number added to the header of packets to help in sequencing them. All UDP datagrams shall be transmitted with valid checksums.

In order to send a packet via TCP/UDP it is necessary to define a port number in addition to the IP address. In general port numbers are configurable and will be used for different purposes as defined below. Refer to Annex B for recommendations on port number to use for CNP.

Using UDP, datagrams can be sent using either unicast or multi-cast addressing. Unicast is point-to-point meaning that a datagram is sent from one IP host to a single other IP host. It is much more efficient to use multi-cast addressing when sending the same datagram to multiple IP hosts. Therefore it is recommended that CNP/IP devices support both unicast and multi-cast IP addressing. Recall that an IP channel is defined by a list of IP addresses. A channel definition or send list can contain any combination of unicast and multi-cast addresses. It is not required that a CNP/IP device support multi-cast in order to inter-operate with a CNP/IP device that does.

In order for a CNP/IP router to use multi-cast addressing across IP routers it will be necessary for the CNP/IP device to inform the IP router of its intention to join the multi-cast group. There are well-established methods for doing this and it is beyond the scope of this document to specify how it is done. The reader should consult RFC 1112.

6.2.2 Informative considerations

Some IP networks contain NAT routers. These routers cannot handle protocols which embed IP addresses in their payloads unless they are specifically designed to do so. The same will be true of the tunnelling protocol specified in this document. In general this protocol will not work across NAT routers. The protocol can still be used in a network that uses NAT routers, as long there exists a router that is capable of handling this protocol. That could either be the NAT router itself or another CNP/IP to CNP/IP router that sits in the same area of the network as the NAT router.

7 CNP/IP device

7.1 Configuration of a CNP/IP device

The CNP/IP device has a dual personality. From a CNP point of view, it is a CNP node on a CNP channel and has all such characteristics. These parameters can be configured and managed using the standard CNP network management procedures and messages.

From the IP point of view, the CNP/IP device is an IP host on an IP network and thus has to be configured like any other host on an IP network.

In addition, there is configuration information that defines the logical IP channel associated with that CNP/IP device.

This clause describes only those elements relevant to configuring the IP host and IP channel parameters.

In general all IP host and channel parameters will be configurable using a number of techniques and protocols. As a minimum all CNP/IP devices shall support manual configuration of the forwarding mechanism for that device in order to guarantee a minimum level of interoperability between devices that may be configured in different ways. By forwarding we mean the act of tunnelling as described above to the other devices on the IP channel.

7.2 Configuration parameters

7.2.1 General

This subclause identifies the parameters that a CNP/IP uses (or may use) to operate. This subclause is not intended to define the data structures used to store the information or define the messages that are used to exchange them. Its only purpose is to have a consolidated section in which all the parameters of a CNP/IP device are identified and defined. Subsections discuss mechanisms for communicating this information between devices.

There are three relevant data sets that form the parameters contained within a CNP/IP device:

— CNP/IP channel definition as described in Clause 6;

— send list as described in Clause 6;

— device parameters relevant to its existence on a CNP/IP channel.

7.2.2 Channel definition parameters

A complete channel definition is logically a list of every CNP/IP device on the channel. Each of the devices on the channel can be associated with the following type of information:

— multi-cast support (yes or no). Since this is optional there shall be an indication of whether it is supported;

— TCP support (yes or no). Since this is optional there shall be an indication of whether it is supported;

— CNP/IP device type (router, node, proxy etc.);

— CNP router type (repeater, learning, configured etc.);

— CNP “Wants all Broadcasts” flag;

— name. Simple text string used for identification purposes;

— channel timeout. This parameter is global to the channel. Each device has this value but it is the same for all devices;

— IP address. This is the unicast IP address of the device.

— unicast port for listening to data;

— list of multi-cast address/port number pairs a CNP/IP device listens on;

— CNP specific unique device ID 1 (router near side or node);

— CNP specific unique device ID 2 (router far side);

— CNP specific unique device ID 3 (auxiliary for configuration);

— CNP Domain length and ID, subnet, node address for each domain;

— the parameters specific to nodes are: CNP group membership info;

— the parameters specific to routers are: CNP routing table.

Note that this list is representative in nature. Complete details as required are left to later clauses of this International Standard.

The tunnelling protocol defined in this specification does not require any specific CNP addressing scheme. The following CNP address types are supported:

— unique device ID;

— domain ID;

— subnet ID;

— node ID;

— group ID.

Refer to Annex A for the specific addressing conventions that correspond to the address types listed above.

7.2.3 Send List arameters

The following parameters are used to define the Send List.

— list of unicast IP addresses and ports;

— list of multi-cast IP addresses and ports.

7.2.4 Device parameters

— IP gateway address;

— IP subnet mask;

— configuration server IP address/port;

— SNTP server IP address.

7.3 Configuration techniques

7.3.1 General

This subclause describes the various techniques that can be used to set the parameters defined in the previous subclause.

These parameters can be set in a number of ways. It is not necessary for a CNP/IP device to support all these methods, but if it supports any of these methods it should support them in a standard fashion.

7.3.2 Manual configuration

For manual configuration there is no configuration server. All the required channel definition parameters send list parameters and device parameters need to be manually entered. There is no standard method for doing this. It is vendor specific and can be accomplished by any of the following methods:

— configuration files;

— terminal interface;

— telnet interface;

— FTP file transfer;

— HTTP interface.

7.3.3 BOOTP and DHCP

7.3.3.1 Background

Two mechanisms exist for devices to obtain an IP address without prior configuration: DHCP (RFC2131]) and BOOTP (RFC951).

DHCP is actually an extension of BOOTP and BOOTP servers that are compliant with RFC951 can understand messages from DHCP clients and respond to those messages correctly.

Basically, a system will boot and make a DHCP request for an IP address from a DHCP server. The DHCP server responds with a valid, unused IP address that the DHCP client can now use.

7.3.3.2 Compliance

Devices that are compliant with this specification and wish to obtain an IP address without prior configuration shall implement a DHCP client and may implement BOOTP client.

7.3.4 Configuration servers

It is desirable for the devices on a CNP/IP channel to be as “plug and play” as possible. Given the large amount of configuration information that a CNP/IP device uses to operate, it is desirable to have a mechanism for distributing it so it does not need to be individually entered into each device. The device that allows for this mechanism to work is called a configuration server.

Devices that use a configuration server shall inter-operate with devices that do not use one. However, it is not necessary for a configuration server to support all the interactions presented in this specification. Specifically *Channel Routing* packets need not be supported.

A configuration server uses IP messages to configure CNP/IP clients in an automated fashion. In general a configuration server may support the following functionality:

- configuration of various client CNP/IP parameters;
- distribution of IP channel definition and send list to client CNP/IP devices. The channel list is a description of the network as a whole whereas the send list is potentially unique to each CNP/IP device;
- automatic maintenance of the channel definition list by detecting when CNP/IP devices come on and go off line.

In addition to the parameters described in 7.2, a CNP/IP device shall also have the IP address of a configuration server, and a port to communicate on to use a configuration server.

CNP/IP devices shall send a device registration message to the configuration server on power up, on reset, and when parameters in the devices channel definition list changes.

This clause is not intended to specify how the server manages the list of parameters that it distributes to the clients. In fact, these parameters should be maintainable using any management method desired. It is recognised that it is desirable that the server has the ability to dynamically create and maintain parameters as clients come on and off line. To that extent the configuration protocol and message formats will be designed to allow servers to support this functionality.

8 CNP/IP messages

8.1 Definition of CNP/IP messages and modes of operation

The purpose of this clause is the following:

- identify all messages that can be exchanged between CNP/IP devices on an IP channel;
- define the message contents;
- specify the protocol and behavioural aspects of the devices while they are exchanging these messages.

Clause 9 will specifies the precise packet formats for messages defined in this clause.

A CNP/IP device uses the IP channel for a variety of purposes and modes of operation. A separate clause in this International Standard will cover each of these. The modes of operation include the following:

- exchanging (tunnelling) CNP data packets;

— exchanging configuration information with configuration servers;

— miscellaneous status messages;

— vendor specific messages and protocols.

For each of these modes of operations a set of messages will be defined which are exchanged between the CNP/IP devices.

8.2 Common message header

Each IP message exchanged by devices on the IP channel are preceded by a fixed header with a format that is common to all CNP/IP messages. This header contains the following fields:

— Version;

— Protocol Flags;

— Vendor code;

— Packet type;

— Data Packet Length;

— Header Size;

— Session ID;

— Sequence #;

— Time Stamp;

— Security Key.

Specific message formats are covered in Clause 9.

It is assumed that all packets with the same Version Number can always be parsed. If a packet is received with an unknown Version Number, this packet should be discarded without further processing.

The Protocol Flags specifies information about the packet such as which CNP protocol packet is encapsulated within the message and whether the message was sent using security.

The Vendor Code allows for vendor-specific packets. This value shall be set to 0 for all packets of standard definition according to this specification. Vendor-specific packets are identified in part by a unique Vendor Code (other than 0).

A unique Packet Type code is assigned to each function as detailed in 9.1. Standard Packet Type codes that are defined within this specification are in the range 0x00 to 0x7F. Vendor-specific packets that convey information as an extension to a standard function defined in this specification may use the same Packet Type code as that standard function, however the Vendor Code shall be set to the unique identifier for that vendor. Vendor-specific packets that have no relationship to existing standard functions defined in this specification shall use a Packet Type code in the range 0x80 to 0xFF.

The Data Packet Length and the Header Size specify the size of the packet and the size of the header respectively.

The Session ID works in conjunction with the Sequence Number to minimise the occurrence of duplicated sequence numbers.

The Time Stamp is used for detecting stale data packets and is based on a time reference that is synchronised among all devices in the CNP/IP channel as defined in 8.4.4.

The Security Key is used for securing packets as described in 8.8.

All CNP/IP messages may be sent using UDP datagrams. Each UDP datagram may contain one or more CNP/IP messages. If there is more than one CNP/IP message within a single UDP datagram (bunching) then each message will have its own header. Since each header has message size information it is possible to extract each message individually.

All CNP/IP devices shall support packet bunching. Figure 3 depicts more than one CNP/IP message within a single UDP datagram (packet bunching).

Common Packet Header Message n-1	Data Portion of Message n-1	Common Packet Header Message n	Data Portion of Message n

Figure 3 — Packet bunching

Certain CNP/IP messages may cross a UDP datagram boundary. In such cases a special segmentation protocol is defined which can be used to facilitate this. Messages that cross a UDP datagram boundary and use the segmentation scheme need not to be bunched with other packets.

8.3 Packet segmentation

8.3.1 Overview

Responses to configuration requests for certain CNP/IP data structures have more data than can be carried in a single UDP packet due to the maximum of 548 bytes of payload per UDP packet. This subclause describes a common method by which these data structures can be conveyed. Note that this method does not inherently prevent data skew where portions of the entire data structure change between the transfer of individual portions. In all of the Configuration Response Packet definitions in this protocol, either A) the response packet format is specifically tailored to eliminate the sensitivity to data skew or B) a simple method is provided by which potential data skew can be detected.

This protocol involves the segment packet type. Any other configuration packet type may be carried as a *payload* in a series of segment packets. Sequences of bytes from the payload packet are selected so that the resulting segment packet will fit in a UDP frame and these segments are provided as responses. If the requested packet will fit in a UDP frame, then it is sent without encapsulation.

Packets sent in segment packets are sent in their entirety. That is the payload packet has a *common packet header* with a packet length of the payload packet. All segment packets have the same packet type.

In order to facilitate this method, each Configuration Request packet that invokes a Configuration Response contains a *Segment ID* field and each Segment packet contains a *Valid* bit and a *Final* bit within a Flags field as well as a *datetime* indicator.

— The ***Request ID*** field is copied from the Request to the segment packet by the responding server. This represents a tag value applied by the source of this request so that it may uniquely distinguish the response to this request. Thus multiple requests may be outstanding simultaneously to the same destination. The Request ID field is 16 bits so that any device information in addition to a request number can be included in this field in an implementation dependent manner.

— The ***Segment ID*** is a reference to a partial block of data within a complete data structure. The Segment ID referencing the first partial block of data is 0x0.

— The ***Valid*** bit in a segment Packet indicates the validity of the data in this packet pertaining to the indicated Segment ID. Valid Set means the data is valid. Valid Clear means there is no valid data for the indicated Segment ID.

— The ***Final*** bit in a segment Packet indicates the existence of additional data within the structure for Segment ID values greater than that returned in this packet. Final Set indicates there is no additional data.

— The ***datetime*** indicator is used to detect acquired data skew. This field indicates when a particular data structure became valid. It is not an indicator of when the data was requested for transfer. For configuration data that is sensitive to it, data skew can be detected by comparing the *datetime* field returned in each of the response packets that represent the entire data structure. If the *datetime* field is not the same in all packets, then some portion of the acquired data has been skewed. The format of the datetime field is specified in 9.5.

8.3.2 Segment exchange

8.3.2.1 General

The typical method by which configuration data is obtained is as follows:

a) the requesting node sends a Request Packet with *Segment ID* = 0. It may optionally set the REQUEST_ALL flag to indicate that all segment packets are to be sent without waiting for additional Request packets;

b) if the response message fits in one frame, then the frame is sent. When the receiver receives a packet of the requested type, it retires the Request ID, and acts on the response;

c) if the response packet does not fit in a single frame, the responding node sends a segment packet with *Segment ID* equal to the Segment ID in the Request Packet. Additionally, if data does not exist for that segment, the response packet is sent with *flags:final* set and *flags:valid* clear. If data does exist for that segment, it is included with *flags:valid* set. Additionally, if more data exists beyond the indicated *Segment ID,* the data portion of the current packet is filled to its capacity, and *flags:final* is cleared;

d) if the requesting node does not receive a response, the request message is repeated until a response is received. There is an increase at an exponential rate for the repeat interval with a maximum interval of 30 s;

e) the requesting node examines the *flags* field of the response message. If *flags:valid* is clear, the remainder of the response packet is ignored. Jump to step g).

f) if *flags:valid* is true, the data is processed. If data skew is to be detected, the *datetime* field is examined.

g) if *flags:final* is set, the request sequence for this data structure is terminated and the acquired structure is marked as valid.

h) if *flags:*final is clear, another Request Packet is sent with *Segment ID* equal to one greater than in the previous request. The process continues with step c). If the datetime for any received packet differs from any other, then the process starts with step a).

It is equally valid to start acquisition of a data structure with any Segment ID value, however a request for Segment 0 will usually result in a response with valid data. Even if the request starts with a non-zero segment, if the response fits in a single UDP frame, the response does not use the segment packet, but responds with the response packet only.

When the requesting node has received a stream of segment packets with a complete set of *Segment IDs* and the same datetime in all packets, then it assembles the payload packet and processes it. If any of the parts have a different datetime, the entire set is discarded and the process starts again with step a). No segments are kept since the response may contain segments with an entirely new datetime.

8.3.2.2 Request ID Values

Segmented packets are never sent by the server in an unsolicited manner.

The Request ID value is transmitted from the device to the server in the request packet and its value is entirely under the control of the requesting device. The Request ID may be any value including zero and it will be echoed in the segment packets transmitted by the server. The value of the Request ID shall be unique for each simultaneously active request. If this were not true, then a request for another segment would be ambiguous in the server. Server behaviour in this case is undefined.

8.3.3 Discussion

8.3.3.1 General

This subclause describes the segmentation process in further detail. The architectural implications of this design and the server and device implementation requirements are described.

8.3.3.2 Purpose and scope

This segmentation scheme allows packets of arbitrary format to be exchanged reliably over a link with a limited frame size. After a sequence of segments is received, the payload from each segment is concatenated to form a byte stream and this byte stream is interpreted as a packet. This packet contains another common packet header with a packet type code and a packet length.

Any future packet can be encapsulated and segmented in this same way. No additional fields are required in the common packet header, and no additional fields are required in each of the current or future packet formats.

Packet segmentation is not intended to be applied to data packets.

Packet segments can be requested in any order, and repeatedly, and if the server responds with packets with the same datetime, each response having the same datetime and Segment ID, shall contain the same bytes as any other such response.

8.3.3.3 Server Implementations

A variety of implementations in a server can assure that the datetime field indicates that the sequence of packets for a Request ID is consistent. Among these are:

— locking down the database for the duration of any unsatisfied requests and applying outstanding updates at the end of a last request;

— taking a snapshot of the data for a request and retiring that snapshot when the final packet has been sent or on some timeout if the request / response sequence is never completed.

8.3.3.4 Device implementations

A variety of implementations in a device can assure that a request that is not completely answered does not continue to consume resources. Among these are:

— sequestering the segments as they arrive and retiring these segments and the control data when:

1) a timeout occurs indicating that the server is not answering requests. (After a suitable retransmission policy);

2) a response of a suitable non-segmented packet indicates the request is satisfied. (No explicit Request ID field is added to the packet header, but this condition is not ambiguous since a device only communicates with a single configuration server, so there is no need to support multiple outstanding requests of the same type with that server.);

3) a sequence of segment packets has been received with the correct Request ID and all with the same date time. The payload of these segments is then decoded as a packet that is the response to this request.

8.4 Data packet exchange

8.4.1 General

This subclause defines how CNP data packets are exchanged over IP channels. Specifically it defines ***how*** CNP packets are forwarded to another CNP/IP device. It does not cover how to determine ***which*** CNP/IP devices to forward CNP data packets to. It is assumed that this determination is made using either the IP channel definition or the send list. Therefore for the sake of discussion let us define the “forward” list to be a list of IP addresses that will receive a particular CNP data packet. Note that the forward list can contain both unicast and multi-cast IP addresses. Again how this forward list is determined is not relevant to the following discussion.

Since CNP uses an end-to-end acknowledged service the following assumptions are made:

— there is no need to add acknowledgements to the forwarded CNP/IP data packets;

— there is no need to re-transmit dropped IP packets.

On the other hand the following conditions shall be maintained in order for CNP to operate properly:

— the packet ordering of CNP packets shall be maintained;

— the sender of CNP packets needs not send more than one copy of a CNP packet to each other member of the CNP/IP channel;

— the receiver shall detect duplicate CNP/IP packets and they need not be forwarded;

— packets that exceed the timeout characteristics for the channel need not be forwarded. These are referred to as “stale” packets.

CNP data packets are exchanged between CNP/IP devices using tunnelling over UDP. Each CNP packet is encapsulated with a header within a UDP datagram. The combination of header and CNP data packet payload is referred to as a “CNP/IP data message.” The header portion is referred to as the "CNP/IP data message header" and the CNP data packet is referred to as the "CNP/IP data message payload". Note that a single UDP datagram may contain more than one CNP/IP data message.

Given a particular forward list a CNP/IP device simply forwards the same CNP/IP data messages to each of the addresses in the list using UDP. Because of the characteristics of IP networks and UDP simply forwarding the messages will not guarantee that the destination host does not receive duplicate, out of order, or stale packets. Therefore it is necessary that the sender of the CNP/IP data messages include additional information to insure that the receiver can deal with each of these conditions appropriately. The following subclauses discuss these issues.

8.4.2 Out of order packets

Packets that are detected as being out of sequence by the receiver need not be forwarded. The CNP/IP device should attempt to re-order packets that are received to correct sequencing problems, but in no case should they be forwarded if they are known to be out of order. If re-ordering is not supported or is not possible then the packets shall be dropped.

The following algorithm should be used to insure that packets are not forwarded out of sequence by the receiver.

— Each CNP/IP data source maintains a session ID (SID) and an unsigned 32 bit Packet Sequence Number (PSN) for each data packet IP destination address. Each CNP/IP data message sent by a data source to a particular IP destination address contains this SID/PSN pair. Each PSN is incremented by 1 after each message is sent whereas the SID does not change between successive CNP/IP data messages. The SID may be common across all the various IP destination addresses that the CNP/IP address is sending messages to.

- If a CNP/IP device is starting a new “session” to a particular destination address (i.e. after a power cycle or re-boot) then it shall use an SID that is different than the previous SID used. In general SID’s remain constant for successive messages while the PSN increments.
- PSNs wrap from 0xFFFFFFFF to 0x00000000. Furthermore let X and Y be PSN’s. It follows that X < Y if (X – Y) < 0x80000000 assuming unsigned 32-bit arithmetic is used.
- Each CNP/IP data sink device maintains a "Last Forwarded Sequence" number (LFS) for each IP Source Address/Destination Address (SA/DA) pair from which a data packet is received.
- Initial value of LFS for each SA/DA is set equal to the PSN of the CNP/IP data message just received from the SA/DA if any of the following conditions are true:
  - after start-up of the data sink device;
  - if the SID is different from the previous CNP/IP data message received from that SA/DA.
- Reception by a data sink of a CNP/IP data message with PSN = LFS + 1 causes the data packet to be forwarded. LFS is incremented by 1.
- Reception by a data sink of a CNP/IP data message with PSN > LFS + 1 may cause the packet to be held in escrow, awaiting arrival and in-sequence forwarding of all other packets with (LFS + 1) < PSN < (PSN of first escrowed packet).
- If (escrowed packets exist) AND (time since reception of the first escrowed packet is greater than the channel timeout period < 1,5 s maximum) then the wait is abandoned for all packets in the gap with PSNs between (LFS + 1) and (PSN of first escrowed packet). LFS is set equal to (PSN of first escrowed packet - 1). Forwarding of packets continues with the next in-sequence packet, either escrowed or live.
- Reception by a data sink of a CNP/IP data packet with PSN < (LFS + 1) is discarded as a duplicate packet.
- Escrowing of CNP/IP data packets from one SA/DA does not affect forwarding or escrowing of CNP/IP data packets from other SA/DA.

8.4.3 Duplicate packet detection

Packets that are detected as being duplicates shall be discarded by the receiver. This can be accomplished using the same algorithm described in the previous subclause to insure the sequencing.

8.4.4 Stale packet detection

CNP/IP devices shall be capable of detecting stale packets. A CNP/IP device shall support the ability to turn off stale packet detection for those situations where it is not necessary. An example of this might be single segment Ethernet LANs where there are not intervening IP routers to cause unknown delays in the network traffic. Assuming stale packet detection is enabled, packets that are detected as being stale shall be discarded by the receiver. A packet is considered to the stale if the time it takes for the packet to be transported from the sender to a receiver in an IP channel is greater than the channel timeout period (CTP). The CTP is a period of time that represents a reasonable upper bound on the time it takes a packet to go from a sender to a receiver on the IP channel. This International Standard does not cover how the channel timeout period is determined. Suffice it to say that it has units of milliseconds and is known by each of the CNP/IP devices in the IP channel. There is a single timeout period that is applied to every CNP/IP device on the channel.

In order for devices to be able to detect when a packet is stale it is necessary to determine how much time has expired since the packet was transmitted onto the IP channel. The method specified here is for the transmitter to assign a time stamp to each data packet before it is forwarded onto the IP channel. In the case of CNP/IP to CNP/IP routers the packets are re-stamped with the most recent time before being forwarded onto the next IP channel. In the case of proxy nodes that are forwarding packets onto the same IP channel the packets are not re-stamped. Also note that all CNP/IP addresses that receive a specific packet shall have the same time stamp.

In order for the time stamp to be effective the clocks on all the devices on the IP channel shall be synchronised. The precise accuracy of this synchronisation is not specified. Depending upon the type of IP network being used a wide range of accuracies are possible. It should be noted that any inaccuracies of the time synchronisation will manifest itself in the form of a transport delay jitter for the protocol being tunnelled. The extent to which this matters for a specific protocol is covered in the appropriate annex of this specification.

The means for synchronising the clocks of all the CNP/IP devices is SNTP as specified in RFC 2030. This implies that there is an SNTP time server somewhere in the IP network and that the CNP/IP devices have been configured to access its IP address. The format of the time stamp specified in RFC 1305 is a 32-bit integer value of seconds and a 32-bit integer value of picoseconds. There is no contiguous range of bits from this format that has the required resolution and arithmetic properties. Therefore, the timestamp format within this specification is 32 bits of milliseconds. It is aligned with the current SNTP time. This timestamp wraps around every 49,7 days.

As long as communications are maintained with the time server then the devices should remain in sync and stale packet detection can occur normally. Problems can occur if communications with the time server are lost for whatever reason. Under these conditions the device may continue forwarding packets only as long as it can be reasonable sure that its clock has not drifted out of range with the other devices on the network. Since the time server is the common basis for time on the network the device may continue forwarding packets only as long as it is reasonably sure that it is within the margin of error of the time server before it went off line. This can be accomplished if the device knows what its own clock drift rate is. If the device has no way of estimating the margin of error between its own clock and that of other devices on the network then it needs not forward packets onto the IP channel.

While the resolution of the NTP time is in picoseconds, the practical precision of a time produced in a system is in terms of timer interrupts that are typically 10 ms or 16 ms. The precision of the timestamp is milliseconds, but of course the low (3 or 4) bits are of no value. This means that rounding or truncating is not an issue. Also, observe that the smallest receive transaction timer value for CNP is 128 ms, so that small values of the difference between two timestamps are not relevant.

8.5 Configuration server interactions

8.5.1 General device interaction

8.5.1.1 General

The dialog between a CNP/IP device (client) and a configuration server works as follows:

a) a device sends a Registration message to the configuration server. Included in this message are some of the configuration parameters. This message is repeated until a response is received from the server. There is an increase at an exponential rate for the repeat interval with a maximum interval of 30 s. If a device has a valid configuration it may use it to route messages while it waits for a response from the server;

b) the server shall answer with one of two possible messages. An Acknowledge message with a value of ACK_DEVICE_REFUSED if it does not want to add the client to the channel, or a Device Configuration message that includes some of the configuration parameters. The server does not retransmit the Acknowledge packet. If the device does not receive it, the device will retransmit the original request;

c) the device shall acknowledge that it received the Device Configuration message sent to it in b). The Acknowledge message can have various values; a value of ACK_OK to say that the device accepted the configuration, a value of ACK_FIXED to say that the device has a fixed configuration, a value of ACK_BAD_MESSAGE to say that the received message was corrupted or a value of ACK_CANT_COMPLY to say that, for whatever reason, the device could not use the configuration parameters;

d) after the registration process the device may then send a series of request messages to the server;

e) the server shall answer these requests with the appropriate response messages.

How the server responds to different acknowledgements is not defined here, but the server should log the messages to allow for debugging.

8.5.1.2 Unsolicited packets from the server

The server can send an unsolicited Device Configuration message to a device. The device acknowledges it in the same way described above. It is best practice for the server to avoid flooding a device that is coming on-line with unsolicited packets until the device has finished requesting packets. Implementation details are a matter of choice. Other configuration packets are never sent in an unsolicited manner. The Device Configuration message sent in an unsolicited manner is never segmented.

The unsolicited Device Configuration message from the server is used to indicate to the device that some other part of the channel configuration has changed. Datetime fields in the Device Configuration message are valid in these messages and indicate the datetimes of the latest Send List and Channel Membership list packets. After comparing these datetimes with the datetimes of the packets currently stored by a device, the device shall request Channel Membership, Send List and Channel Routing packets from the server in order to get the most recent versions.

The configuration of a CNP/IP channel can be static even though a configuration server is being used. Such would be the case if the configuration was known a priori and fixed in the configuration server. This is one of the reasons a server may answer a client registration message with a CNP/IP device refused message. The device registering with the server may not be part of the fixed configuration. The way in which the server determines how to respond to a client registration message is vendor specific.

8.5.1.3 Requests from devices or other nodes

Devices shall answer requests from other nodes, possibly devices. No guarantee, beyond those of segmentation, is made for the data requested by another device or node that is not a device or server. By segmentation guarantees is meant that the series of packets will have the same datetime and a consistent set of bytes. A device shall maintain global integrity with respect to its communication with the server.

For example, if a device (A) responds to another device (B) for a channel routing packet, and during this exchange, the device routing information of (A) changes, the device (A) has no responsibility to inform the device (B), to fail the responses to (B) or to inform the device (B) that the data has changed by changing the datetime (unless the remainder of the data during this exchange is from the changed data set). The only integrity responsibility that the device (A) has is to inform the server of the updated information.

Devices have no responsibility to respond to, or to process unsolicited configuration messages from devices other than the server. They shall respond to unsolicited request messages from devices other than the server. This means that if a device receives an unsolicited Channel Membership, Send List, Channel Routing or Device Configuration message from a node that is not the server, it may drop it on the floor.

8.5.1.4 Datetime

All configuration packets contain a datetime. This is the date and time for which the data is valid. Newer or older versions of this data may be determined with a resolution of 1 s by looking at this field. This field is constrained so that if more than one version of the data is created in the same second, that they each have unique and increasing values for this field.

If the network supports SNTP or NTP, the value of datetime is the seconds portion of the NTP date time from RFC 1305. This is the number of seconds since 0h 1 January 1900. This time will expire in 2036. See RFC 2030 Clause 3 for more details.

If a network does not support the SNTP or NTP protocol, the datetime may be a small integer, not including zero. These dates, in the early 1900's, are clearly not wall clock time, but are constrained to obey the same requirement of uniqueness for all time as mentioned above. That is, these values, as emitted by such a device, never wrap around or repeat for different data.

The datetime is expressed in UTC [Co-ordinated Universal Time], or equivalently GMT, so that the timezone does not affect the value. Also daylight saving time is not adjusted. Since UTC is used, if a device moves between timezones, or if daylight savings time comes into affect, the times reported for configuration packets are always monotonically increasing.

As clocks drift and clocks are reset, automatically or manually, the time set in the device may change forward or backward. A device shall never emit a configuration packet with a datetime earlier than that for any packet it ever emitted. This can be accomplished in several ways. One way is to have the device save, in non-volatile memory, the datetime of the last configuration packet that it emitted. Each time a configuration packet is to be emitted, then the device chooses the latest of the current datetime, and the last saved datetime + 1. If the device emits new configuration packets at an average rate of less than one per second, the configuration datetime and the system datetime will eventually cross and the system datetime will be later.

8.5.2 General protocol interaction

In general, each packet sent has an expected response. The response may be either an ACK or another message in a protocol exchange. Receipt of the other message shall be interpreted as an ACK_OK for the previous message.

- Packets are re-transmitted a few times with a back-off algorithm to avoid congestion. For example, the retransmit timer might be set to 1 s, and each time it might double until 30 s is reached.
- ACKs are not retransmitted, but the state change in a device is always such that receipt of a duplicate previous message in the protocol causes no state change and causes retransmission of the appropriate ACK.

A configuration server may support multiple channels. In this case, the configuration server is configured with a list of device IP addresses for each channel. Device Registration messages contain the IP address of the device, so that channel membership of the device can be determined. For all other messages, the sending IP address can be determined as the source address of the UDP frame, or the source address of the TCP connection.

8.5.3 Packet Segmentation

8.5.3.1 UDP

The UDP payload length of approximately 548 bytes causes restrictions in the membership of channels and causes other problems when nodes are considered. Dealing with large packets can be confined to the configuration protocols and are of primary concern for these packet types:

- Channel Membership packets for populations larger than about 128 devices;
- Send List packets for populations larger than about 64 devices;

— Channel Routing information for nodes with more than about 4 Domains.

When these data sets exceed the 548-byte limitation of the UDP packet size then the segmentation scheme described in 8.3 is used.

8.5.3.2 TCP

TCP is an optional means of communicating with a server. When the IP_PROTOCOL field of the Device Registration packet indicates TCP or BOTH TCP AND UDP, then the server may use a TCP link to exchange data with the device.

In this case:

a) segment packets are never used. Packets, regardless of size, are sent in their entirety;

b) ACK messages are not sent on TCP connections;

c) retransmissions are not performed on TCP connections;

d) requests can be sent from a device to the server using either TCP or UDP. Such requests may be serviced by the server using either TCP or UDP. If the device connects to the server using TCP, and the server supports TCP, then the server shall answer using TCP;

e) when the configuration server detects a configuration change, it may send the configuration data unsolicited, over the TCP link to the device. The Channel Membership packet is always sent first, followed by any relevant updated packets. The following packets may be sent in this way: Send List, Device Configuration or Channel Routing packets;

f) when a TCP connection breaks in a non-graceful way, the device, not the server, is responsible for re-establishing the connection. The assumption is that the device has failed or become partitioned from the network. When a new update occurs, the configuration server tries again, once, to establish a connection with the device to forward the update. The device should try continuously, with some prudent implementation dependent period, to establish a connection with the server and should continue to route messages based on any existing configuration data. When TCP is used, either the device or server may initiate the connection. The connection remains open for a prudent, implementation dependent, period of time awaiting its use by further traffic. Either side may terminate the connection after its own timeout. Graceful TCP connection shutdown is encouraged, but not required.

When a device that supports TCP begins its first communication with the server, it does not know whether the server supports TCP or not. The device may use the following mechanism to determine whether to use TCP.

i) The device attempts a TCP connection with the configuration server.

ii) The device need not wait for the connection to succeed or timeout before sending requests via UDP to the server. For example, the device might start the TCP connection and then send the Device Registration packet to the server immediately. Note that the Device Registration packet contains the IP Protocol field that states that the device supports TCP.

The only way for the device to determine whether the configuration server supports TCP is to have a connection succeed. This mechanism assumes that a node that does not support TCP cannot respond with *Connection Refused.*

If the number of simultaneous connections supported by a server is smaller than the number of devices on the channel, devices might receive connection refused messages from the server when trying to connect. Such responses should indicate that there are insufficient resources on the server. In this case they try again after a suitable amount of time, or elect to use UDP.

Servers should accelerate timeouts of cached TCP connections so as to provide one or more open connections for new device conversations.

8.5.4 Device Registration

This interaction is designed to allow a device to inform the Configuration Server of its identity and to obtain IP operational parameters from the server.

The device requires the following information to begin interacting with the server. All other information can be obtained from a server based on the uniqueness of this data:

a) IP address/port of the device;

b) IP address/port of the configuration server.

There are several cases to be solved with this interaction:

i) the device has some identity information, and needs to obtain the remainder from the server. The server knows about the device;

ii) the device has a fixed, possibly locally entered, configuration and needs to register with the server, and obtain a send list and or channel membership list;

iii) the device may or may not be able to comply with configuration information *suggested* by the server. The protocol does not allow for *negotiation* of configuration. If the device is unable to comply with the *suggested* configuration from the server, it reports this, and ceases operation;

iv) when a device *ceases operation* it continues to process messages from the configuration server, but does not route messages. It does not instigate further activity with the server, but processes Device Response messages.

This protocol is depicted in Table 1.

Table 1 —Device Registration with Configuration Server Protocol

Device	←→	**Server**
Send device registration packet	→	(1) Add to configure or reject
Cease operation until reset.	←	ACK_DEVICE_REFUSED on reject or
Begin operation with new parameters, and	←	Device configuration packet
ACK_OK begin operation or	→	Stop retransmit of device configuration, indicate in database that device is a member.
ACK_CANT_COMPLY cease operation until reset.	→	No further action, no database change.
Perform next step.	←	Send unsolicited device configuration packet.
Send ACK_OK or ACK_CANT_COMPLY	→	Request further information if necessary based on datetimes in device response

In the solicited case, the server does not retransmit the Device Configuration packet. If the client doesn't receive the Device Configuration Packet in time it will retransmit the Device Registration Packet to the server.

In the unsolicited case the server should retransmit the Device Configuration packet if it does not receive an ACK_OK or an ACK_CANT_COMPLY from the client.

A Configuration Server may inform the members of a channel of a change of configuration server through the use of these messages as well. This would be required if the configuration server's IP address or port changed. This protocol is depicted in Table 2. At present this is not a secure operation. Devices may refuse this request, and so shall be configured in some implementation dependent manner for the identity of the Configuration manager. This protocol, in and of itself, allows no significant additional security compromise, however, since IP spoofing is possible, and so another node could usurp the IP identity and role of the Configuration manager whenever it is off the net.

Table 2 — Server to Device Unsolicited Configuration Message Protocol

Device	←→	Server
Build response	←	Send Configuration Request packet
Send Device Configuration packet	→	Build Device Configuration packet
	←	Send Device Configuration packet.
Adopt new IP address / port for Configuration Server and Time Server Send ACK_OK	→	Proceed normally as Configuration server
Send ACK_CANT_COMPLY. Cease operation so as not to interfere with further operations of channel.	→	Mark Device as no longer a member of the channel.

If the device adopts a policy so that it does not cease operation when it receives a change to the server that it cannot comply with, or refuses to comply with, then the device is only interoperable with like devices. This is because a new configuration manager shall rely on all devices either obeying its commands and information, or shall rely on them failing until they are manually updated in some other way.

The server should request the client's configuration before sending a new Device Configuration packet. It is not a requirement that the client receive such a request before processing a new Device Configuration packet from the server.

8.5.5 Channel Membership

The purpose of this interaction is to allow device to obtain a complete list of other devices on the channel. This protocol is depicted in Table 3.

If the channel membership list changes after some devices are active, the server can send an unsolicited Device Configuration packet with a new datetime for the Channel Membership packet.

Table 3 — Device to Server Channel Membership Request Protocol

Device	←→	Server
Send Channel Membership Request packet	→	Find device in database and identify channel.
Cease operation until reset.	←	ACK_DEVICE_REFUSED if no channel defined or device is not a member of any channel or
Begin operation with new parameters.	←	Send Channel Membership packet

The Channel Membership packet is not retransmitted by the server. If the device does not receive the packet, it requests again.

When any change occurs to the configuration, the server notifies each of the members of the change. There are changes of three kinds:

a) a device is added to the list;

b) a device is removed from the list;

c) a device changes its configuration requiring other nodes to update their Send List or Channel Routing information for that node.

These cases are all handled by the server sending all devices an unsolicited Device Configuration packet with an updated datetime for the Channel Membership packet. The device then requests a new Channel Membership packet. If the node receiving the packet finds it is not in the membership, it ceases routing packets and awaits further configuration messages. If the node does not have a channel membership list, meaning it has just been added to the channel, then it requests Channel Routing packets or Send List packet.

Each entry in the Channel Membership packet contains a datetime which indicates the last valid datetime of the Channel Routing packet. Also it contains a SendListDateTime which indicates the last valid time of a send list packet and a DeviceRequestDateTime indicating the last valid time of that packet for the device. These datetimes are used by the device to request new information.

8.5.6 Send List

This interaction allows device to obtain a send list for the channel. This protocol is depicted in Table 4.

If the send list changes after some devices are active, the protocol can begin with the server sending a Device Configuration packet as an unsolicited message (see 8.5.4).

The send list is an optional means by which routers may forward packets. By the use of send lists, a configuration server may centrally manage multi-cast addresses for a channel. Servers shall support the creation of Send List packets and shall respond to requests by devices for these packets. Support of Send List packets by devices is optional and devices need not request these packets from servers.

How a Send List is derived or configured into the server is implementation dependent as long as it adheres to the three rules specified in Clause 6.1. In the absence of a configured Send List or a more sophisticated algorithm for deriving it, the server may use the following algorithm to generate a Send List.

Send List Generation Method

If and only if every Device in the CNP/IP channel belongs to the same multi-cast group(s) use Algorithm A, else use Algorithm B. Note that which multi-cast groups a device belongs to are part of the device's configuration which the server has full knowledge of. This knowledge is gained either through the Device Registration packet sent from the device or the Device Configuration packet sent from the server to the device.

Algorithm A (Highly Optimized)

Select a single multi-cast group that every device on the CNP/IP channel belongs to. Use the multi-cast IP address for this group as the one and only entry in the Send List. Note that this algorithm depends upon every device in the CNP/IP channel belonging to the multi-cast group specified in the Send List. If this becomes untrue for whatever reason then the Send List shall be modified accordingly.

Algorithm B (Brute Force)

For each device in the Channel Membership list add the corresponding unicast IP address for that device to the Send List. In this scenario the Send List shall maintain a one to one correspondence with the Channel Membership list. If Devices are added or deleted from the Channel Membership list the Send List shall be modified accordingly.

When using send lists, every packet forwarded by a device is sent to every address in the list, unconditionally. The device listens on ports and addresses specified in the Device Configuration messages.

Table 4 — Device to Server Send List Request Protocol

Device	←→	Server
Send Send List Request packet	→	Find device in database and identify send list.
Cease operation until reset.	←	ACK_DEVICE_REFUSED if no channel defined or device is not a member of any channel or
Begin operation with new parameters	←	Send Send List packet

The Send List packet is not retransmitted by the server. If the device does not receive the packet, it requests again.

8.5.7 Channel Routing

8.5.7.1 General

This interaction allows device to obtain Channel Routing information for the device or to allow the device to send the Channel Routing information to the server for distribution to other devices. This protocol is depicted in Table 5 and Table 6.

These packets are exchanged between CNP/IP devices and represent the routing information for the device indicated in the message. When a configuration server sends several of these packets, each unicast IP Address corresponds to an IP address in the channel membership list. The routing information refers to that node. All IP addresses of channel members shall be unique.

Support of Channel Routing packets is optional.

Specifically the routing tables which consist of subnet and group masks are sent in this packet. The subnet and group mask information may be used by routers to avoid sending every packet to every other router which might otherwise be wasteful of bandwidth if, for some reason, multi-cast addressing between the routers is not allowed on the IP network.

The SubnetMsk, GroupMsk, and Domain structure may be repeated in the packet if the router can route to a more than one CNP domain or as a proxy.

If the channel routing information changes after some devices are active, the protocol can begin with the server sending an unsolicited Device Configuration message (see 8.5.4).

When a device supports Channel Routing packets, and it has new channel routing information, the device creates a new Channel Routing packet. This packet is then transmitted to the server.

Table 5 — Device to Server Channel Routing Update Protocol

Device	←→	Server
Send Channel Routing packet	→	Find device in database and identify routing list.
Cease operation until reset.	←	ACK_DEVICE_REFUSED if no channel defined or device is not a member of any channel or
Stop re-transmission of Channel Routing Packet.	←	Send ACK_OK.

When a device supports Channel Routing packets, it requires the Channel routing packets of the other devices on the channel. It obtains these packets in the manner shown in Table 6.

Table 6 — 6 Device to Server Channel Routing Request Protocol

Device	←→	Server
Send Channel Routing Request packet	→	Find device in database and identify routing list.
Cease operation until reset.	←	ACK_DEVICE_REFUSED if no channel defined or device is not a member of any channel or
Begin operation with new parameters.	←	Send Channel Routing packet.

The transmission by the server is not repeated and no ACK is performed by the device. If the device does not receive the message, it requests again.

The Channel Routing Request message contains two fields of interest in the request to the server:

Datetime	indicates that data should be sent if any data is newer than this datetime. Zero indicates always send the data;
IP Unicast Address	if non-zero, indicates that only the channel routing for this device is to be sent. If zero, indicates to send all channel routing information for all members of the channel.

These options may be used to optimise requests to the server. For example, if a router requests a channel membership list and finds a new device in the list, it might request only the channel routing information for that device.

8.5.7.2 Routing to subnet/node addresses

The Channel Routing packet contains a list of subnet/node addresses, as well as a list of domains with subnet masks. These may be used by a router to perform optimal routing in the following manner:

Subnet/node messages addressed to a matching domain/subnet/node address go to the corresponding device. Matching subnet broadcasts and subnet directed Unique ID messages would go to all nodes configured in that subnet. Such subnets would not be set in the subnet mask of the domain. Domain/subnet/node addresses will typically be unique but in some cases (certain error cases or transitory conditions) there may be multiple devices claiming the same address. In such cases, messages should be routed to the multiple devices.

8.5.7.3 Semantics for Wants All Broadcasts

The CNP Flags field of the Channel Routing packet contains a bit called WANTS_ALL_BROADCASTS. This field conditions optimised routing as follows:

All devices will represent at a minimum one CNP addressable node. For example, for CNP/IP Routers, this node is the router side connected to the IP channel. If a device has at least one CNP addressable node, which is in the un-configured state, then it shall report that it needs all broadcasts (domain-wide or subnet), regardless of the domain ID in the broadcast message. This is because CNP nodes respond to all broadcasts regardless of the domain ID when in the un-configured state.

8.6 Miscellaneous Status Messages

8.6.1 General

These messages are intended to allow network tools to extract information from a device to aid in the management and debugging of devices on a network. In general all available information about a devices health, statistics and configuration can be obtained by messages described in this subclause.

They are not intended to be used for the configuration of the devices or to replace functions already defined for the Configuration Server.

The following general operations are supported:

— Request general health/status of CNP/IP device.
— Request general health/status/statistics of CNP/IP device and have the statistics cleared upon sending them to requestor.
— Request configuration of a device.
— Request send list of a device.
— Request channel membership of a device.
— Request channel routing information of a device.

All these operations are in the form of a simple request/response transaction. When appropriate, message formats that have already been defined for the Client/Server interactions are re-used here. In any of the request/response transactions described in this subclause it is important to note that a device which is not the Configuration Server, should not respond to requests which do not correspond to the device being requested. For example device A should not respond to a request for information about device B unless device A is a configuration server.

8.6.2 CNP/IP Device Status

8.6.2.1 General

All this information is optional. The means to supply this information is not dictated in any way. Possibilities for access to this information include, but are not limited to:

— local serial line on the router
— CNP parameter access;
— CNP network variable access;
— private (as yet to be defined) messages over IP;
— HTML server running in the router.

These data are not intended to duplicate or interfere with any optional support for SNMP MIB-II in the TCP/IP stack of the router.

8.6.2.2 Status Information

In general, the statistical information is supplied as unsigned 32 bit fixed point integer format. If a statistic is not supported, the value of that statistic is set to 0xFFFFFFFF. If a statistic is in the overflow state, then its value is set to 0xFFFFFFFE. Support of any of the individual statistics is optional, however, any CNP/IP node shall respond with this packet to a request for the information. Vendors are free to add implementation specific statistics as an additional response packet. See 8.7 for information on Vendor Specific message extensions.

1) Time since last Counter Reset. (format: 32 bit unsigned integer number of seconds)

2) Time of last Counter Reset. (In GMT.) (format: datetime, see 9.4)

3) Number of members on the channel. (format: 32 bit unsigned integer)

4) Number of members on the channel to which messages were sent in the recent past. (Method of measurement is left undefined.). (format: 32 bit unsigned integer)

5) CNP packets received. (Packets received from CNP channel.). (format: 32 bit unsigned integer)

6) CNP packets received but discarded due to selective forwarding. (format: 32 bit unsigned integer)

7) CNP total bytes received. (format: 32 bit unsigned integer)

8) CNP packets sent. (Packets sent onto CNP channel). (format: 32 bit unsigned integer)

9) CNP total bytes sent. (format: 32 bit unsigned integer)

10) CNP packets sent onto IP channel. (format: 32 bit unsigned integer)

11) CNP bytes sent onto IP channel. (format: 32 bit unsigned integer)

12) CNP packets received from IP channel. (format: 32 bit unsigned integer)

13) CNP bytes received from IP channel. (format: 32 bit unsigned integer number)

14) IP packets containing LT packets onto IP network. (format: 32 bit unsigned integer number)

15) IP packets containing LT packets from IP network. (format: 32 bit unsigned integer number)

16) Average aggregation onto IP channel. (Value is expressed as a pair of 32 bit unsigned integers. Total LT packets <10> / Total IP packets <14>.)

17) Average aggregation from IP channel. (Value is expressed as a pair of 32 bit unsigned integers. Total LT packets <12> / Total IP packets <15>.) (Since all these parameters are optional, duplicate data forms is not necessarily a problem. One might elect not to support 12, 15, but might support 17 instead.)

18) Number of UDP packets sent. (format: 32 bit unsigned integer)

19) Number of TCP packets sent. (format: 32 bit unsigned integer)

20) Number of Multi-cast packets sent. (format: 32 bit unsigned integer)

21) Stale LT packets from IP dropped. (Stale means that the have been too long in the IP network). (format: 32 bit unsigned integer)

22) Count of TCP Connection failures. (format: 32 bit unsigned integer)

23) Number of different hosts experiencing TCP Connection failures. (format: 32 bit unsigned integer)

24) Number of router configuration messages sent. (format: 32 bit unsigned integer)

25) Number of router configuration messages received. (format: 32 bit unsigned integer)

26) Number of configuration changes. (format: 32 bit integer)

27) Running average number of UDP packets/second sent. (format: 32 bit integer)

28) Running average number of UDP packets/second received. (format: 32 bit integer)

29) Running average number of TCP packets/second sent. (format: 32 bit integer)

30) Running average number of TCP packets/second received. (format: 32 bit integer)

8.6.3 Device Configuration

A Device Configuration may be requested by some host. Such a request shall be responded to if it specifies the device receiving the request.

Device Configuration request/response transactions use the same packets as described in 8.5.4, but use a simpler interaction as shown in Table 7.

Table 7 — Protocol for Requesting a Device's Configuration

IP Host	←→	Device
Send Configuration Request packet	→	Confirm Request Valid
DONE	←	ACK_DEVICE_REFUSED on reject or
DONE	←	Send Device Configuration Packet

8.6.4 Device Send List

A device's send list may be requested by some host. Such a request shall be responded to if it specifies the device receiving the request.

Device Send List request/response transactions use the same packets as described in 8.5.6, but use a simpler interaction as shown in Table 8.

Table 8 — Protocol for Requesting a Device's Send List

IP Host	←→	Device
Send Send List *Request* packet	→	Confirm Request Valid
DONE	←	ACK_DEVICE_REFUSED on reject or
DONE	←	Send Send List packet

8.6.5 Channel Membership List

A device's channel membership list may be requested by some host. Such a request shall be responded to if it specifies the device receiving the request.

Device Channel Membership List request/response transactions use the same packets as described in 8.5.5, but use a simpler protocol interaction as shown in Table 9.

Table 9 — Protocol for Requesting a Device's Channel Definition

IP Host	←→	Device
Send Channel Membership Request packet	→	Confirm Request Valid
DONE	←	ACK_DEVICE_REFUSED on reject or
DONE	←	Send Channel Membership packet

8.6.6 Channel routing information

A device's channel routing information may be requested by some host. Such a request shall be responded to if it specifies the device receiving the request.

Device Configuration request/response transactions use the same packets as described in 8.5.7, but use a simpler interaction as shown in Table 10.

Table 10 — Protocol for Requesting a Device's Channel Routing Information

IP Host	←→	**Device**
Send Channel Routing Request packet	→	Confirm Request Valid
DONE	←	ACK_DEVICE_REFUSED on reject or
DONE	←	Send Channel Routing packet

8.7 Vendor Specific Messages

The Vendor Code in the Common Packet Header allows for vendor-specific packets. This value shall be set to 0 for all packets of standard definition according to this specification. Vendor-specific packets are identified in part by a unique Vendor Code (other than 0).

A unique Packet Type code is assigned to each function as detailed in 9.1. Standard Packet Type codes that are defined within this specification are in the range 0x00 to 0x7F. Vendor-specific packets that convey information as an extension to a standard function defined in this specification may use the same Packet Type code as that standard function, however the Vendor Code shall be set to the unique identifier for that vendor. Vendor-specific packets that have no relationship to existing standard functions defined in this specification shall use a Packet Type code in the range 0x80 to 0xFF.

Vendor Specific packets received by devices that do not understand that Vendor Code/Packet Type combination shall be discarded.

8.8 Authentication of CNP Packets

Security in CNP/IP devices is optional. If the security bit in the protocols flags field of the CNP/IP header is set (see 9.2 Common CNP/IP Header) then the authentication scheme below shall be implemented as described in this subclause.

The level of security described in this subclause is authentication. Messages sent using the scheme described here are authenticated as coming from a trusted source. Information inside the messages is not encrypted and not hidden from inspection. This scheme ensures that the sender formed the packet using a valid shared secret and that the packet has not been tampered with since sending. For more information on this scheme, the reader is referred to the "MD5 Message-Digest Algorithm", see RFC 1321.

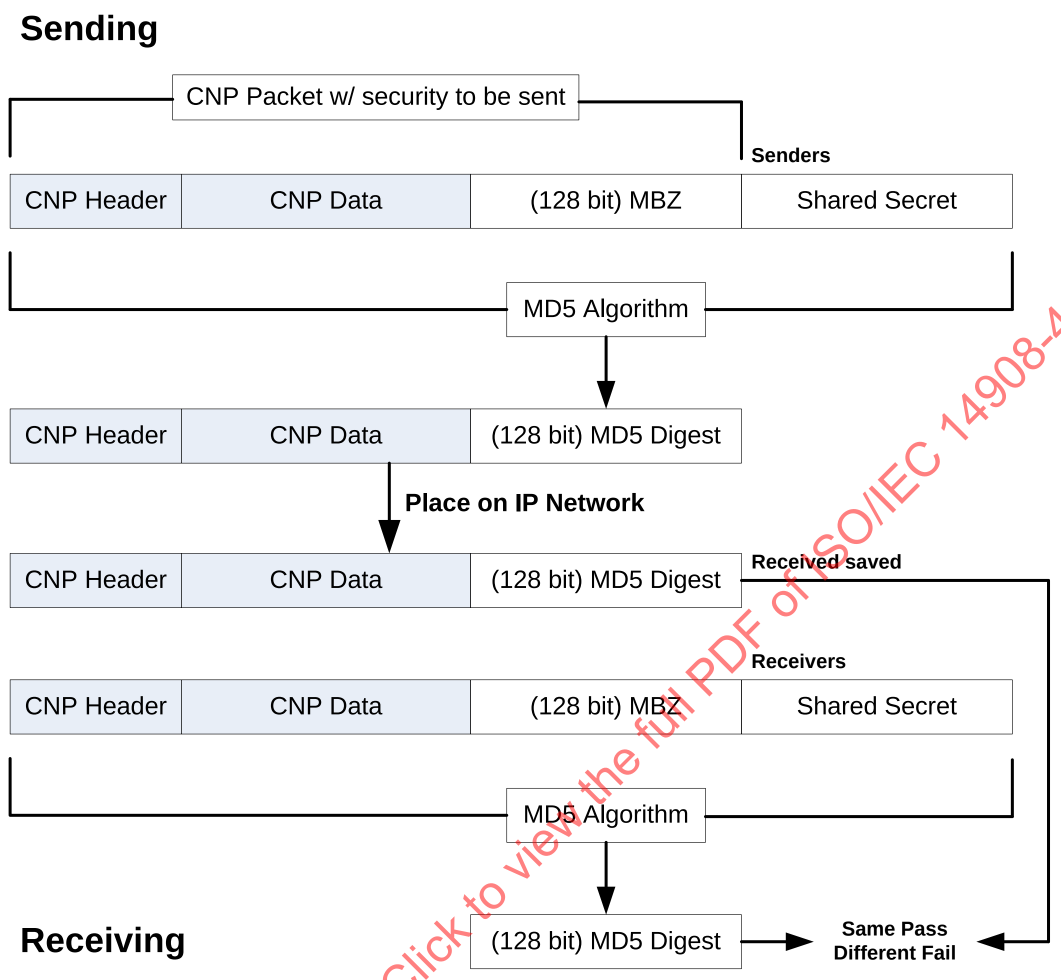

Figure 4 — Authentication encoding and decoding of CNP packets

Figure 4 illustrates the procedure for encoding and decoding packets for which the security bit in the CNP header is set.

Before transmitting a secured message the sender appends a shared secret to the message as shown above. The "Shared Secret" shall at a minimum represent a value which contains 128 degrees of freedom. For example, a 16 byte array where all of the bits are available for selection, or a 22 byte string, where 6 bits per byte are available, would both meet the minimal requirements for a CNP "Shared Secret". The CNP "Shared Secret", however, is not limited to 128 degrees of freedom, and may be larger.

Other requirements of the “Shared Secret” are:

— the “Shared Secret” should never be transmitted openly on the network;

— CNP devices shall allow for a manual secure means of setting and updating device secrets;

— a CNP device shall provide for a least a single “Shared Secret” which can be used to validate requests, if and only if authentication is present in the device implementation;

— optionally manufactures may initially seed products with a default “Shared Secret” that may be updated later using secure network messages.

Next, a Digest of the packet is generated by applying the MD5 Message-Digest Algorithm(see RFC 1321, to the complete CNP/IP packet plus the “Shared Secret”. The generated MD5 Digest is then inserted into the MBZ key field above and the packet minus the Secret is transmitted to the network.

A receiving device performs the same procedure on the incoming packet.

a) Stores off the incoming Digest and replaces the Digest field with zero's.

b) Performs a digest of the packet with its own “Shared Secret”.

c) Compares the incoming Digest to the new Digest it produces.

d) Matching MD5 Digests denotes valid packets.

e) Non-authenticated packets should be discarded and may be noted by the CNP device.

9 Packet formats

9.1 Packet Types

This clause gives a detailed description of all CNP/IP packet contents and formats. The reader is referred to Clause 6 for how the various packets described in this subclause are used.

All CNP/IP packets are composed of a header followed by a data portion. The header format is identical for all CNP/IP messages. The data portion is dependent upon the type of CNP/IP packet.

All fields are in network byte order except where noted.

Table 11 is a cross-reference for all the packets used in the system.

The following acronyms are used to depict various data sets:

DC – Device Configuration;

CM – Channel Membership list;

SL – Send List;

CR – Channel Routing.

Table 11 — Message type cross reference

Packet Name	Format (Clause 9)	Transactions used (Clause 8)	Code
Data packet	9.4	CNP data packet exchange (8.4)	0x01
Device configuration request	9.8	DC request to server (8.5.4) DC request to Device (8.6.3)	0x63
Device registration	9.5	Device registration with server (8.5.4)	0x03
Device configuration	9.5	DC response from server (8.5.4) DC response from device (8.6.3) Server's unsolicited update of Device's Configuration (8.5.1.2)	0x71
Channel membership request	9.8	CM request to server (8.5.5) CM request to device (8.6.5)	0x64
Channel membership	9.6	CM response from server (8.5.5) CM response from Device (8.6.5)	0x04
Send list request	9.8	SL request to server (8.5.6) SL request to device (8.6.4)	0x66
Send list	9.10	SL response from server (8.5.6) SL response from device (8.6.4)	0x06
Channel routing request	9.8	CR request to server (8.5.7) CR request to device (8.6.6)	0x68
Channel routing	9.7	CR response from server (8.5.7) CR response from device (8.6.6)	0x08
Acknowledge	9.9	8.5.2	0x07
Segment	9.3	All CM, SL, and CR transactions in 8.5 and 8.6.	0x7f
Status/health/statistics request	9.8	Status request to device (8.6.2)	0x60
Status/health/statistics response	9.11	Status response from device (8.6.2)	0x70

9.2 Common CNP/IP Header

All CNP/IP packets have the common header shown in Table 12.

Table 12 —Common Packet Header format

Byte 0	Byte 1	Byte 2	Byte 3
Data Packet length		Version	Packet Type
Extended Header Size	Protocol Flags	Vendor Code (16 bits)	
Session ID			
Sequence Number			
Time Stamp			

This header is present on all packets. The following is a description of each field.

Version
This is the version number of the packet. It is assumed that packets with the same version number can always be parsed. If a node receives a header with a version number that it does not understand, it discards the packet without further processing of the packet contents. The current version number is 0x01. The version number is the lower 5 bits of this byte. The upper 3 bits are defined as follows:

Bit 5-6:	Reserved, MBZ
Bit 7:	Vendor Specific Packet Follows; a packet follows this one that contains a vendor code other and is meant to be processed with this packet.

Protocol Flags
These are bit flags are used to depict various aspects of the tunnelling protocol. The following bits are used:

Bit 7:	Reserved, MBZ
Bit 6:	Reserved, MBZ
Bit 5:	Security bit; if this bit is set to 1 then this packet contains authentication information as described in 8.8.
Bits 4–0:	Protocol code; this field specifies the protocol being tunnelled within the packet. The following protocols are supported: 0x00 – EIA-709 0x01 – EIA- 600 0x02 – CNP

Vendor Code
This field is used to indicate vendor specific codes. A value of 0x0000 indicates that this packet is a compliant packet as specified in this document. Any value other than 0x0000 indicates that this is a vendor specific packet. See 8.7 for more information on Vendor Specific messages.

Each vendor code will be assigned and administered by the appropriate industry council for each. Vendor codes are only required to be unique within a specific protocol.

Packet Type
This determines the type of the packet. Packet Type codes for packets defined in this specification are in the range of 0x00 to 0x7F. Vendor specific packets that convey information as an extension to a standard function defined in this specification may use the same Packet Type code as that standard function, however the Vendor Code shall be set to the unique identifier for that vendor. Vendor-specific packets that have no relationship to existing standard functions defined in this specification shall use a Packet Type code in the range 0x80 to 0xFF. See 8.7 for more information on Vendor Specific messages.

Data Packet Length
This field is length of the packet in bytes, including the bytes in this common packet header. The value of this field added to the address of Version is the offset of the next Version field of the following packet, if there is more than one in the current frame.

Extended Header Size
This field is the length of the packet header in numbers of four byte records beyond the standard header size. If there are no extensions to the common header then this field is 0. This also implies that headers shall be multiples of four bytes and aligned on four byte boundaries. This field allows for future revision of this protocol but allowing backward compatibility. All packets generated that adhere to version 1 of this protocol shall have Header Size set to 0.

Session ID
The session ID works in concert with the sequence number to guarantee that packets are received in the order they are sent over a standard UDP channel. Every CNP/IP node keeps a session ID chosen at random. If the CNP/IP node re-boots, resets, etc. and forgets the sequence numbers it was using, the node sends its next messages to each destination with a new session ID. The new session ID is also chosen at random with the constraint that it is not identical to the session ID in use by the node prior to the re-boot/reset event.

When a node receives a message from a source with a different session ID than was in use on the previous message from that source, the node assumes the accompanying message is in sequence. Additionally the receiver remembers the session ID/sequence number pair contained in the new message for the order preservation of the subsequent messages.

Sequence Number
Sequence number for the current packet. See 8.4.2 to 8.4.4 for algorithm details.

Timestamp
The timestamp for the current packet, or zero if no timestamp, is assigned to the packet. This timestamp indicates the time that this packet was originated on this IP channel.

The timestamp field is always zero for configuration packets. It is only valid for data packets.

9.3 Segment Packet

This packet is used to encapsulate other CNP/IP messages that exceed a single UDP datagram size.

The segment packet has the format shown in Table 13.

Table 13 — Segment Packet format

Byte 0	Byte 1	Byte 2	Byte 3
Common Packet Header			
Date Time			
Flags	Request ID		Segment ID
Bytes of payload	Bytes of payload	Bytes of payload	Bytes of payload
Bytes of payload	Bytes of payload	Bytes of payload	Bytes of payload
Bytes of payload	Bytes of payload	Bytes of payload	Bytes of payload
Bytes of payload	Bytes of payload	Bytes of payload	Bytes of payload
Bytes of payload	Bytes of payload	Bytes of payload	Bytes of payload

The following is a description of all fields in this packet.

Packet Type
Refer to 9.1

Date Time
See 9.5 for details of the date time format. This datetime field is copied from the datetime of the encapsulated packet. All the segments for a particular encapsulated packet in which all data elements are coherent have the same datetime. If any of the data making up the encapsulated packet changes during the segmented transfer of that encapsulated packet, then the datetime in the affected segment packets will also change to that of the new data set.

Flags

Bit 7:	Valid; this bit set indicates that valid data exists in this packet representing the indicated Segment ID.
Bit 6:	Final; this bit set indicates that no data exists for Segment ID values greater than that returned in this packet.
Bits 5-0:	Reserved for future use

Request ID
A copy of the Request ID value from the request message that caused this response to be generated.

Segment ID
A copy of the Segment ID value from the request message that caused this response to be generated. This is the identifier of the data that is contained in this response.

9.4 CNP Data Packets

CNP/IP data packets have the format shown in Table 14.

Table 14 — Data Packet format

CNP/IP Common Header	CNP Data Packet

CNP/IP Header
See 9.2.

CNP Data Packet
This is the complete "PDU" packet as described in 6.4 of the CNP standard. The number of bytes making up the PDU is equal to (Packet Size - Header Size).

Note that, as described in 8.4.2, CNP Data Packets shall be processed in the same order as they were originated. Thus, IP packets that have arrived out of order should be reordered before being delivered to a CNP protocol stack. If reordering is not supported, packets received out of order shall be ignored.

9.5 CNP/IP Device Registration/configuration packets

If a configuration server is being used then CNP/IP devices send Device Registration packets to the server to inform them of their existence. The server then responds with a Device Configuration packet. This registration packet will be used by the server to both configure the CNP/IP device and possibly maintain a list of all CNP/IP devices in the IP channel.

The Device Configuration packet is the only packet, which is sent unsolicited by the configuration server to a device.

The CNP/IP Device Registration and Device Configuration packets have the format shown in Table 15.

Table 15 — Device registration/configuration packet format

Byte 0	Byte 1	Byte 2	Byte 3
Common Packet Reader			
Datetime			
IP Flags	CNP Router Type	CNP Flags	Node Type
Mc Address Count [M]	MBZ	Channel Timeout	
Total Unique ID Bytes [U]		IP Unicast Port	
IP Unicast Address			
Channel Membership Datetime			
Send List Datetime			
Config Server IP Address			
Primary Time Aerver IP Address			
Secondary Time Server IP Address			
Config Server IP Port		Primary Time Server IP Port	
Secondary Time Server IP Port		MBZ	
IP Mc Address [0]			
IP Multi-cast Port [0]		MBZ [0]	
IP Mc Address [M-1]			
IP Multi-cast Port [M-1]		MBZ [M-1]	
Unique Id [0]	Unique Id [1]	Unique Id [2]	Unique Id [3]
Unique Id [4]	Unique Id [5]	Unique Id [6]	… [U-1]
Name Length [N]	Name [0]	Name [1]	Name [2]
Name [3]	Name [4]	Name [5]	… [N-1]

NOTE The squared brackets contain the index of a field in an array.

Packet Type
Refer to 9.1.

Datetime
This is the date and time for which the data is valid. Newer or older versions of this data may be determined with a resolution of 1 s by looking at this field. This field is constrained so that if more than one version of the data is created in the same second, that they each have unique and increasing values for this field.

If the network supports SNTP or NTP, the value of datetime is the seconds portion of the NTP date time from RFC 1305. This is the number of seconds since 0h 1 January 1900. This time will expire in 2036. See RFC 2030, Clause 3 for more details.

If a network does not support the SNTP or NTP protocol, the datetime may be a small integer, not including zero. These dates, in the early 1900's, are clearly not wall clock time, but are constrained to obey the same requirement of uniqueness for all time as mentioned above. That is, these values, as emitted by such a device, never wrap around or repeat for different data.

IP Flags
Mask values include:
0x01 - UDP supported
0x02 - TCP supported
0x04 - Multi-cast supported

CNP Router Type
In the case where the *Node Type* of the device is a router, this field signifies the following modes of operation:

0x00:	Configured: For each specified domain, forward subnet/node, subnet broadcasts and subnet directed Unique ID messages (non zero subnet) to the device if the subnet mask bit is set. If the case where the subnet is 0 in any of the above messages, the message should be forwarded as if the subnet 0 bit were set to 1. For group messages, forward if the bit is set in the group mask.
0x01:	Learning: For each specified domain, forward subnet/node, subnet broadcasts and subnet directed Unique ID messages (non zero subnet) to the device if the subnet mask bit is set. In the case where the subnet is 0 in any of the above messages, the message should be forwarded as if the subnet 0 bit in the mask were set to 1. Subnet mask bits are set for those subnets that are on or could be on the "other side" and not set for subnets that are known to be on "this side". For groups messages, always forward (group bits are ignored).
0x02:	Bridge: For each specified domain, forward all messages (subnet and group bits are ignored).
0x03:	Repeater: Forward all messages (domain/subnet/group information ignored).

CNP flags
Mask values include:

0x01:	WANTS_ALL_BROADCASTS. See 8.5.7.3 for a description of this function.
0x02:	Supports security as described in 8.8.

Node Type
Type of the device. The behaviour of some of these types remains unspecified at this time.

0x01:	Non IP channel to IP channel Router.
0x02:	IP channel Node.
0x03:	IP channel Proxy.
0x04:	IP channel to IP channel Router.

McAddress Count
Number of multi-cast IP addresses and ports to follow.

Channel Timeout
The value in milliseconds of the timeout as discussed in 8.4.4 and 8.4.2. Values of

1 ms to 1 500 ms are valid. Although this parameter can be specified down to 1 ms, due to the nature of SNTP this level of precision can not be guaranteed and should not be expected. More practical levels of the minimum precision are in the 10 ms to 16 ms range.

Total Unique ID Bytes
Total number of bytes of unique IDs to follow. Shall be a multiple of 6 bytes - the size of a Unique ID. There shall be at least one entry. If the device is a node the Unique ID corresponds to the unique ID as defined in ISO/IEC 14908-1. A router device may have up to three Unique ID’s, one for each side of the router and one for configuration purposes. The one that is required to be specified in this field is

the one that corresponds to the side of the router that is directly connected to the CNP/IP channel under consideration.

IP Unicast Port
The port number the CNP/IP device uses when listening for unicast IP messages.

IP Unicast Address
The unicast IP address of the CNP/IP device in network byte order. The CNP/IP device will listen for incoming unicast IP packets on this address.

Channel Membership Datetime
The datetime of the most recent Channel Membership packet from the server. This datetime is used to determine whether a new Channel Membership packet should be requested from the configuration server when the device receives an unsolicited device response packet from the server. Content is not meaningful when the packet is sent from the device to the server.

Send List DateTime
The datetime of the most recent Send List packet for this device. This datetime is used to determine whether a new Send List packet should be requested from the configuration server when the device receives an unsolicited device response packet from the server. Content is not meaningful when the packet is sent from the device to the server.

Config Server IP Address
IP Address of the Configuration server. A new configuration server for this channel may send a Device Configuration packet setting the IP address and port of the Configuration server and Time server.

Primary / Secondary Time Server IP Addresses
IP Address of the NTP time servers. These are set by the Configuration server as discussed above. The device uses the primary server if it can be reached, otherwise it uses the secondary server. This assures that the devices use a predictable server.

Config Server IP Port
The IP port number for the configuration server. Set as above.

Primary / Secondary Time Server IP Ports
The IP port numbers for the time servers. The standard NTP port (123) should be used unless extenuating circumstances prohibit this.

IP Multi-cast Address
The multi-cast IP address of the CNP/IP device. The *McAddress Count* is zero if not supported. The CNP/IP device will listen for incoming multi-cast IP packets on this address. The 8 bytes of address, port, MBZ are repeated *McAddress Count* times.

IP Multi-cast Port
The port number the CNP/IP device resides on when listening for multi-cast IP packets.

Unique IDs
Unique IDs are packed together, 6 bytes each, expressed in CNP network byte order.

Name Length
The length of the name field to follow. The Name shall be zero terminated and this count includes the zero.

Name
This is a variable length name up to 128 bytes long exclusive of the zero terminator. It is associated with the device. The name need not be unique among devices.

9.6 Channel Membership Packet

This packet is used by a Configuration Server to notify a device of the channel membership. For the sake of simplicity this messages contains the IP address and port number of all CNP/IP devices in the IP channel. Once a device has the list of IP addresses it can send a device request message to each device in the list requesting its full configuration.

The channel membership packet has the format shown in Table 16.

Table 16 —Channel Membership Packet format

Byte 0	**Byte 1**	**Byte 2**	**Byte 3**
Common Packet Header			
Datetime			
Send List Datetime			
MBZ			
List Size		MBZ	
The following fields are repeated once for each CNP/IP device			
IP Unicast Address			
IP Unicast Port		MBZ	
Datetime of Channel Routing Packet for Device			

The following is a description of all fields in this packet.

Datetime
Datetime of the creation of the Channel Membership packet for this channel. See 9.4.

Send List Datetime
Datetime of creation of the Send List packet for this device. See 9.4.

MBZ
Must Be Zero. See 3.9

List Size
The number of devices in the table of CNP/IP devices.

IP Unicast Address
The unicast IP address of the CNP/IP device in network byte order. The CNP/IP device will listen for incoming unicast IP packets on this address.

IP Unicast Port
The port number the CNP/IP device uses when listening for unicast IP messages.

Datetime of Channel Routing packet for Device
This is the datetime of the most recent Channel Routing packet for the device that is available to the Configuration Server. This is used to determine whether to request a new channel routing packet from the configuration server.

9.7 Channel Routing Packet

The Channel Routing packet has the format shown in Table 17.

Table 17 — Channel Routing Packet formats

Byte 0	**Byte 1**	**Byte 2**	**Byte 3**
Common packet header with packet type (0x08)			
Datetime			
IP Multi-cast Port		IP Unicast Port	
IP Mc Address			
IP Unicast Address			
IP Flags	CNP Router Type	CNP Flags	Node Type
Total Unique ID Bytes [U]		MBZ	
Total SubnetNode Bytes [S]		Total Domain Bytes [D]	
Unique Id [0]	Unique Id [1]	Unique Id [2]	Unique Id [3]
Unique Id [4]	Unique Id [5]	Unique Id [6]	… [U-1]
the following fields are repeated for each node address (CNP node addresses)			
Subnet Number [0]	Node Number [0]	Domain Index [0]	
Unique Id Index [0]		Subnet Number [1]	Node Number [1]
Domain Index [1]		Unique Id Index [1]	
Subnet Number [S-1]	Node Number [S-1]	Domain Index [S-1]	
Unique Id Index [S-1]		…	…
The following fields are repeated once for each domain (CNP Domain List)			
SubnetMsk[0]	SubnetMsk[1]	SubnetMsk[2]	SubnetMsk[3]
SubnetMsk[4]	SubnetMsk[5]	SubnetMsk[6]	SubnetMsk[7]
…	…	…	…
SubnetMsk[28]	SubnetMsk[29]	SubnetMsk[30]	SubnetMsk[31]
GroupMsk[0]	GroupMsk[1]	GroupMsk[2]	GroupMsk[3]
…	…	…	…
GroupMsk[28]	GroupMsk[29]	GroupMsk[30]	GroupMsk[31]
Domain Length	MBZ	Domain[0]	Domain[1]
Domain[2]	Domain[3]	Domain[4]	Domain[5]

NOTE The squared brackets contain the index of a field in an array.

The following is a description of fields in this packet.

Datetime
See 9.4 for a description of this field.