

ANSI/CEA Standard

Enhanced Protocol for Tunneling Component Network Protocols over Internet Protocol Channels

ANSI/CEA-852.1-A

April 2014



CEA
Consumer Electronics Association

Documents
www.CE.org
Document Sharing Hub

NOTICE

Consumer Electronics Association (CEA[®]) Standards, Bulletins and other technical publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such Standards, Bulletins and other technical publications shall not in any respect preclude any member or nonmember of CEA from manufacturing or selling products not conforming to such Standards, Bulletins or other technical publications, nor shall the existence of such Standards, Bulletins and other technical publications preclude their voluntary use by those other than CEA members, whether the standard is to be used either domestically or internationally.

Standards, Bulletins and other technical publications are adopted by CEA in accordance with the American National Standards Institute (ANSI) patent policy. By such action, CEA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the Standard, Bulletin or other technical publication.

This document does not purport to address all safety problems associated with its use or all applicable regulatory requirements. It is the responsibility of the user of this document to establish appropriate safety and health practices and to determine the applicability of regulatory limitations before its use.

This document is copyrighted by the Consumer Electronics Association (CEA[®]) and may not be reproduced, in whole or part, without written permission. Federal copyright law prohibits unauthorized reproduction of this document by any means. Organizations may obtain permission to reproduce a limited number of copies by entering into a license agreement. Requests to reproduce text, data, charts, figures or other material should be made to CEA.

(Formulated under the cognizance of the CEA **R7 Home Networks Committee.**)

Published by
©CONSUMER ELECTRONICS ASSOCIATION 2014
Technology & Standards Department
www.CE.org

All rights reserved

FOREWORD

This standard was developed under the auspices of the Consumer Electronics Association (CEA) R7 Home Networks Committee.

Unless otherwise indicated, the designator 852.1 refers to the latest revision of CEA-852.1, which at the time of this writing was CEA-852.1-A.

Unless otherwise indicated, the designator 709 and 709.1 refers to the latest revision of CEA-709.1, which at the time of this writing was CEA-709.1-D.

Unless otherwise indicated, the designator 852, and references to CEA-852-A and CEA-852-B refers to the latest revision of CEA-852, which at the time of this writing was CEA-852-C.

[This page intentionally left blank.]

Table of Contents

Table of Contents	iii
<i>Enhanced Protocol for Tunneling Component Network Protocols Over Internet Protocol Channels.....</i>	<i>1</i>
1 General	1
1.1 Scope.....	1
1.2 Definitions.....	1
1.3 Acronyms	5
2 Normative References.....	6
2.1 Reference Acquisition	7
3 Introduction.....	7
4 Requirements.....	9
5 852.1 Device and Channel Specification	9
5.1 IP Transport Mechanisms.....	13
5.1.1 Informative Considerations	14
5.1.2 BOOTP and DHCP.....	14
5.2 Device IDs	15
5.3 Device CN Configuration	16
5.3.1 CN Packet Formats	16
5.3.2 CN Addressing Schemes	16
6 Configuration Data & Services	18
6.1 DateTime.....	18
6.2 Data Ownership Classes	19
6.3 Notification Services	20
7 852.1 Packet Exchange.....	20
7.1 Introduction.....	20
7.2 Packet Type Categories	22
7.3 Session IDs	23
7.3.1 Session IDs for Hybrid Multi-cast and Uni-cast Channels	23
7.3.2 Session ID Rules.....	24
7.4 Backward and Forward Compatibility	26
7.5 Vendor Specific Messages	26
7.6 Message Segmentation.....	27
7.7 Packet Aggregation.....	28

7.8	Authentication	28
7.8.1	Introduction	28
7.8.2	Authentication Procedure	29
7.9	NAT Support	31
7.9.1	Introduction	31
7.9.2	Description	32
7.9.3	Standard NAT Extended Headers	32
7.9.4	Algorithm	34
7.10	Dynamic IP Support	36
7.10.1	Dynamic IP Resolution using DNS	37
7.10.2	Probing Method	38
7.10.3	Probe Interval and Dynamic NAT Addresses	41
7.10.4	Triggering IP probes	41
8	CN Data Packet Exchange	42
8.1	Out of Order Packets	42
8.2	Duplicate Packet Detection	44
8.3	Stale Packet Detection	44
8.4	Selective Forwarding	45
8.5	Send List	48
9	852.1 Transactions	49
9.1	852.1 Transaction Description & Theory	49
9.2	Transaction Timeout	52
9.3	Retries	52
9.4	Transaction Rules	53
9.5	Packet Assignment	55
9.5.1	Assignment Rules	55
9.6	Duplicate Configuration Data Detection	56
9.7	Duplicate Configuration Transaction Packets	56
9.8	Configuration Packet Exchange Rules	57
9.8.1	Solicited Packets	57
9.8.2	Unsolicited Packets	62
10	Configuration and Operation Modes	66
10.1	Channel Formation	66
10.2	Updating Dynamic Device Specific Information	67
10.3	Configuration Server Presence	68
10.4	Multiple Channel Configuration Servers	69
10.5	Channel Operation Modes	69
10.5.1	Manual Mode	69
10.5.2	Server Mode	69
10.5.3	Peer Mode	69
10.5.4	Peer Notify Mode	70

10.5.5	Auto Mode	70
10.6	Device Operation Modes	70
10.7	Responsibilities to Answer Messages and Update Information	71
10.8	Device and Configuration Server Interactions	72
10.8.1	General Device Interaction	72
10.8.2	Device Registration	73
10.8.3	Channel Membership	73
11	Packet Types and Headers	74
11.1	Packet Type and Message Categories	74
11.2	Packet Headers	76
11.3	Common Sub-header Definition	77
11.4	CN Data Packet Header Specific Fields	80
11.5	Configuration Data Packet Header Specific Fields	80
11.5.1	Transaction Sub-Header	82
11.6	Stacked Extended Packet Headers	85
11.6.1	Standard NAT Extended Headers	86
11.6.2	Authentication Extended Header	87
12	Configuration Data Message Sections	88
12.1	Configuration Data and Diagnostic Section Types	88
12.2	Configuration Request Section	89
12.3	Configuration Acknowledgement Section	90
12.4	Configuration Data Section Header	91
12.5	Configuration Data Sections	91
12.5.1	Channel Parameters Section	92
12.5.2	CS Self ID Section	93
12.5.3	CS IP Uni-cast Section	94
12.5.4	CS IP Node Name Section	94
12.5.5	Channel Membership Section	95
12.5.6	Channel Send List Section	98
12.5.7	Channel State Section	99
12.5.8	Device ID and Mode Section	99
12.5.9	Device DNS Server Section	101
12.5.10	Device Time Server Section	102
12.5.11	Device IP Probe Target Section	102
12.5.12	Device Self ID Section	103
12.5.13	Device IP Node Name Section	104
12.5.14	Protocol Revision Section	104
12.5.15	Device IP Uni-cast Section	106
12.5.16	Device IP Multi-cast Section	106
12.5.17	Device Channel Routing Section	107
12.5.18	Device State Section	111
12.6	Diagnostic Packet Configuration Sections	111
12.6.1	IP Probe Request Section	111
12.6.2	IP Probe Response Section	112
12.6.3	Device Status Section	113

12.6.4	Device IP Uni-cast Hint Section	116
13.	<i>Optional Special Features</i>	116
13.1	<i>Hybrid Channel Bootstrap</i>	116
13.1.1	Introduction.....	116
13.1.2	Device Behavior	117
13.1.3	Configuration Server Behavior	118
Annex A	<i>Specifications for CEA-709.1-D (Normative)</i>	120
Annex B	<i>Specifications for CEA-709.1-D (Informative)</i>	122
Annex C	<i>Specifications for CEA-600 (Normative)</i>	122
Annex D	<i>Specifications for CEA-600 (Informative)</i>	122
Annex F	<i>Example Channel and Device Configuration Using Sectioned Configuration Messages (Informative)</i>	122
	CS adds a device to the channel	123
	Device registers in a new channel	124
	Remove a device from the channel	126
Annex F	<i>Example Diagnostics (Informative)</i>	126
	NAT Address Discovery	126
	Firewall/NAT Diagnosis	127

Enhanced Protocol for Tunneling Component Network Protocols over Internet Protocol Channels

1 General

1.1 Scope

The CEA-852.1 standard specifies a communications method that allows networked data acquisition and control devices to communicate with each other over the internet. The purpose of such devices are widely varying and include functions such as appliance monitoring, meter reading, and HVAC and lighting control to name a few. CEA-852.1 does not replace existing device communications protocols, but instead allows those protocols to use the internet as a communications medium. CEA-852.1 currently supports the existing device protocols CEA-600 (CEBus) [1] and CEA-709.1 (LonTalk®) [2] and was designed to allow the support of others.

1.2 Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3].

A Component Network (CN) is a network of nodes (devices) that are capable of computing, sensing, and actuating. Typically these devices are used for control and telemetry purposes including applications such as HVAC, security, energy management, machine control, etc. These devices are typically not used for data processing or general purpose computing purposes. Some of the characteristics of CNs that set them apart from traditional data networks include:

- Typically lower bandwidth requirements.
- Small packet/message sizes.
- Tighter constraints on message reliability and delivery latencies.

A number of existing standards apply to networks of this type including CEA-709.1 [2] and CEA-600 [1].

Tunneling refers to the encapsulation of one protocol's packet within the payload of another protocol's packets.

The Internet Protocol (IP) is a packet switched protocol that allows networks to be built that consists of a loose confederation of hosts that can communicate over a wide range of physical media and topologies to exchange data. Refer to RFC 791 [4].

An IP Packet is a contiguous chunk of data transferred over the Internet using standard Internet protocol (IP). Each packet begins with a header containing addressing and system control information. The terms IP packet, IP frame, IP fragment, and IP datagram

are often used synonymously. Only UDP/IP and TCP/IP packets are considered in this specification.

A CN Packet is a contiguous chunk of data transferred over a Component Network (CN). Each packet begins with a header containing CN Protocol addressing and system control information.

An 852.1 Message is a collection of data that is tunneled over the IP network between two or more 852.1 devices and is meant to be processed as a whole. When small enough, a complete 852.1 Message MAY be encapsulated as the payload portion of a single 852.1 Packet. When not small enough, an 852.1 Message MUST either be sent as a data stream using TCP or must be segmented into multiple 852.1 Packets and sent using UDP. The 852.1 Message is then re-assembled at the receiving end by appropriately concatenating the payloads portions of the associated 852.1 Segments.

A Configuration Data Message is an 852.1 Message that is composed of 852.1 configuration data.

An 852.1 Packet or 852.1 Segment is a block of data with an 852.1 packet header that is tunneled over an IP network between two 852.1 devices. An 852.1 Packet MUST fit within one IP packet whose size is typically bounded by the minimum MTU (Maximum Transmission Unit) less any IP headers and trailers. Multiple 852.1 Packets may be contained within one IP packet. The payload of an 852.1 Packet or Segment may be all or part of an 852.1 Message. Although structurally identical, typically Packet is used when the associated 852.1 Message fits completely within a single payload and Segment is used when the 852.1 Message is divided up into multiple payloads.

A CN Data Packet is an 852.1 Packet that uses the 852.1 CN Data Packet Header and has as a payload a CN Packet. Segmentation cannot be used with 852.1 CN Data Packets. 852.1 CN Data Packets may not be sent using TCP/IP streams.

A CN Data Message is an 852.1 Message that is composed of a single CN Data Packet.

A Configuration Data Packet or Segment is an 852.1 Packet that uses the 852.1 Configuration Packet Header and has all or part of a Configuration Data Message as its payload.

An Aggregation of is a group of 852.1 Packets that are concatenated and tunneled via a single UDP/IP packet.

A CN device is considered to be any device that uses the CN protocol to communicate with other CN devices. The CN protocols that are compatible with the CEA-852-C [5] standard are provided in the appendices. References to CN devices in this specification assume the device is using a compatible CN protocol. All CN devices are either CN Routers, CN Nodes or some collection of CN Router and CN Nodes. A CN Protocol analyzer is a special case of a CN Router.

A CN router is special type of CN device that routes CN protocol packets between two or more channels.

A CN node is special type of CN device that can send or receive CN protocol packets, but does not route them between channels.

A CN group is a collection of CN devices that share a common multi-cast address in the CN protocol.

A CN/IP device is a CN device that communicates with other CN devices over an IP channel using either the 852 or 852.1 protocols. All CN/IP devices are CN/IP Routers, CN/IP Nodes or some collection of CN/IP Router and CN/IP Nodes. A CN/IP Protocol analyzer is a special case of a CN/IP Router. In every case all the nodes and routers in a CN/IP device must use either 852.1 or 852 but not both.

852.1 Device is a CN/IP device that uses the 852.1 protocol.

852 Device is a CN/IP device that uses the legacy 852 protocol. Unless otherwise specifically indicated an 852 device refers to a device that is compliant with 852 protocol revision B or later.

A CN/IP router is a CN router in which at least one of the channels it routes packets over is an 852 or 852.1 tunnel on an IP channel. An 852.1 router uses the 852.1 protocol. An 852 router uses the legacy 852 protocol.

A CN/IP node is a CN node in which at least one of the channels it sends and receives packets over is an 852 or 852.1 tunnel on an IP channel. An 852.1 node uses the 852.1 protocol. An 852 node uses the legacy 852 protocol.

A Bilingual Device supports both the 852 protocol and the 852.1 protocol and is allowed to operate using either protocol.

A CN Channel refers to the common communications transport mechanism that a specific collection of CN devices share and communicate over without the use of a router. Channels are used to transport CN packets below the link layer of the CN protocol stack. 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.

An 852.1 channel is a special case for a CN Channel. It is a virtual or logical channel consisting of a collection of 852.1 devices that are configured such that they can intercommunicate. Devices on an 852.1 channel may be on different IP channels, domains, and subnets.

An 852 channel is a special case for a CN Channel. It is a virtual or logical channel consisting of a collection of 852 devices that are configured such that they can intercommunicate. Devices on an 852 channel may be on different IP channels, domains, and subnets.

The Address Resolution Protocol (ARP) is a TCP/IP protocol used to convert an IP address into a physical address such as an Ethernet address. A host wishing to obtain a physical address broadcast an ARP request onto the TCP/IP network. The host on the network that has the IP address in the request then replies with its physical hardware address.

There is also Reverse ARP (RARP) that can be used by a host to discover its IP address. In this case, the host broadcasts its physical address and a RARP server replies with the host's IP address.

The Internet Control Message Protocol (ICMP) is an extension to the Internet Protocol (IP) defined by RFC 791 [4]. ICMP supports packet containing error, control, and informational messages. The PING command, for example, uses ICMP to test an Internet connection.

Internet Group Management Protocol (IGMP) is defined in RFC 1112 [6] as the standard for IP multi-casting in the Internet. It's used to establish host memberships in particular multi-cast groups on a single network. The mechanisms of the protocol allow a host to inform its local router, using Host Membership Reports that it wants to receive messages addressed to a specific multi-cast group. All hosts conforming to level 2 of the IP multi-casting specification require IGMP.

A domain is a group of computers and devices on a network that are administered as a unit with common rules and procedures. Domains are defined by an address referred to as the "domain address". Different network protocols specify the domain address in different ways.

A subnet is a portion of a network that shares a common address component known as the "subnet address". Different network protocols specify the subnet address in different ways.

A node ID is a logical network address that differentiates nodes within the same subnet or domain.

A unique ID is a network address that is globally unique. No two nodes anywhere can share the same unique ID.

Data Skew refers to the situation that exists when a data set that forms the basis for a message changes while the message is in the course of being transmitted to another device.

The term MBZ in protocol fields stands for Must Be Zero. This stands for a reserved field that may be used in following versions of the protocol. Such fields shall be sent as zero and ignored by the receiver in implementations conforming to the current version of the specification.

Public IP address is an IPv4 address that could be directly accessible by other IP devices. These are typically used on the WAN side of an IP router.

Private IP address is from one of the designated address ranges such as 10.0.0.0. through 10.255.255.255. Private addresses cannot be accessed directly by nodes on the WAN side of an IP router. Private addresses are used on the LAN side of an IP router.

1.3 Acronyms

ACK	Acknowledge
ARP	Address Resolution Protocol
BOOTP	Bootstrap Protocol
CN	Component Network
CRC	Cyclical Redundancy Check
CTP	Channel Timeout Period
DHCP	Dynamic Host Configuration Protocol
GMT	Greenwich Mean Time
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Number Authority
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LAN	Local Area Network
LFS	Last Forwarded Sequence
LPDU	Link layer Protocol Data Unit
MAC	Media Access Control
MBZ	Must Be Zero
MPDU	MACS Protocol Data Unit
NAT	Network Address Translation
NTP	Network Time Protocol
PDU	Protocol Data Unit
PSN	Packet Sequence Number
QOS	Quality of Service
RARP	Reverse ARP
SA/DA	Source Address/Destination Address
SID	Session ID
SNTP	Simple Network Time Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UTC	Coordinated Universal Time

WAN Wide Area Network

2 Normative References

- [1] CEA, “CEBus Standard”, CEA STANDARD CEA-600 CEBus Set, March 1996
- [2] CEA, ANSI/CEA-709.1-D, Control Network Protocol Specification, April 2014
- [3] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, BCP 14, RFC 2119, Harvard University, March 1997
- [4] Postel, J., “Internet Protocol”, STD 5, RFC 791, USC Information Sciences Institute, September 1981
- [5] CEA, ANSI/CEA-852-C Tunneling Device Area Network Protocols Over Internet Protocol Channels, April 2014
- [6] Deering, S.E., “Host extensions for IP multi-casting”, STD 5, RFC 1112, Stanford University, August 1989
- [7] Postel, J., “User Datagram Protocol”, STD 6, RFC 768, USC Information Sciences Institute, August 1980
- [8] Postel, J., “Transmission Control Protocol”, STD 7, RFC 793, USC Information Sciences Institute, September 1981
- [9] Droms, R., “Dynamic Host Configuration Protocol”, RFC 2131, Bucknell University, March 1997
- [10] Croft, W.J., Gilmore, J., “Bootstrap Protocol”, RFC 951, September 1985
- [11] Mills, D., “Network Time Protocol (Version 3) specification, implementation and analysis”, RFC 1305, University of Delaware, March 1992
- [12] Mills, D., “Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI”, RFC 2030, University of Delaware, October 1996
- [13] Rivest, R. “The MD5 Message-Digest Algorithm”, RFC 1321, April 1992

- [R1] R. Droms, Dynamic Host Configuration Protocol. RFC 2131, Standards Track, March 1997
- [R2] Dynamic DNS, www.dyndns.org
- [R3] J. Rosenberg, et al., STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). RFC 3489, Standards Track, March 2003

2.1 Reference Acquisition

CEA Standards:

Global Engineering Documents, World Headquarters, 15 Inverness Way East, Englewood, CO USA 80112-5776; Phone: 800-854-7179; Fax: 303-397-2740; Internet: <http://global.ihs.com>; E-mail: global@ihs.com.

IETF Documents:

Internet Engineering Task Force (IETF) Secretariat, c/o Corporation for National Research Initiatives, 1895 Preston White Drive, Suite 100, Reston, VA 20191-5434, USA; (703) 620-8990 / (703) 620-9071 (FAX); URL: www.ietf.org, IETF RFCs may be downloaded from www.ietf.org/rfc.html, IETF Internet drafts may be downloaded from www.ietf.org/ID.html.

3 Introduction

The protocol described herein, that is, the CEA-852.1 protocol is a non backwards compatible enhanced replacement for the CEA-852 protocol. After several years of active use in the field, it was determined that that CEA-852 protocol needed to be upgraded primarily to enable better support for dynamic IP addressing, NAT router support, better scalability and performance for large networks, more flexible configuration services and eventual support for IPV6. Because of the extreme difficulty in adding support for these features in both an elegant and backwards compatible way, it was decided to break backwards compatibility. Instead, the approach contemplated for backwards compatibility is to use Bilingual devices that can run in either 852 or 852.1 mode and correctly determine which mode to run in.

Despite the fact that CEA-852.1 is not backwards compatible, it still retains much of the flavor of CEA-852 and where possible preserved field names and functionality. Because of the many similarities between the two protocols, effort was made to point out where the CEA-852.1 specification differs from CEA-852. It is expected that CEA-852 will continue to be deployed for some time and in some cases in hybrid networks that have both 852.1 and 852 channels. Unless otherwise indicated, the designator 852 refers to the latest revision of CEA-852, which at the time of this writing was CEA-852-C [5].

This document provides specifications for the transporting of Component Network (CN) packets over Internet Protocol (IP) networks using a tunneling mechanism wherein the CN packets are encapsulated within the IP packets. It applies to both CN nodes and CN routers.

The purpose of this specification is to insure interoperability between various CN devices that wish to use IP networks to communicate using a particular CN protocol. The specification is intended to be general enough to handle a wide range of CN protocols. In particular this specification will cover the following standard CN protocols:

- CEA-709.1 [2]
- CEA-600 [1]

The main body of this specification is independent of the CN 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 CEA-709.1 [2]. Likewise Annex C and Annex D are the normative and informative, respectively, aspects of this specification as they apply to CEA-600 [1].

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

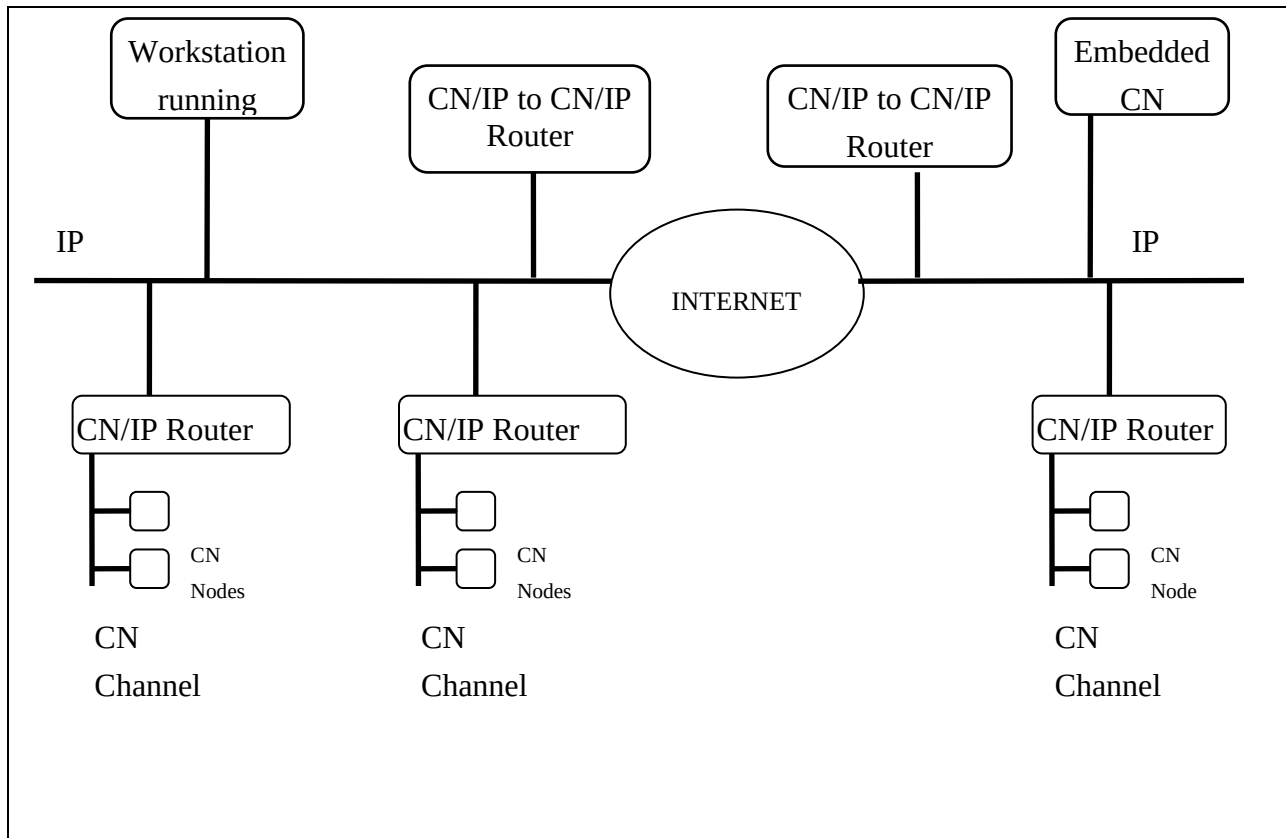


Figure 1 Typical CN/IP Application

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

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

4 Requirements

The following is a set of general requirements for the transporting of CN 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 tunneling protocol should not rely on the existence of a socket interface or how that interface may be used.
- Insure that CN packet ordering is preserved.
- Insure that CN 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 prioritize IP packets.
- Optional security measures to prevent malicious users from tampering with devices.
- Scalable.
- Allow status information to be extracted from 852.1 devices.
- Support the exchange of configuration information between 852.1 devices and configuration servers.
- Provide for the bootstrapping of hybrid 852.1/852 devices.

5 852.1 Device and Channel Specification

An 852.1 device has a dual personality. From a CN point of view, it is a CN node on a CN channel and has all such characteristics. These parameters can be configured and managed using the standard CN network management procedures and messages.

From the IP point of view, the 852.1 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 852.1 channel associated with that 852.1 device.

In general all IP host and channel parameters will be configurable using a number of techniques and protocols. As a minimum all 852.1 devices **MUST** 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 tunneling as described above to the other devices on the IP channel.

IP channels are not like typical CN channels that currently exist. Typical CN 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 CN 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 must 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 CN devices on an IP channel are much higher than those experienced with typical CN channels.

As depicted in Figure 1 the IP channel is used as an intermediary transport mechanism for the CN packets by a variety of 852.1 devices. When a CN packet is transported on an IP channel, an IP packet encapsulating the CN packets is sent to other 852.1 devices on that IP channel. On reception of one of the IP messages by an 852.1 device the CN packets are extracted and processed. A single IP packet may contain more than one CN packet. Therefore the IP packets must be formatted in such a way to allow the extraction of the individual CN packets. This is referred to a packet “aggregation”. 852.1 devices **MUST** support the reception of aggregated packets. Likewise the aggregation **MUST** be done in such a fashion that each CN packet contained within an aggregated IP packet is complete, i.e. CN packets should not cross IP packet boundaries as a result of aggregation. It is also a requirement that intermediate IP devices be capable of unbundling aggregated CN packets and aggregating them in a different manner before forwarding them.

An 852.1 device **MUST** 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. An 852.1 device **MUST** have at least one IP Uni-cast address and **MAY** be capable of belonging to as many as 32 multi-cast groups. It is **OPTIONAL** that an 852.1 device support multi-casting. This document does not address the routing of IP packets between subnets or through the Internet. The CN/IP devices **MUST** be compatible with whatever standard mechanisms (IP routers, switches, etc.) are required to perform the IP routing functions.

An 852.1 Channel is specified by a list of unique Device IDs (DID). Associated with each Device ID is the current IP addressing information that allows other 852.1 devices to communicate over the IP network to the associated Device. Because the IP address may be dynamic or the device may be behind a NAT router or both. The nature of and the manner in which the IP addressing information is used are context and time dependant. Each device, however, **MUST** be uniquely addressable on the IP network.

The IP addressing information might be a DNS registered IP Node Name that is converted dynamically into an IP Uni-cast host and port address.

For example, multiple devices MAY share the same IP Uni-cast host address but MUST have unique IP ports. Conversely, multiple devices may all use the same port but must have different host addresses.

When devices are behind NAT routers, more than one IP address may be needed for a given device depending on whether the communicating device is also behind the NAT router or not.

If every 852.1 device on an IP channel contained an updated list of the current IP Uni-cast addresses for every other 852.1 device on that IP channel, this is all that would be required to enable the tunneling of CN packets. In order to make the problem tracking the changes in IP addresses tractable, CEA-852.1 assigns each device in an 852.1 channel a unique Device ID.

In the most brute force approach, for each CN packet to be forwarded on the IP channel a separate IP Uni-cast message could be sent to each 852.1 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 852.1 device. Therefore a complete definition of the 852.1 channel should contain not only the Device IDs and IP Uni-cast addresses of all the 852.1 devices on the channel but also the IP Multi-cast groups to which they belong. Each 852.1 device can belong to up to 32 multi-cast addresses.

Selective Forwarding

Selective forwarding refers to examining the contents of the CN packet before forwarding it to determine if it should be sent to a particular 852.1 device. In order to do this additional CN specific information must be known about each potential destination. If the 852.1 device is a router then the information necessary to perform selective forwarding is the routing tables of the 852.1 router. If the device is simply a node then the domain, subnet, node ID, unique ID, and CN groups that the node belongs to should be known. Therefore all this information is also part of a complete 852.1 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 852.1 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 CN/IP device the following conditions are always true:

1. CN protocol packets are always received by all 852.1 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 852.1 devices should receive a CN 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.

2. A specific CN packet should never be transmitted twice to the same 852.1 device unless it is because of some retry mechanism above the link layer of the CN protocol stack. Due to the nature of IP networks it may happen that an 852.1 device may receive duplicate IP messages, but this should never be the result of the message being transmitted more than once from another 852.1 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 852.1 devices belonging to CN domain ID 'W'. If a CN 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.

Send List

In recognition of the fact that the complete IP channel definition can be unwieldy to use and maintain it is not a requirement that 852.1 device use it to forward packets. An alternative data structure called the "send list" can be maintained within each 852.1 device. The send list may contain both Uni-cast and multi-cast addresses and is subject to the same conditions given above. It can be created and loaded into the 852.1 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 CN packets and is structured to simplify the forwarding process such that the 852.1 device need only forward packets to every address (Uni-cast or multi-cast) in the send list. In order to allow 852.1 devices to blindly forward packets to each address in the list the following conditions must be true:

1. CN protocol packets MUST be received by all 852.1 devices that need to receive them regardless of whether they are routers or nodes (same as above).
2. A specific CN packet is never transmitted twice to the same 852.1 device (same as above).
3. 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 CN 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 852.1 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 the 852.1 device; this means that the send list may be provided to a device through some explicit configuration process. Although the send list is a configured property it does not preclude an 852.1 device from performing self-configuration by generating its own send list.

In order to have tight controls over the behavior of 852.1 devices and how they forward packets it SHOULD be possible to configure an 852.1 device to use an explicit send list and ignore any IP channel configuration information it may have.

5.1 IP Transport Mechanisms

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.

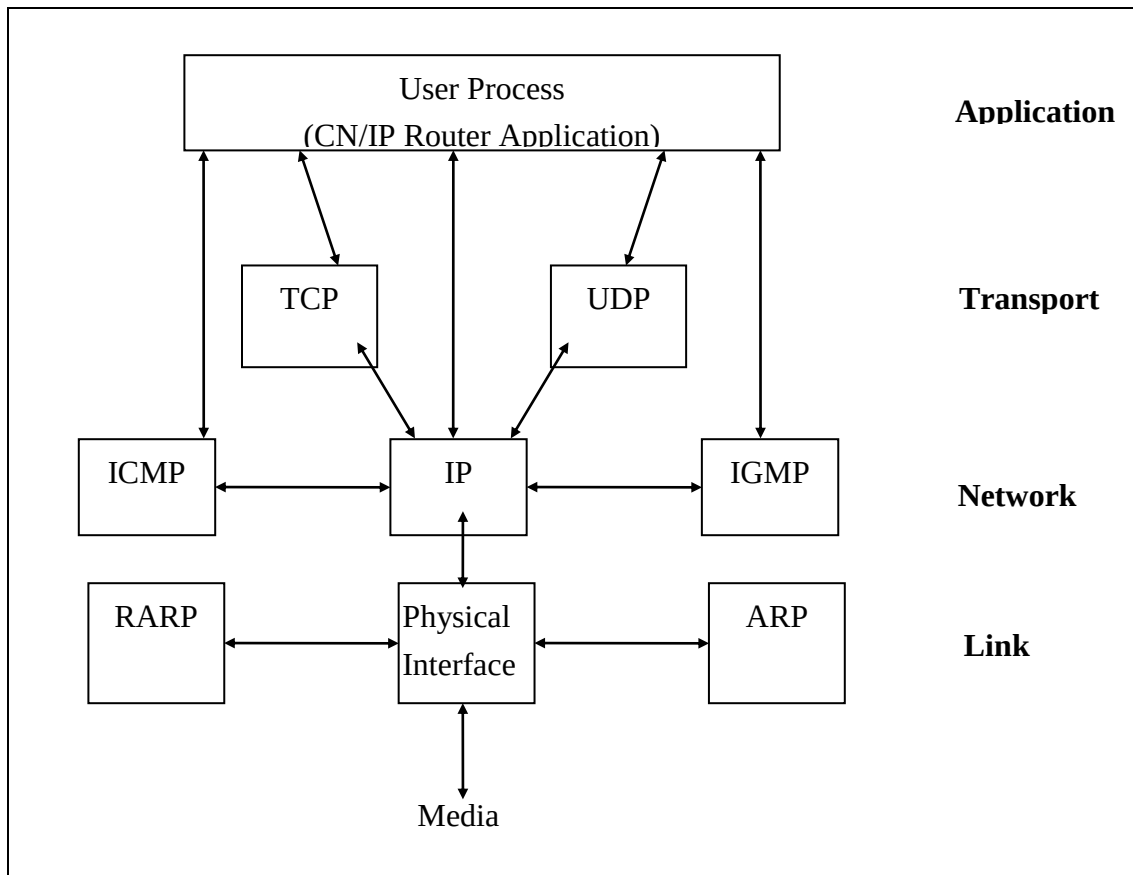


Figure 2 IP Protocol Stack

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

- Raw IP
- TCP
- UDP

UDP [7] and TCP [8] are transport protocols built on top of IP [4]. Given the increased efficiencies of UDP regarding the transport of CN data messages and its support of multi-cast addressing, it will be used to communicate between CN/IP devices. All CN/IP

devices MUST support UDP. All UDP datagrams MUST be transmitted with valid checksums.

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 CN/IP devices is OPTIONAL.

To address the sequencing issue there will be a sequence number added to the header of CN Data packets to help in sequencing them. 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 (Informative) and Annex D (Informative) for recommendations on port numbers to use for CEA-709.1 [2] and CEA-600 [1] respectively.

Using UDP, datagrams can be sent using either Uni-cast or multi-cast addressing. Uni-cast 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 RECOMENDED that 852.1 devices support both Uni-cast and multi-cast IP addressing. A channel definition or send list can contain any combination of Uni-cast and multi-cast addresses. It is not required that 852.1 device support multi-cast in order to inter-operate with 852.1 device that does.

In order for an 852.1 router to use multi-cast addressing across IP routers it will be necessary for the 852.1 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].

5.1.1 Informative Considerations

Some IP networks contain NAT routers. CEA-852.1 includes an optional method for working across NAT routers.

5.1.2 BOOTP and DHCP

Background

Two mechanisms exist for devices to obtain an IP address without prior configuration: DHCP (RFC 2131 [9]) and BOOTP (RFC 951 [10]).

DHCP is actually an extension of BOOTP and BOOTP servers that are compliant with RFC 951 [10] can understand messages from DHCP clients and respond to those messages correctly.

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

Compliance

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

5.2 Device IDs

The 852 protocol made the implicit assumption that 852 devices had static IP addresses. Consequently the static IP address could be used as the unique identifier of a device on an 852 channel. Because CEA-852.1 needs to provide support for dynamic IP addresses including NAT, a static IP assumption is no longer valid. One approach to supporting dynamic IP addresses is to track the changes in a device's IP address and provide a mechanism for updating the channel membership list and configuration data.

Moreover, another requirement of CEA-852.1 is that the channel can continue to operate without an online configuration server. This requirement is more difficult to satisfy with dynamic IP addresses as the configuration server is not available to help track the IP address changes. Without the configuration server each device has to do more work to track changes. Schemes were investigated to track and update IP address changes both with and without an online configuration server. In every case some other unique identifier for each device was needed to help track IP address changes. Without a unique device identifier (besides the IP address) tracking IP address changes gets very complicated and prone to failure. As a result, CEA-852.1 uses a unique Device ID for each device on a channel. The Device ID is used for both Configuration and CN Data packets. This changes the fundamental way that a devices indexes information about other devices. The unique identifier for a device on an 852.1 channel is the Device ID and not the IP Uni-cast Host:Port address as is the case with CEA-852. In CEA-852.1, changes in IP addresses constitute a change in the configuration of a channel not the identity or specification of a channel.

A device's ID (DID) is 16 bits long and **MUST** be unique on a given channel. Device ID 0 is special and is not a valid device ID. The configuration server, if it exists, also has a DID. This allows up to 65,534 devices on a channel although other limitations will reduce the number of allowed devices to well below 65,534. A previously unused field in the configuration packet header now holds the Source Device ID (SDID) and Destination Device ID (DDID) for a given configuration packet. The motivation for using a number for the DID is improved speed when indexing devices in a channel or looking up a device's configuration data or transaction data. A text based URL or IP Node Name would be much slower.

When in Normal mode the DID is assigned by the configuration server. This poses a bootstrap problem. In order for a device to register with the configuration server it **MUST** initiate a transaction with the configuration server before the device has been given a DID. As a result, the initial registration transaction **MUST** be conducted without DIDs and the configuration server and registering device **MUST** temporarily use IP addresses to identify the transaction until the registration is completed. This implies that the registration process might not complete successfully if the IP address of either the

configuration server of device change during registration. Since registration should complete in milliseconds, this is not a serious problem.

To avoid ambiguity, in the case of a failed registration or when removing a device from a channel, the configuration server **MUST** not reuse that device's DID unless that device has been explicitly deleted from the channel and the DID removed from the device's internal registration data. This is known by the configuration server if the device positively acknowledges the change. This might be accomplished by sending a registration packet to the device with the device's DID set to zero.

In some cases a device might drop off a channel when communication between the device and configuration server is interrupted. In this case the configuration server cannot zero out the device's DID or may not get a positive acknowledgement. In either case the configuration server **SHOULD** avoid reusing a missing device's DID for a new device on the same channel unless there are no more previously unused DIDs. This might be accomplished by incrementing the DID each time a new device is added to the channel and then rolling over when the count reaches the maximum.

In cases where the implementation limits the number of devices on a single channel to a sub set of 65,535 devices then the higher order bits of the DID could be used as a channel identifier. This might simplify indexing or management of multiple channels on the same network. Implementations must still account for the DIDs of removed devices that have not positively acknowledged they are no longer using their old DID.

Also if a device registers with a different configuration server or reregisters on a new channel, the device **MUST** assume that its prior DID is not longer valid and **MUST** zero out its DID.

5.3 Device CN Configuration

5.3.1 CN Packet Formats

The general format of CN packets which are tunneled over the IP channel are those packets that are received from or sent to the Link layer (layer 2) of the CN protocol stack. Refer to Annex A (CEA-709.1 Normative) and Annex C (CEA-600 Normative) for a precise specification of the packet formats corresponding to specific CN protocols. Link layer packets are called LPDU in the OSI protocol stack formalism.

5.3.2 CN Addressing Schemes

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

- Unique device ID
- Domain ID
- Subnet ID
- Node ID
- Group ID

Different CN protocols generally use different addressing schemes to exchange packets. Although it is generally not necessary to understand the contents of a CN packet or its addresses in order to tunnel CN packets over IP, some aspects of the CN 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 tunneling. Since CN 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 CN protocols. The following CN 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 IDs 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 IDs should be unique within a particular network. This means that in a particular network where Domain is used if two devices have the same Domain ID they belong to the same Domain. Domain IDs 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 IDs should be unique within a particular domain. This means that in a particular network where subnet IDs is used if two devices have the same Domain ID and the same Subnet ID, then they belong to the same Subnet. Some CNs does not use Domains in which case the Subnet may be the highest level of address for a device. Subnet IDs are generally logical in nature and can be changed and configured.
- Node. This is the lowest level of any hierarchical addressing scheme. Node IDs should be unique within a particular Subnet. No two devices within the same subnet should have the same Node ID. Node IDs 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 CNs 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 specific CN protocols to these terms. In general how the various addressing schemes work within a CN protocol is 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 a particular CN protocol. Some CN protocols support more routing functions than others. This is relevant because one of the optional configuration packets that can be exchanged include routing information. For those CN protocols that do not support routing this message may simply not be supported or the addresses specified within the message appropriately limited.

For each CN protocol supported in this specification, a table is given in the appropriate appendix that specifies how the appropriate addresses used in that protocol map to the terms given above. Refer to Annex A (CEA-709.1-D Normative) and Annex C (CEA-600 Normative) for the specific addressing conventions that correspond to the address types listed above.

6 Configuration Data & Services

In CEA-852, configuration data and configuration packets had a one to one mapping, that is, each assemblage of configuration data had a corresponding configuration packet and an associated configuration date time. In addition to updating or transmitting the associated configuration data, some configuration packets also provided a notification service of other configuration data by including the DateTimes of other configuration packet data.

Since these other “DateTimes” were included as part of the configuration data, changes in these other DateTimes resulted in a change in the configuration DateTime. This double duty has the unfortunate consequence that a hierarchical DateTime dependency chain was created with cascading updates. While this cascade was desirable and intentional in CEA-852, it has unfortunate side effects on channels with dynamic IP addresses. For large channels with dynamic configuration data, this cascading dependency results in unnecessary updates of configuration data merely to update the “other” DateTimes, when what is needed is purely a notification service.

In CEA-852.1 the DateTime hierarchy is flattened. Also the concept of “notification service” is added independent of the configuration data. We also specifically classify the types of configuration data according to the concept of “ownership” where ownership means the entity responsible for modifying the associated configuration data. This is distinct from tracking configuration data or storing a copy of configuration data owned by someone else.

There are two owners of configuration data, the Configuration Server and each Device. Normally, data is “owned” by one or the other. There is an exception in that some data owned by the Device might have its ownership overridden by the Configuration Server.

The configuration data is divided into sections. Each section has an associated DateTime. DateTimes belong to sections of configuration data not packets. The factors used to determine how data is partitioned into sections include, ownership, size, commonality of update frequency, and shared update causing events. This allows better matching of data sections to configuration packets and minimizes redundant transmission of unchanged configuration data. The cost of this better matching is bigger size due to the section headers.

6.1 DateTime

All configuration data sections 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 second 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 DateTime from RFC 1305 [11]. This is the number of seconds since 0h 1 January 1900. This time will expire in 2036. See RFC 2030 [12] Section 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 [Coordinated Universal Time], or equivalently GMT, so that the time zone does not affect the value. Also daylight saving time is not adjusted. Since UTC is used, if a device moves between time zones, or if daylight savings time comes into effect, 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 **MUST** never emit a configuration data message with a DateTime earlier than that for any configuration data message 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 data message that it emitted. Each time a configuration data message 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 data messages 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.

6.2 Data Ownership Classes

Data ownership refers to the concept of who is responsible for modifying the data. There are three classes of data ownership as follows:

1. Data owned by the Configuration Server
2. Data owned by the Device
3. Hybrid data owned by the Device but whose ownership can be overridden by the Configuration Server

Data that is owned by the Configuration Server is either channel wide data that is the same for every device or Configuration server specific data that characterizes the configuration server and is shared channel wide. The Configuration Server is responsible for updating the channel members anytime the Configuration Server or Channel data changes. Once a channel is configured and if the CS is removed, the CS owned data cannot be changed until the CS returns. The CS maintains the DateTime of these data.

Data that is owned by a Device is information specific to a given device. The Owner of the data is the associated device. A device is responsible for updating the other members

of the channel including the configuration server anytime that device's data changes. Depending on the operational mode the update may be via the CS or directly peer to peer. The device maintains the DateTime of these data.

Hybrid Data with overridden ownership is device specific data that benefits from central management by the configuration server. This special class is owned by the associated device by default, but the ownership MAY be overridden by the CS if the CS so chooses. Typically, the device sets the initial or default values but accepts any subsequent changes made by the CS. This data class typically shows up in the Device Registration packet but could also be a part of the CS Registration packet. The device maintains the DateTime of these data. When the CS overrides the data it sends a Sectioned Configuration packet containing the altered data sections using their original DateTimes. If the device accepts the override data it updates the DateTimes accordingly.

6.3 Notification Services

Independent of the need to update other devices on the channel with changes in configuration data, a configuration server or device also need to notify other devices about changes in configuration data so that a notified device is able to take appropriate action such as requesting the associated configuration data. This notification service is provided by including DateTimes in packets. The service could be provided by sending a packet with nothing but date times, however to reduce channel bandwidth load it is more efficient in some cases to piggyback the notification on another packet. When a DateTime field is included in a Section for notification purposes, the associated Section DateTime is not updated when the included DateTime changes. This avoids a cascade. In the document tables where configuration data sections are defined, notification fields are shaded in light gray.

In addition to DateTime change notification, a notification service is needed to bootstrap device to device communications on the channel. This is done by sending initial IP address information to all the devices in the channel via the Channel Membership list. The IP address information in the CM list is provided as notifications. The stored DateTime for a device's copy of the CM List is not updated if the IP address information is updated.

7 852.1 Packet Exchange

7.1 Introduction

The 852.1 IP tunneling protocol uses two categories of packet exchange between 852.1 devices: 1) Transactioned and 2) Transactionless. A Transactioned packet exchange involves a two-way transfer of a set of packets between two devices. The two main motivations for a transactioned exchange are to 1) provide reliable delivery of data typically with an Acknowledgement returned upon receipt of a data packet and/or to 2) unambiguously associate a data Response with the initiating Request for that data. The pair of devices participating in a transaction need to preserve state information about the

transaction until it completes. With request/response the response provides an acknowledgement so no additional actions are needed to verify delivery. Conversely a transactionless packet exchange is a one-way transfer and no state information is required (send and forget).

The 852.1 protocol has two main functions as follows: 1) Tunnel CN data packets across an IP network between 852.1 devices and 2) Configure 852.1 devices so they can communicate with each other. With the first function, the 852.1 protocol makes the assumption that the associated CN protocol is responsible for reliable packet exchange or request/response services. Consequently CEA-852.1 MAY use a transactionless one-way exchange to transport CN data packets. With the second function, the 852.1 protocol is responsible for accurately and reliably configuring the 852.1 devices. This requires a transactioned two-way packet exchange.

How CEA-852.1 operates for transactioned exchanges depends on the underlying IP communications mechanism. With UDP/IP there is no associated reliable delivery of packets and as a result, CEA-852.1 must provide its own mechanism for reliable delivery. With TCP/IP there is an associated reliable delivery of packets and CEA-852.1 does not need to provide its own mechanism for reliable delivery. In either case, however, CEA-852.1 must still provide a transactioned service for request-response.

UDP

The UDP payload length of approximately 548 bytes limits both the size of a configuration data message that may be transmitted without segmentation and the size of a CN Data packet. Because any 852.1 configuration data message may be segmented the UDP size restriction does not place any limits on the configuration or number of devices on an 852.1 channel.

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:

1. Configuration data message segmentation is never used. Configuration data messages regardless of size are sent in their entirety in one packet.
2. ACK messages are not sent on TCP connections.
3. Retransmissions (retries) are not performed on TCP connections.
4. 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 MUST answer using TCP.
5. When the configuration server detects a configuration change, it MAY send the configuration data unsolicited, over the TCP link to the device
6. 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.

1. The device attempts a TCP connection with the configuration server.
2. 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.

7.2 Packet Type Categories

There are two categories of 852.1 packet types corresponding to the two packet exchange mechanisms, transactioned and transactionless.

- 1) The transactionless category contains only the CN Data Packet type.
- 2) The transactioned category contains the Request packet type, the Response/Announce packet type, and the Acknowledgement packet type. These three types collectively are referred to as Configuration Packets. Acknowledgements can be interpreted as either positive or negative acknowledgements.

The Configuration packet types **MAY** be sent in a solicited manner as a response to a request or in an unsolicited manner directly without a request.

When an 852.1 Configuration Data Request is sent as a solicitation, either a response in the form of the requested Configuration Data or an Acknowledgment packet indicates reliable delivery of the request. Usually when an acknowledgement (using the appropriate Acknowledgement type) is returned it is to indicate that the receiving device cannot fulfill the request. If neither a Configuration Data packet nor an Acknowledgment is returned in response to a request then the requesting device MAY keep resending the request (retry).

When a Configuration Data packet is sent in an unsolicited manner, an acknowledgement (using the appropriate Acknowledgement type) is returned by the receiving device to indicate reliable delivery. If an acknowledgement is not returned the initiating device MAY keep resending the unsolicited Configuration Data packet (retry).

7.3 Session IDs

The Session ID behavior in CEA-852.1 has changed from CEA-852. CEA-852.1 has two different Session IDs: a CN Data Session ID (DSID) and a Configuration Session ID (CSID). CEA-852 had only one Session ID that was shared by both CN Data Packets and Configuration Packets.

In CEA-852 each new session was required to have a unique value for the Session ID. This required that the device record the last value of the Session ID in nonvolatile memory so that on a reset or power cycle it could guarantee that the new value for the Session ID was distinct (unique) from the last value. A new Session ID value could be chosen randomly as long as it was unique.

7.3.1 Session IDs for Hybrid Multi-cast and Uni-cast Channels

The physical medium for component network channels is typically some form of shared multi-drop channel. The relevant feature is that a packet sent by one node is received in parallel simultaneously by all the other nodes. This feature is difficult to emulate on an IP channel. The typical addressing mechanism is IP Uni-cast. To emulate a multi-drop channel with IP Uni-cast could require that each packet be sent N-1 times where N is the number of 852 devices in the channel. Whereas IP multi-cast more closely matches a multi-drop channel, since each packet sent is received by every device. If that were the only consideration, then multi-cast would be a far superior addressing scheme for CEA-852. However there is an associated scalability performance trade-off on an IP channel, that is, channel bandwidth versus device bandwidth as the number of devices increases. This trade-off is governed by the topology of the component network bindings, that is, many to many versus one to a few. Selective forwarding with Uni-cast can reduce the multiplier effect by intelligently forwarding packets only to devices that need it.

With multi-cast addressing, every 852 device must process every packet sent by every other device to see if the packet was meant for a given device. If most of the traffic is many to many then there is a high correlation or hit rate between packets received and kept versus packets received and then dropped. In this case Uni-cast would scale very

poorly since selective forwarding would not have much effect and the number of copies of each packet sent would be close to $N-1$. Alternatively, if most of the traffic is one to a few or one to many then Uni-cast addressing with selective forwarding would scale better since a given device will likely only process a subset of the total traffic and the number of copies of a message sent out would be much closer to 1 than to $N-1$.

Inexpensive 852 devices might not have the performance to process every IP packet and might not be able to process every 852 packet on large 852 channels. An optimal forwarding algorithm would use a combination of Uni-cast and multi-cast addressing based on the number of destination devices for a given packet. If the number of destinations is close to one then Uni-cast is better. If the number of destinations is high then multi-cast is better.

The CEA-852 specification provided some support for channels that included both multi-cast and Uni-cast addressed devices through the Send List. CEA-852 did not support channels where a single device could be addressed with both multi-cast and Uni-cast on successive packets without reconfiguration of the Send List. CEA-852 did not provide a means for optimizing channel bandwidth by dynamically changing the selective forwarding rules to use multi-cast and Uni-cast in an optimal manner, such as, mapping broadcasts and group addressed packets to multi-cast addresses or by using multi-cast initiated configuration transactions to multiple devices at the same time.

The fundamental problem for CN Data packets is that the sequence numbers are used to preserve packet ordering. Each IP Uni-cast address of a device in the channel list could have its own sequence number space. Multi-cast addresses are shared by multiple devices, so synchronizing the sequence numbers across multiple destinations when switching to or from multi-cast to Uni-cast is problematic. Consequently, multi-cast addressed CN Data packets require a distinct (from the Uni-cast) sequence number space. Therefore dynamically changing the IP addressing type from Uni-cast to multi-cast and vice-versa would make it difficult to ascertain the ordering of packets from two different sequence number spaces without some corresponding modification in the Session ID numbering as well.

By modifying the session numbering rules it is possible to support some form of a dynamic hybrid channel with more optimal selective forwarding of CN Data packets using Uni-cast and multi-cast addressing. The key is the Session ID. Since the sequence numbering is restarted whenever a new session is started, packet sequence is determined by the combination of the 32 bit Session ID and the 32 bit Sequence Number. If the Session IDs are also monotonic increasing (modulo 32), then the combination forms a 64 bit packet sequence number space.

By modifying the rules for assigning new Session IDs for CN Data packets, preserving packet order in a hybrid channel is possible.

Packet ordering is not a problem for configuration packets.

7.3.2 Session ID Rules

In CEA-852.1, each Session ID value is not chosen randomly but **MUST** be monotonically increasing. The DSID and the CSID are independent and each has its own

Session ID number space. The last value of both the DSID and CSID must be preserved in nonvolatile memory so that the ordering is preserved across power cycles.

The following rules apply separable to each of the DSID and CSID:

- Each subsequent Session ID value **MUST** be greater (modulo N, where N is 2^{32}) than the last one and the delta between two successive Session ID values **MUST** be less than N/2 to avoid rollover ambiguity.
- When the Session ID rolls over, it rolls over to one not zero.
- The saved or last Session ID value **SHOULD** not be zero if it ever was non-zero.
- Reboots always start a new Session and require a new unique Session ID value.
- The Session ID sequencing **MUST** be preserved across power cycles.

The following rules apply only to the Configuration Packet Session ID (CSID):

- A CSID of zero is special. All configuration packets with a CSID value of zero are always considered by the receiving device as having a valid CSID (i.e. equivalent to the current Session ID stored by the receiving device for that sending device). A CSID of zero **MAY** be used to bootstrap a device after a power cycle or reset.
- Changing the CSID value will abort any outstanding configuration transactions initiated by the device that changed its CSID value.
- Changing the type of IP addressing (Uni-cast to/from multi-cast) does not require a change in the CSID.

The following rules apply only to the CN Data Packet Session ID (DSID):

- CN Data packets **MUST** never use a DSID value of zero. This implies that the DSID of a sending device **MUST** be incremented to some non-zero value before it **MAY** forward packets. This is done to avoid ambiguity with respect to CN Data packet order.
- A new session **MUST** be started every time the type of IP addressing used for forwarding CN Data packets to a given device changes (Uni-cast to/from multi-cast). Or in other words, during any given DSID session, a device **MUST** use either Uni-cast or multi-cast addressing of CN Data packets forwarded to another device but not both.
- To prevent gaps when the DSID changes, packets from the new DSID value **MUST** be escrowed for one channel escrow timeout period to allow any straggler packets from the previous session to arrive. This limits the rate of change of addressing type for a given destination device to once per channel escrow timeout period. The number of sessions is 2^{32} . If the Session ID changed once every 10 ms it would not wrap around for 497 days. This limitation on rate of change of Session IDs restricts the usable application of Uni-cast/multi-cast switching to low latency channels, such as, a fully switched LAN, which require only a very

small escrow timeout. But it is this application where scalability to high numbers of devices is most likely needed. As long as the reboot time is longer than one escrow time-out then the limitation on rate of change of DSID values will be satisfied. If not then packet forwarding **MUST** not resume until one escrow timeout period has expired after a reboot with its associated use of a new DSID value.

7.4 Backward and Forward Compatibility

In order to support future changes in the configuration packet formats while ensuring backwards compatibility, devices and configuration servers must not discard fixed format configuration packets solely because the length of the payload portion of the packet is longer than the format specified in a particular version of the CEA-852.1 specification. Specifically, implementations must not discard fixed format configuration packets whose length is less than or equal to the maximum UDP packet size simply because the size exceeds the format specified in the protocol for a given packet type. This applies to either UDP or TCP packets. Packets may be discarded if the length of the packet does not agree with the Data Packet Length field in the Packet Header. Implementations may limit the maximum size of segmented configuration packets in an implementation dependent manner to reflect memory.

Any new fields added to the end of the payload portion of a fixed format configuration packet or configuration data section must not change the semantics of the pre-existing fields for devices that do not support or recognize the new fields.

Moreover to further ensure future forwards compatibility, packets **MUST NOT** be discarded simply because fields marked reserved or must be zero in a particular revision of the protocol are not zero.

7.5 Vendor Specific Messages

The Vendor Code in the 852.1 Packet Headers allows for vendor-specific packets. This value **MUST** 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).

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 **MUST** be set to the unique identifier for that vendor. Vendor-specific packets that have no relationship to existing standard functions defined in this specification **MUST** use a Packet Type code in the range 0x80 to 0xFF.

Vendor Specific packets received by devices that do not understand a specific Vendor Code/Packet Type combination used by the packet **MUST** be discarded.

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.

7.6 Message Segmentation

When UDP is used, all Configuration Data Messages when too large to fit into the payload section of one UDP datagram **MUST** be broken up and sent in segments. The segments are recombined at the other end to reconstitute the original Configuration Data Message.

The underlying data in all the segments **MUST** be coherent (that is have the same DateTime) throughout the segmented transaction. If an implementation cannot preserve coherency then it **MUST** abort the segmented transaction by sending the appropriate negative acknowledgement.

When a Configuration Data Message (solicited or unsolicited) is segmented then additional acknowledgements or requests, all belonging to the same transaction, might be required depending on the mode in which the segments are sent. This is explained later in more detail.

The transaction sub-header of the configuration data packet header also contains all the information needed to send segmented messages via multiple packets by way of bits in the Transaction Flags field and the Segment ID field. As a result the 852-A segment packet type is no longer needed in CEA-852.1. The Transaction Flags field also includes all the request flags. These two consolidations simplify and normalize the use of segmented packets.

Packet segmentation **MAY** only be used for configuration packets.

The Segment ID in a Configuration Data (Response) Packet indicates the segment contained in that packet.

The Segment ID in a Request Packet indicates the desired segment that **SHOULD** be sent in response to the request.

When requesting the retransmission of a particular segment of a configuration data message, the Configuration Request sections **MAY** be omitted. The Segment ID field in the packet header indicates which segment to retransmit. The information stored in the transaction database together with the packet header enables the association of a subsequent segment request to the appropriate data without resending the original request sections.

The Packet Length field in the Configuration Data Packet header is the length of a single segment in bytes, including the bytes in the packet header. It is not the length of the configuration data message which may span multiple segments. All but the last segment must be the same size and the last segment must be less than or equal to the length of the other segments. This requirement along with the Segment Count Field in the packet header enables implementers to calculate the maximum memory required to hold all the segments.

7.7 Packet Aggregation

With packet aggregation, a single UDP/IP Packet MAY contain more than one 852.1 CN Data packet. Therefore the UDP/IP packets must be formatted in such a way to allow the extraction of the individual CN Data packets. This is referred to as packet “aggregation”. CEA-852.1 devices MUST support the reception of aggregated packets. Likewise the aggregation MUST be done in such a fashion that each CN Data packet contained within an aggregated UDP/IP packet is complete, i.e. CN packets should not cross UDP/IP packet boundaries as a result of aggregation. It is also a requirement that intermediate IP devices be capable of unbundling aggregated CN Data packets and aggregating them in a different manner before forwarding them. Packet aggregation MAY only be used with CN Data packets, not with 852.1 configuration packets. An aggregated 852.1 IP message MUST only contain CN Data packets.

All 852.1 devices MUST support packet aggregation. Figure 3 depicts more than one 852.1 packet within a single UDP datagram (packet aggregation).

Aggregation is performed by queuing CN packets to be sent for some time period or until a UDP/IP packet size is reached and then sending all the queued CN packets in a single UDP/IP packet. The maximum length of time that packets may be queued by a specific device is governed by the Device Aggregation Timeout in the Device ID and Mode Configuration Data Section for that device.

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

Figure 3 Packet Aggregation

To clarify although TCP will inherently concatenate multiple 852.1 packets, this is not what is meant by packet aggregation. Aggregation is the explicit concatenation of multiple 852.1 CN Data Packets into a single UDP packet.

7.8 Authentication

7.8.1 Introduction

CEA-852.1 provides a generic mechanism for message authentication using a Shared Secret. Whereas in CEA-852-B, the only supported authentication algorithm is MD5 [13]. Moreover, the 852-B authentication mechanism assumes a fixed digest length is added to the packet. This additional length is not included in the packet length field. This prevents using different algorithms with different length digests.

In contrast, CEA-852.1 provides support for different algorithms via an authentication extended header. The authentication header includes information about the digest length and authentication algorithm. Also the length of extended headers is explicitly included in the packet length field. MD5 and SHA2 are supported in CEA-852.1 but should only be implemented for backward compatibility reasons as these algorithms have been deprecated. New implementations must use SHA2-256 to authenticate packets.

Every device on the channel SHOULD be configured to support a minimum level of authentication or a set of allowed authentication types. Any packets that do not satisfy the required authentication types are discarded. Since the packets are not encrypted, the packet may be intercepted and the authentication information stripped from the packet and then resent. Consequently, authentication requirements must be enforced at the device such that non-authenticated packets are dropped when authentication is required.

When packet aggregation occurs, a number of 852.1 packets are concatenated into one UDP Datagram. When sending 852.1 packets over TCP, packets are also concatenated in the TCP byte stream. With authentication this means that in both cases each 852.1 packet is authenticated separately with its own Authentication Extended Header.

The authentication methods require a Shared Secret. For security reasons, the Shared Secret is not an 852.1 configuration parameter. The Shared Secret and allowed authentication type(s) are configured at the devices in a manner external to the 852.1 protocol. The Shared Secret MUST have as a minimum the appropriate number of degrees of freedom = bits (DOF) for the given authentication scheme, that is, 128 DOF for MD5-128 and 256 DOF for SHA2-256. The Shared Secret, however, is NOT limited to the minimum 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.
- CN devices MUST allow for a secure means of setting and updating device secrets.
- A CN device MUST 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 manufacturers may initially seed products with a default “Shared Secret” that may be updated later using secure network messages.

7.8.2 Authentication Procedure

In CEA-852.1, authentication consists of generating and adding the Authentication Extended Header to a packet. When employed, authentication is performed on an 852.1 packet by 852.1 packet basis. Every 852.1 packet or 852.1 segment is individually authenticated. Both CN Data and Configuration packet types are authenticated. When included, the receiving device processes the header to verify the packet. If authentication is required and authentication fails, then the packet is discarded.

1.8.2.2 Transmission Algorithm

The method for computing the digest for a transmitted packet is as follows:

- Generate the packet with all headers, including the Authentication header.
- Zero out the Digest field.

- Append the Shared Secret to the end of the packet. Typically, the digest algorithm can be chained, where the data is partitioned across multiple calls to the digest algorithm. This means that the Shared Secret does not have to be physically appended just included as the last data over which the digest is computed.
- Compute the appropriate digest over the whole packet including the Authentication header with zeroed Digest field and the Shared Secret at the end.
- Copy computed digest into the Digest field.
- Truncate packet to remove the Shared Secret.
- Send the packet.

7.8.2.2 Reception Algorithm

The method for authenticating a received packet is as follows:

- Copy out the received Digest field.
- Zero out the Digest field.
- Append the Shared Secret to the end of the packet. Typically, the digest algorithm can be chained, where the data is partitioned across multiple calls to the digest algorithm. This means that the Shared Secret does not have to be physically appended just included as the last data over which the digest is computed.
- Compute the digest over the whole packet including the Authentication header with the zeroed out Digest field and the Shared Secret at the end.
- If the computed digest matches the copy of the received digest then the packet is considered authentic otherwise it is dropped.

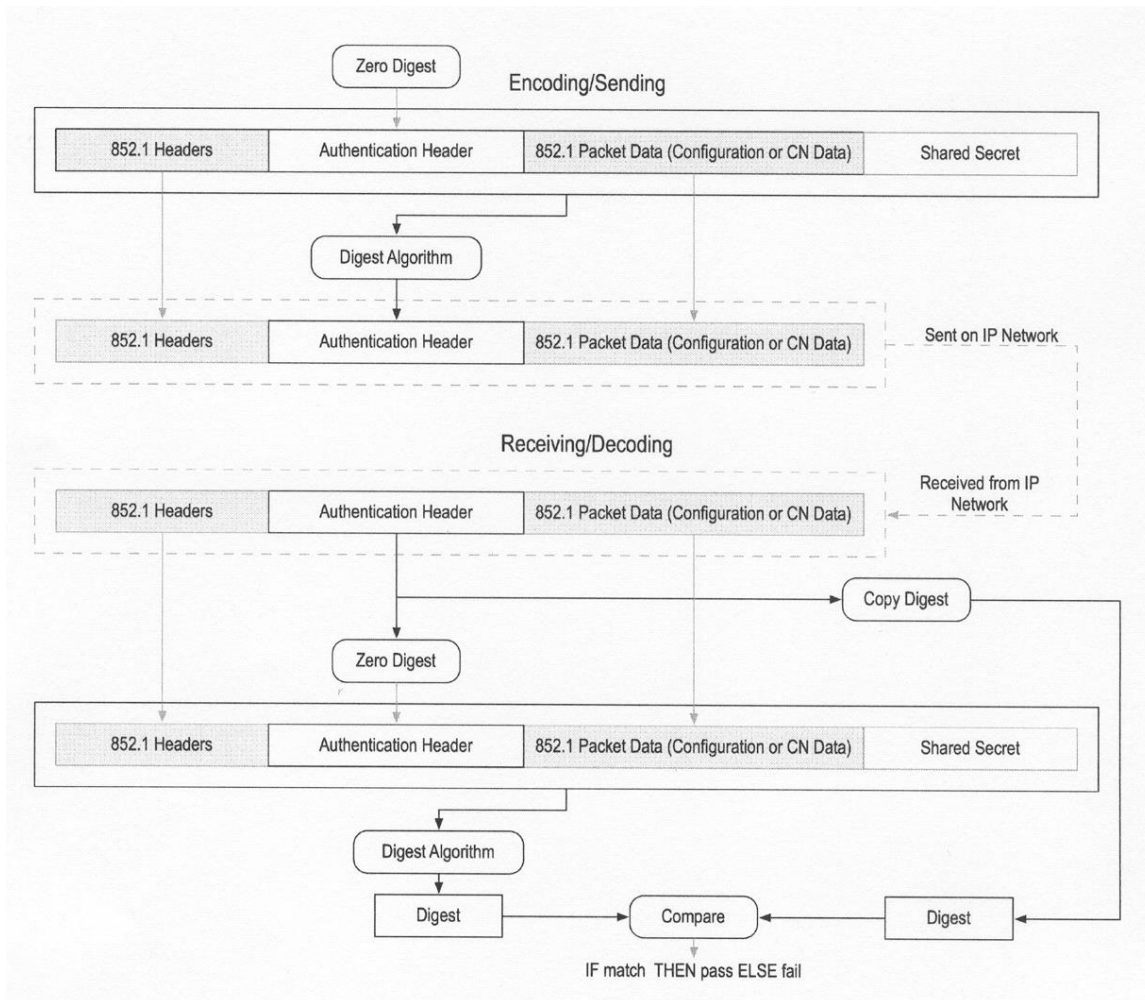


Figure 4 Authentication Procedure

7.9 NAT Support

7.9.1 Introduction

NAT refers to the technique of IPv4 Network Address Translation. Presented here is a method for managing NAT with 852.1. This approach will work with either Normal or Manual mode and is compatible with the dynamic IP and distributed configuration approaches. However, it shall be pointed out, that this approach is independent of the latter and stands on its own. This approach does not require that the configuration server be online and will allow interoperability between nodes that are configured in either manual or normal mode. It reduces the use of extended headers and provides a consistent view of the IP address that is used as the unique identifier for an 852.1 device.

This NAT approach uses the terms Public, Private, External, and Internal to describe IP addresses. Public follows the standard definition, that is, an IPv4 address that is directly

accessible by other IP devices. These are typically used on the WAN side of a router. Private follows the standard definition, that is, those few IP address ranges that are on the LAN side of a router. Private addresses cannot be accessed directly by nodes on the WAN side of the router.

An external address could be either a Public or Private IP address depending on the context. The method presented here provides a mechanism by which all packets will include sufficient information to uniquely determine the sending device's External IP address and port and thereby match the packet to a single channel member.

If all the 852.1 devices on the same 852.1 channel are on the same LAN, that is, they can all access each other using only Private IP addresses (even if they are on different domains or subnets) then both the External and Internal IP Address **MUST** be set to the Private IP address of a given device.

Otherwise, when all the 852.1 devices on the same 852.1 channel do not share the same LAN;

- If an 852.1 device has a Public IP addresses, then both the External and Internal IP addresses **MUST** be set to the Public IP address of that device.
- If an 852.1 device is behind a NAT router then the External IP address for that device **MUST** be set to the Public IP address (WAN side) of the NAT router. The Internal IP address **MUST** be set to the Private IP address of the device (LAN side). The NAT support flag **MUST** be set for that 852.1 device. The External port **MUST** be set to the Public IP port that is mapped by the NAT router to the Private IP Host:Port of a given 852.1 device. Multiple devices behind the same NAT router **MUST** have unique NAT router ports and the NAT router **MUST** be configured with appropriate port forwarding rules for each of these ports.

7.9.2 Description

This approach depends on the following configuration information:

- A NAT support flag in the Device Operational Mode field.
- The Internal IP address in the registration packet is the static local IP address of the device if applicable.
- The External and Internal IP Host:Port addresses for each device in the Channel Membership Packet. The Internal Host:Port is needed so that devices behind the same NAT router can bootstrap their mutual communications. It might not be possible for them to use the External IP Host:Port only.
- The two standard NAT extended headers might also be needed.

7.9.3 Standard NAT Extended Headers

The NAT extended header is typically only needed to bootstrap a device onto an 852.1 channel. Once all the devices have been assigned a Device ID (DID), communication with other devices may be disambiguated via the Source DID (SDID) and Destination

DID (DDID) contained in the 852.1 packet header. The headers are reproduced here from Section 11.6.

Basic NAT Extended Header			
Byte0	Byte1	Byte2	Byte3
Header Type	Header Size	External NAT IP Port	
External NAT IP Address			

Table 1

The Header Type is 0x01.

The Header Size is 2.

The External NAT IP Port is the NAT destination port on the WAN side of a NAT router for the device sending the packet. This also serves as unique identifier of the source of a packet for NAT routers that do not have static symmetric mapping of NAT source and NAT destination ports. This avoids ambiguity in determining the source Device when multiple 852.1 devices are behind the same NAT router and the NAT router dynamically assigns source ports in the IP packet header. A device receiving a packet from another device that is behind a NAT router SHOULD use the External NAT IP Port Field instead of the IP source port to determine the IP Host:Port identifier for the sending device. This is only needed for the registration of a device with a Configuration Server. Once the Device is registered and has received its Device ID (DID) then the SDID in the packet header can be used to uniquely identify the source of a packet regardless of the Port included in the IP Source of the packet.

Enhanced NAT Extended Header			
Byte0	Byte1	Byte2	Byte3
Header Type	Header Size	Internal NAT IP Port	
Internal NAT IP Address			

Table 2

The Header Type is 0x02.

The Header Size is 2.

The Internal NAT IP Port is the source port on LAN side of a NAT router for the device sending the packet. This header MAY be used for debugging purposes such as when it is

desirable to verify or discover how the NAT router is mapping ports. This might also be the case for an automated configuration scheme that dynamically determines the NAT status of a device. If used, the Enhanced NAT Extended Header **MUST** always be preceded by the Basic NAT Extended Header.

7.9.4 Algorithm

7.9.4.1 Assumptions

Each device has an External IP Host:Port and an Internal IP Host:Port in the channel list.

Incoming packets from devices on the WAN (public) side to devices on the LAN (private) side of all NAT routers have a static destination port mapping in the NAT router.

Outgoing packets from devices on the LAN (private) side to devices on the WAN (public) side of all NAT routers might use a different source port mapping from the destination port for a given local device and the source port for a given local device might vary depending on the destination.

7.9.4.2 IP Address Assignment

IF: the device is behind a NAT router AND NOT all devices share the same LAN
THEN:

- The NAT bit flag is set.

- The device **MAY** send NAT headers.

- The device's Internal IP Host:Port is set to its Private IP Host:Port address on the local LAN behind the NAT router.

- The device's External IP Host is set to the Public (WAN) IP Host of the NAT router.

- The device's External IP Port is set to the destination Port mapped by its NAT router to that device.

ELSE: (the device is not behind a NAT router or all devices share the same LAN)

- The NAT bit flag is cleared.

- The device **MAY** not send NAT headers.

- Any other device **MAY** assume that it can send packets to this device using this device's External IP Host:Port.

- Any other device **MAY** assume that it can uniquely resolve this device as the source from the IP header of this device's packets (which are equal to its External IP Host:Port).

IF: all the devices on a channel share the same LAN THEN:

FOR EACH device: