GODDARD TECHNICAL STANDARD

GSFC-STD-8012A

Goddard Space Flight Center
Greenbelt, MD 20771

Approved: 06-18-2026
Revalidation Date: 06-18-2031
Superseding GSFC-STD-8012

BULK SECURITY STANDARD FOR SPACECRAFT COMMUNICATION

THIS STANDARD HAS BEEN REVIEWED FOR EXPORT CONTROL RESTRICTIONS;
APPROVED FOR PUBLIC RELEASE
DISTRIBUTION IS UNLIMITED

Prepared By:

Victor Sank (affiliate) Digitally signed by Victor Sank (affiliate) Date: 2026.06.15 11:49:05 -04'00'

Victor Sank, CTI
Communications SME
NASA Goddard Space Flight Center

Approved By:

Matthew Ritsko Digitally signed by Matthew Ritsko Date: 2026.06.18 10:28:05 -04'00'

Matthew W. Ritsko
Deputy Director, Engineering and Technology Directorate
NASA Goddard Space Flight Center

David Reth Digitally signed by David Reth Date: 2026.06.18 10:40:49 -04'00'

David A. Reth
Director, Safety and Mission Assurance
NASA Goddard Space Flight Center

NASA GODDARD SPACE FLIGHT CENTER
Greenbelt, Maryland 20771

DOCUMENT HISTORY LOG

Status	Document Revision	Approval Date	Description
Baseline	-	08-08-2024	Initial Release
Revision	A	06-18-2026	Add Tail for finding coded length; Increase MAC from 64 to 128 bits; Update numbers influenced by increased MAC size and Tail; Limit CLTU to one transfer frame; Several editorial corrections; Pattern of CSM user selectable; Remove/caution use of NRZ-M;

FOREWORD

This standard is published by the Goddard Space Flight Center (GSFC) to provide uniform engineering and technical requirements for processes, procedures, practices, and methods that have been endorsed as standard for NASA programs and projects, including requirements for selection, application, and design criteria of an item.

This standard describes a method named Bulk security where the entire data frame is secured. At the sending end (may be ground or space), this standard first applies authenticated encryption and then applies an error correcting code, so the spacecraft first applies error correction decoding and then applies authenticated decryption. The encryption and authentication are similar to the methods used by many GSFC missions but establishes a novel standard where there was none. Early chapters describe the many components of security and error correcting coding. Chapter 15 describes the assembly of the security and coding into the data units that are modulated and transmitted to the satellite. It is expected that a project will use only one of the cases shown there for a given mission phase. When the word "encryption" is used, authenticated-encryption is often implied.

Requests for information, corrections, or additions to this standard should be submitted via "Contact Us" on the GSFC Technical Standards website at http://standards.gsfc.nasa.gov.

Robert Sticka
Digitally signed by Robert Sticka
Date: 2026.06.15 12:08:36 -04'00'

Robert A. Sticka
Technical Standard Program Manager
NASA Goddard Space Flight Center

ACKNOWLEDGEMENT

This standard could not have been completed without the two co-authors, Brent Andres, GSFC, and Greg J. Kazz, JPL.

TABLE OF CONTENTS

Section **Page**

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

1. SCOPE

This standard covers only bulk security for the command link and does not cover the telemetry link. It includes the use of (heritage) Communications Link Transmission Units (CLTUs) with the Bose–Chaudhuri–Hocquenghem (BCH) code internal to the CLTU, and Telecommand (TC), Advanced Orbiting System (AOS), and Unified Space Link Protocol (USLP) transfer frames.

This Standard does not cover details of the cipher (Advanced Encryption Standard (AES)-256 is specified but could be changed in the future) and does not cover key management. Missions may employ post-quantum cryptographic (PQC) algorithms.

In addition to the BCH code, the Consultative Committee for Space Data Systems (CCSDS) now has forward link Low Density Parity Check (LDPC) coding that is included in this standard.

This standard aligns with NASA-STD-1006 (Space System Protection Standard). This standard covers the use of error correction coding applied after the security, but a project can choose to not have error corrections coding if the link analysis indicates a very low error rate.

1.1 Purpose

The purpose of this document is to define a bulk security (encryption and authentication) standard for NASA missions, perform the error correction/detection decoding prior to processing the security at the receive end (the spacecraft), allow the use of heritage transfer frames, update from the Counter with Cipher Block Chaining Mode (CCM) security structure to Galois Counter Mode (GCM), and add the option of LDPC codes. In addition, for the near term (~15 years) allow systems to use a heritage CLTU to avoid the need for changing both existing ground software that generates the CLTU and spacecraft software/firmware that processes the CLTU. In alignment with recent CCSDS development, the term CLTU has been expanded and is further expanded here to what we call a secured CLTU (Secured-CLTU).

This document describes the various fields needed for bulk security and defines where those fields are placed and what information they contain.

1.2 Background

Near-Earth projects are defined as those that are at or less than 2 million kilometers from the Earth (Category A). For projects in the Earth/Moon environment, the current non-standard use of bulk security applied to a CLTU causes the internal BCH code to become useless but does not degrade the telecom performance of the link. Currently the BCH code used in Earth/Moon applications is generally used for error detection, not correction, and the authentication provides better error detection than the BCH code. In addition, for near-Earth orbiters there is sufficient, uplink margin for near error-free service.

For missions at L1 or L2 (approximately 1.5 million kilometers from the Earth), however, the link may not be near error-free and applying an error correcting code prior to security at the receiver would be beneficial to missions. Figure 1 is a quick overview of the data structure transmitted to the spacecraft that is the result of this standard. Figure 6 shows the function flow.

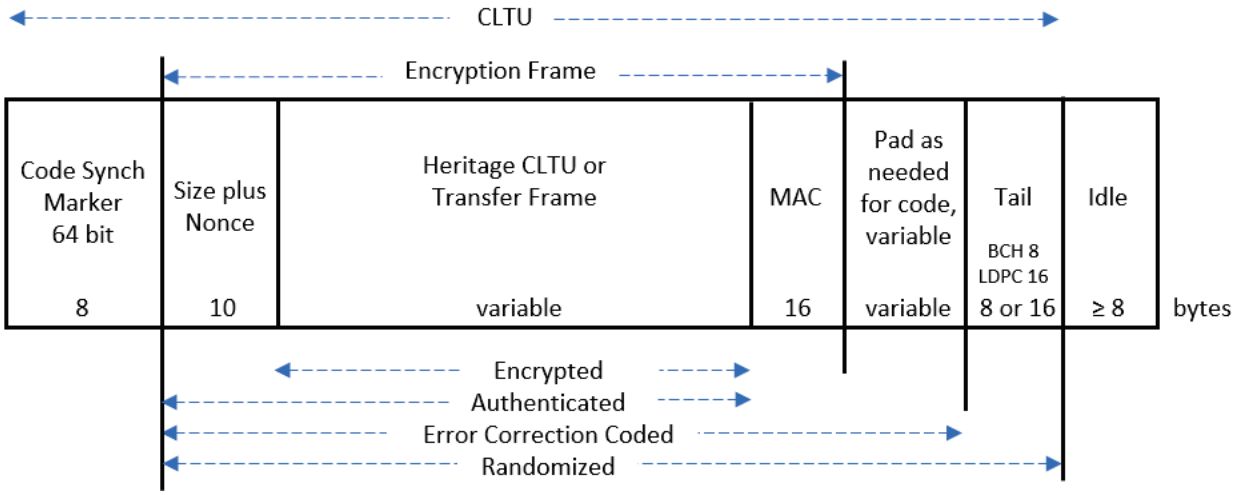

Figure 1. Bulk Secured and Coded Data

As a practical solution, several near-Earth satellites have been using a Bulk security method with the CCM protocol in legacy systems which, after removing the security in the receiver, results in an output of a plaintext (PT) heritage CLTU that is processed by existing command and data handling (C&DH) systems. However, without a NASA/GSFC standard for applying the CCM the implementation differed from satellite to satellite. Prior to the use of security, for near-Earth projects, link errors were reported in the Communication Link Control Word (CLCW) in a timely manner due to the short near-Earth ranges, and a command was resent. With security, authentication performs a similar function of reporting a failure if there is a channel error. For future missions where the link margin is limited, including those at L1 or L2, it makes sense to put the Radio Frequency (RF) link error control correction/detection code prior to the spacecraft security function. This allows the error correction mode of the BCH code. It also makes sense to upgrade to the more current GCM.

The CCSDS has defined an alternative method, i.e., the Space Data Link Security Protocol (SDLS) [6] which also places the BCH error detecting/correcting code prior to the decryption and authentication function, however only the data field portion of the transfer frame is encrypted. The CCSDS design requires projects to change ground and spacecraft Heritage CLTU processing, which several projects find undesirable. This standard maintains the Heritage CLTU.

CCSDS now has forward link LDPC coding that performs better than BCH coding. The LDPC is more complicated to decode, so there is still a place for using the BCH code if it is applied after security at the sending end. With bulk security there is no need to change the heritage CLTU processing software/firmware and adding either BCH or LDPC coding after the bulk security at the sending end makes the added code useful. CCSDS limits the length of a TC transfer frame in the CLTU to a maximum of 1024 bytes. CCSDS allows multiple transfer frames in a CLTU, but this is rarely if ever used and for this standard there is only one transfer frame per CLTU (see section 7).

Errors that are first corrected by the on-board decoder can be used as an aid in distinguishing between physical link errors and malicious ones, i.e., those caused by a bad actor. Spacecraft

telemetry will inform the operators if a command failure is due to a channel error (decoding failure) or an authentication (security) failure. If all channel errors are corrected, and there is an authentication failure, an indication of a possible attempted security incident has occurred. Telemetry sent to the ground would indicate successful decoding as well as the authentication failure. However, this method is not 100% foolproof. False positives may occur as the following scenario illustrates. The BCH undetected error rate is not zero so there can occasionally be a channel error that is missed by the BCH and then detected by the authentication function as what appears to be a security error. Section 7.1.1 contains additional comments related to decoding and/or decryption failures.

This document does not cover telemetry, but if the project wanted to secure their telemetry data and still have virtual channel sorting at the ground station available, the project could secure the instrument data in the packets or use techniques similar to those in Appendix F.

2. APPLICABLE DOCUMENTS

Projects shall document tailoring/adoption decisions in the Project Protection Plan and complete the NASA-STD-1006A Appendix A compliance matrix.

The CCSDS documents listed are current at the time of this writing, but the most current version should be used. The right most value is the version number.

1. CCSDS 201.0-B-3, Telecommand, June 2000 now Silver, replaced by 231.0-B-4
2. CCSDS 231.0-B-4, TC Synch and Channel Coding July 2021.pdf
    a. Contains BCH (63,56), LDPC, CLTU, Randomize, PLOP
    b. Two binary LDPC codes are specified, with codeword lengths (n=128, k=64) and (n=512, k=256)
3. CCSDS 232.0-B-4 TC Space Data Link Protocol Oct 2021.pdf; covers Transfer Frame, FECF, CLCW
4. CCSDS 232.1-B-2 Communications Operations Procedure 1; Telecommand Go-Back-N ARQ protocol
5. CCSDS 912.1-B-5 SPACE LINK EXTENSION (SLE) FORWARD CLTU Aug 2016.pdf
6. CCSDS 355.0-B-2 SPACE DATA LINK Security (SDLS) Protocol Sept 2015.pdf
7. CCSDS 732.0-B-4 e1 AOS Space Protocol Sept 2015 2018.pdf
8. CCSDS 732.1-B-2 Unified Space Data Link Protocol USLP Oct 2021.pdf
9. NIST SP 800-38C CCM May 2004; structure that uses a cipher for authentication and encryption
10. NIST SP 800-38D GCM Nov 2007; structure that uses a cipher for authentication and encryption
11. NIST FIPS 197 AES Nov 26, 2001; cipher that is recommended for use in the above structures.

Note: The 63,56 BCH code discussed here is defined in CCSDS 231.0-B-4 which includes a fill bit so the codeword as used is 64 bits in length. This code is referred to as a 64-bit (code symbol) codeword in this document. Details of the LDPC codes are covered in ref [2].

3. ACRONYMS AND DEFINITIONS

3.1 Acronyms and Abbreviations

AAD	Additional Authenticated Data
AES	Advanced Encryption Standard
AOS	Advanced Orbiting System
ARQ	Automatic Request for Retransmission
ASM	Attached Synchronization Marker
BCH	Bose–Chaudhuri–Hocquenghem
C&DH	Command and Data Handling
CCM	Counter with Cipher Block Chaining Mode
CCSDS	Consultative Committee for Space Data Systems
CLCW	Communication Link Control Word
CLTU	Communication (Command) Link Transmission Unit
COP-1	Communications Operating Procedure-1
CRC	Cyclic Redundancy Check
CSM	Code Synchronization Marker
CT	Ciphertext (cyphertext)
DSN	Deep Space Network
ECC	Error Correction Code
FEC	Forward Error Correction
FARM-1	Frame Acceptance and Reporting Mechanism
FECF	Frame Error Control Field
FOP	Frame Operation Procedure
GCM	Galois Counter Mode
GSFC	Goddard Space Flight Center
HCLTU	Heritage CLTU
ICB	Initial Counter Block
IV	Initialization Vector
LEF	Large Encryption Frame
LDPC	Low Density Parity Check
MAC	Message Authentication Code
MAC’	Message Authentication Code Prime
MOC	Mission Operation Center
PT	plaintext
RF	Radio Frequency
SA	Security Association
SCID	Spacecraft ID
SDLS	Space Data Link Security
SEF	Small Encryption Frame
SLE	Space Link Extension
TC	TeleCommand
Transfer Frame	CCSDS Transfer Frame
USLP	Unified Space Link Protocol
VCID	Virtual Channel Identifier

3.2 Definitions

Advanced Encryption Standard	Used as the cipher in this standard.
Block	A group of 128 bits associated with the cipher. (Also refers to an error correcting code of a fixed size.)
Bose-Chaudhuri-Hocquenghem	Binary error correction/detection block code with n,k = 63,56 bits.
Cipher	The algorithm used to encrypt the data (also spelled cypher).
CLTU	Secured and coded transfer frame, or for the heritage case a secured and coded heritage CLTU, with a code synch marker and Tail. When several SEFs are used, the combination of SEFs is the CLTU
Coded Encryption Frame	An Encryption Frame with error correction coding.
Communication Link Transmission Unit	A structure that contains the command transfer frame (CLTU).
Encode	The process of error correction coding. Encode and encrypt can get confusing. This document will generally use “code” rather than “encode” to avoid that confusion.
Encryption Frame	Data frame that contains encrypted data and unencrypted Size, Nonce, and MAC fields in the order Size, Nonce, CT and MAC fields. This standard defines two approaches, one with Small Encryption Frames and the other with Large Encryption Frames. The Small Encryption Frames are roughly equivalent to the AES CCM Block used on several previous satellites.
Fill	Same as padding. Used when additional bits or bytes are needed to complete a codeword, 0x5 used. See CCSDS 201.0-B-Telecommand, or 231.0-B-, or 232.0-B-.
Frame Error Control Field	An error check, not an error correction method.
Idle	Bits used between frames or between commands to maintain bit synchronization 010101… is commonly used. Proximity-1 links use 352EF853.
Initial Counter Block (ICB)	Used by the cipher in counter mode.

Initialization Vector	Used to generate the ICB; created from the nonce plus a static portion.
k	Size of message in an error correcting/detecting codeword.
Low Density Parity Check (LDPC)	Binary block code used for error correction.
Managed parameter	A value or set of values chosen by the project that is likely to apply for the life of the mission or a phase of the mission.
Message Authentication Code Prime	Calculation of the MAC done in the spacecraft to compare with the received MAC.
n	Size of codeword including the parity (k is the size of the message portion of a codeword, $k < n$. Also used to indicate the number of small Encryption Frames.
Nonce	A changing, usually incrementing, value that is used only once within a specified context (different for each Encryption Frame).
Padding	Also referred to as Fill bits or bytes (see Fill).
Plaintext	Text which has not been encrypted.
Secured-CLTU	A CLTU applied with bulk security defined in this Standard
Security function	Encryption and authentication or their removal
Tail	8 bytes of data at the end of a CLTU to indicate its end. When a heritage CLTU is used there are two Tails, one in the heritage CLTU and another following the padding (when needed) of an encoded encryption frame.
Unified Space Link Protocol	Unifies up (forward) and down (return) link into a common structure.

3.3 Nomenclature

The following conventions apply through this Recommended Standard:
– the words ‘shall’ and ‘must’ imply a binding and verifiable specification;
– the word ‘should’ implies an optional, but desirable, specification;
– the word ‘may’ implies an optional specification;
– the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

4. CCSDS

The CCSDS defines the SDLS in which only the data field of the transfer frame is encrypted. There are three advantages of CCSDS SDLS over the legacy bulk encryption method. First, at the receive end, the BCH or LDPC error correction/detection decoding is applied prior to processing the decryption and authentication of the transfer frame data field in the CLTU. Second, by applying these two functions in the correct order (error correct/detect then decryption), operators can distinguish between channel errors and malicious errors. The third advantage is that since only the transfer frame data field is encrypted, the CLTU Start Sequence, the transfer frame primary header, and the CLTU Tail are in the clear. (When a transfer frame is used without a CLTU, the primary header is in the clear.) Having a plaintext header has value for international cross support where a receiving node (International Space Station for example) can read the spacecraft ID (SCID) and virtual channel identifier (VCID) in the transfer frame primary header to route the transfer frame to the appropriate international space agencies' equipment. SDLS also allows some frames to be transmitted in plaintext based on the VCID. When international interoperability is required, the CCSDS SDLS is recommended.

However, leaving the transfer frame primary header in the clear has disadvantages as well. An intruder may learn a lot, not just from the traffic analysis but from the SCID, which may compromise space agency identifying information by exposing traffic to a particular satellite or equipment at an international node. The bulk security standard encrypts the SCID and all other fields in the transfer frame although an option exists within this standard to leave the SCID/VCID un-encrypted. This standard is not aimed at international cross support, although it can be used internationally and there are options in appendix F that allows for international cross support.

The CCSDS method has another disadvantage in that it adds security fields in the transfer frame, requiring the spacecraft C&DH software/firmware to be changed. One of the fields contains the unencrypted key index, exposing which key is in use. The bulk standard requires an encrypted command to change the key via its index and allows the security to be removed prior to sending a heritage PT CLTU or transfer frame to the C&DH, hence no programming changes are required in the C&DH.

This Bulk Security Standard preserves compliance with the Communications Operating Procedure 1 (COP-1) in either sequence-controlled or expedited mode including securing COP-1 control commands (TC frame type BC) which the SDLS procedure does not. Additional comments on COP-1 are in appendix C.

Many of the NASA satellites do not require CCSDS international cross support and inter-operability. Furthermore, the concept of international cross support may be misunderstood. Both legacy and standard bulk security can also be used for international cross support since CCSDS ground stations do not interpret the command bytes, rather they just modulate and transmit the commands. The command data contained within the CLTUs is transparent to the ground stations. A bulk secured CLTU or transfer frame can be sent to a foreign ground station and then sent to the intended NASA satellite, so bulk security per se allows some level of international cross support. Even though such support is technically feasible, some US government agencies prefer not to use this capability. When Bulk security is applied at the Mission Operation Center (MOC),

the service provider (Near-Earth Network, Deep Space Network (DSN), commercial, international) that radiates the command, cannot see the CLTU or transfer frame. This does not cause any difficulty since the service provider receives an encapsulated Space Link Extension (SLE) Protocol Data Unit and transmits that data without examining it, [5] section 2.1.

5. NUMBERING OF BITS IN A FIELD

In this standard, (and following the convention used by CCSDS) for a field of N bits, the most significant bit is on the left, is numbered 0 and is transmitted first. The Nth bit on the right is numbered N-1 and is transmitted last. See Figure 2.

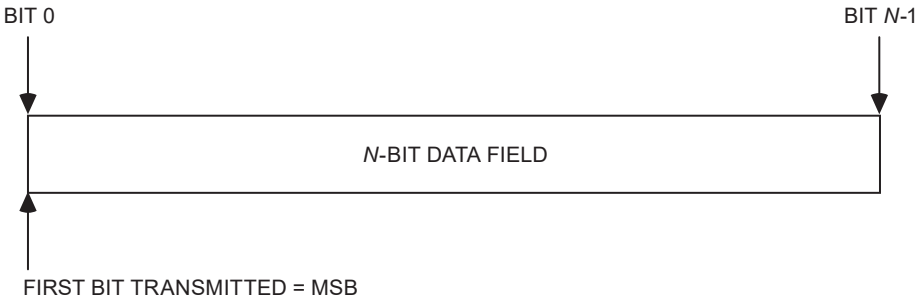

Figure 2. Bit Numbering Convention

6. MANAGED PARAMETERS

Managed parameters are values that are chosen by a mission to configure a given protocol implementation. A mission often selects these parameters and uses them throughout the mission lifetime or mission phase. Therefore, a mission may have more than one set of managed parameters. For security at the transfer frame layer (data link), the CCSDS (but not this standard) uses the term “Security Association” (SA) which has an associated set of managed parameters per VCID. Once selected, they are fixed for a given SA. Analogously, in this standard we refer to various cases, and once a case is chosen, only that case is used for the mission or mission phase.

In this standard some but not all of the managed parameters are:

1. Nonce Field Length (how the size field in the nonce is used).
2. Transfer frame maximum length (This drives CLTU size limitation).
3. Encrypted block length (e.g., cipher block length, currently 128 bits with AES).
4. Number of Encryption Frames used (the use of a single or multiple Encryption Frames).
5. Coding (BCH or LDPC codes, or no error correction code).
6. Specific LDPC Encoding (either LDPC (128, 64) or (512, 256) code).
7. Indicate use of the optional Frame Error Control Field (OCF) and/or the Cyclic Redundancy Check (CRC). (This is an optional trailer field in the transfer frame when

using a heritage CLTU but mandatory for transfer frames when a heritage CLTU is not used.)
8. Data Link layer Automatic Request for Retransmission (ARQ) Protocol (COP-1 used or not used).
9. Data format NRZ-L or NRZ-M. NRZ-L recommended to avoid error doubling.
10. TF primary header included as additional authenticated data (see appendix F).

Projects may implement additional or stronger security mechanisms (e.g., enhanced authentication layers, encrypted telemetry paths, additional network-layer protections) as long as the Bulk Security field structure and the order of operations defined in this standard remain unchanged.

The figures in later sections are drawn based on a single CCSDS CLTU maximum transfer frame length of 1024-bytes, but missions may choose to use the USLP recommendation that limits the CCSDS transfer frame length to 65 K bytes (65,536 bytes). In that case, the maximum length will be a managed parameter, and missions may set a maximum length ≤ 65,536 bytes.

7. DELIMITING THE CLTU

For this standard, there shall be only one transfer frame per CLTU.

A tail is a convenient way to delimit the CLTU i.e. find the end of a variable length secured and coded CLTU. An alternate and more complicated method is to use the Length field in the command transfer frame.

At the sending end, a Tail shall be applied immediately following the Code Padding which immediately follows the Encryption Frame.

When the size of the Encryption Frame is such that no padding is needed for the coding, the tail shall be applied immediately following the MAC (last field in the Encryption Frame, see Figure 1)

When BCH coding is used after the security, the Tail pattern shall be
0xC5C5C5C5C5C5C579.
When LDPC code is used after the security, the Tail pattern shall be the 128-bit pattern shown in CSDS 231.0-B- section 5.2.4.2 (spaces between characters are to be omitted), in the de-randomized form.

Immediately following the Tail, there shall be at least 8 bytes of idle, 01010101…

Note: It is recommended that this idle be part of the command, rather than depending on the ground station. The Tail should terminate the CLTU or CLTU segment when small encryption frames are used, but if the spacecraft is looking for the idle pattern it must exist.

7.1 Heritage CLTU

A heritage CLTU is composed of a 16-bit Synchronization (synch) Marker (called a Start Sequence), a BCH coded transfer frame, and a Tail Sequence to indicate the end of the CLTU. The term CLTU has been expanded in recent years to include an optional LDPC coding in lieu of BCH coding and includes other changes. For that reason and for changes related to Bulk security, we must distinguish between a heritage CLTU (HCLTU) and the current versions. Command transfer frames are variable in length and have a 10-bit length field. Section 4.1.2.7.2 of CCSDS 232.0-B-4 specifies that the 10-bit field shall contain a length count *C* which equals one fewer than the total octets in the transfer frame. The CLTU will have greater length than the transfer frame due to the addition of the Start Sequence, BCH or other coding and the Tail Sequence. The Tail Sequence is required for the heritage CLTU for several reasons. The transfer frame header is unreliable prior to the BCH decoding, decades ago when the heritage CLTU was created it was impractical to read the contents of the CLTU while simultaneously delimiting it, and it was undesirable to cross processing layers. CCSDS recommends a Tail sequence that is designed to fail the BCH decoding as the indicator of the end of the CLTU. This also signaled the receiving side to start searching for the next Start Sequence. If the BCH decoding of codewords prior to the Tail finds no errors, a decoded CLTU is sent to the next processing layer, i.e., to derandomize the transfer frame and process the command. Note that when LDPC code is used, CCSDS requires that the de-randomizing is done first, prior to the decoding, instead of the incorrect order of decoding and then de-randomizing, that is used in processing the heritage CLTU. With the LDPC codes, CCSDS uses the tail sequence to delimit the coded variable length transfer frame.

With the heritage CLTU structures and no security applied, if there is a decoder failure, the logic interprets it as the tail sequence and begins searching for the next Start Sequence (synchronization marker). If the decode failure is internal to the CLTU the current command is lost. If the lost command was a hardware command the MOC is informed by telemetry and the command must be re-sent. If the lost command was a software command and COP-1 is being used, the command would be re-transmitted automatically by the transmission side of the COP-1 protocol (the Frame Operation Procedure, FOP-1). For near-Earth missions with a short signal transit time this doesn't present a significant problem but for deep space and missions at L1 or L2, for example, this long latency is typically unacceptable to mission operations. For deep space it is common to send commands twice and the COP ignores the second if the first was received properly.

With a heritage CLTU and previous non standardized bulk security, the security is processed prior to the point in time where the error correction/detection code has a chance to detect an error. If there was a bit error, the authentication fails, providing the same error detection function which BCH would have provided. Part of the purpose of this standard is to specify an error correction code (ECC) that is applied outside of the heritage CLTU and is decoded prior to the decryption and authentication function to allow for error correction and provide coding gain.

With a bulk encrypted and secured and coded heritage CLTU, the Start and Tail Sequences do not need to be encrypted, but we have chosen to encrypt them so that after decryption, what is delivered to the C&DH is the heritage CLTU. Hence there is no need to change existing ground and spacecraft CLTU processing software, in the near term. The length field in the CLTU

transfer frame is encrypted and hence when the decoding and decrypting is being done on the spacecraft, some other method is needed for delimiting the data stream to be decoded and decrypted. Further on in this standard, an unencrypted Code Synchronization Marker (CSM) and a second Tail field are defined to meet this need.

This size field that follows the synchronization marker is used by several vendors to delimit the received data, but there is no published standard which specifies this, defines the length of this field, or defines the unit that the value refers to (bytes, 8-byte units, bits). Defining such a standard is part of the purpose of this document.

7.1.1 Partial Frame Loss and Flushing

With respect to this standard, when a codeword is lost or there is a security failure, all following bits are dropped up to the point where a synchronization marker is found. At that synchronization marker, after decoding and security processing, the PT is passed to the protocol layer as it normally would be. That layer will assemble all of the PT pieces and search for a CCSDS frame marker (EB90 Start Sequence or Attached Synchronization Marker (ASM)). The CLTU with the failed codeword will be lost but following CLTUs will be found. COP-1, JPL COP or some other protocol logic will then come into play.

See CCSDS 232.0-B-4, section 4.4.9. Where CCSDS refers to the channel coding sublayer, with respect to this standard, it should be interpreted as the combination of the channel coding and security sublayer. The loss shall be reported by the layer where it occurs, but it is at the protocol layer where the Transfer Frame that is lost gets identified. The concept of this standard is that the decoding layer delivers to the security layer and the security layer delivers PT to the Protocol layer and transfer frames (and possible lost transfer frames) are identified.

7.2 CLTU

In the CCSDS TC Standard, reference 2, the term CLTU is generalized to include a CLTU other than the heritage version. When referring to the heritage CLTU, the CCSDS book specifies a CLTU “when BCH Coding Is Used**”**. When not referring to a heritage CLTU the CCSDS book specifies a CLTU “when LDPC Coding Is Used”. In this document we use similar terminology but define a CLTU as a CSM, coded and secured data, and a Tail where at the sending end security is applied prior to applying coding. The term “secured CLTU” is sometimes used but is the same as CLTU defined here. Further detail of the CLTU follows in later sections.

8. THE CIPHER

Projects may implement additional or stronger security mechanisms as long as the Bulk Security field structure and the order of operations defined in this standard remain unchanged.”

8.1 AES Cipher

The cipher used shall be the AES cipher with a 256-bit key defined in NIST FIPS 197 AES Nov 26, 2001. Much of the information in this section is not intended to be specific to the AES cipher but is included to help the user understand parts of the NIST 38D GCM specification.

Implementations of AES-256/GCM for command stack protection shall be provided by cryptographic modules validated to FIPS 140-3 (or FIPS 140-2), Level 1 or higher. Projects shall record certificate IDs and the NASA-STD-1006A compliance matrix in their Project Protection Plan.

Note: the security structure in this document is expected to be malleable and change as ciphers are improved. Public key – private key ciphers depend on finding the two very large prime factors of an even larger composite number N, where N = prime1 x prime2. The AES cipher does not function this way and is expected to be less susceptible to code breaking techniques employing quantum computing but future quantum computing still must be considered as a threat.

Note: cipher = cypher

Encryption Process

Each group of up to 128 bits is called a block. The cipher in the NIST 38C and 38D document is not applied directly to the PT data but instead it encrypts 128 bits (16 bytes) that are based on a parameter called an Initial Counter Block (ICB) [NIST 38 D]. Figure 3 is an example of a Counter Block. Within an Encryption Frame, the nonce is fixed but the Encryption Block Counter is incremented for each new set of 128-bits to be encrypted. Each new set of encrypted bits are then applied to (XORed with) each 128-bit length pieces of the PT data to encrypt it. Since the data is of arbitrary length, when it is broken into pieces of length 128 the last piece may be less than 128 bits. To encrypt that remainder, a final counter block of 128 bits is encrypted but only the most significant bits of that block, equal in length of the remainder, are used to encrypt that segment. The result is that the cyphertext (CT) is the same length as the PT.

The GCM standard limits the length of an input string of PT to $\leq 2^{39}$ -256 bits but for this standard the maximum string length is limited to the maximum length CLTU which is much shorter (CCSDS allows a heritage CLTU to contain several transfer frames but for this standard there is only one transfer frame per CLTU). Large CLTUs may be broken into several pieces and secured in several Small Encryption Frames (see section 15) or may be processed as a single Large Encryption frame. When using Small Encryption Frames, which limits the size of the PT data to be encrypted in a single Encryption Frame, the limit has been chosen as 1920 bits (240 bytes, 15 blocks). The frame that contains this data will also include other fields that are not encrypted. As explained in the previous paragraph, the length of PT in a frame does not need to be a multiple of 128 bits, (16 bytes).

8.1.1 Cyphertext Length

The size of the CT is the same as the size of the PT. This may be considered “obvious” since the encryption of the PT is simply an exclusive OR with an equal length set of previously encrypted bits. The result of this design is that no padding is needed to be applied to the PT, unless there is a separate requirement for byte alignment or alignment to a particular number of bytes, such as 8. Error correction coding, however, may require padding.

8.2 Initialization Vector

A 96-bit Initialization Vector (IV) shall be created by preceding the 64-bit nonce with a 32-bit fixed value. The fixed value is expected to be spacecraft dependent.

8.2.1 The Nonce

The nonce shall be used to generate the IV, see Figure 3. There is one nonce value per Encryption Frame.

Note: from section 5.2.1.1 of the NIST 38D doc: "The IV is essentially a nonce, i.e., a value that is unique within the specified context, which determines an invocation of the authenticated encryption function on the input data to be protected."

8.2.2 Counter Block

The Counter Block shall be composed of a static Fixed field, the Nonce, and an Encryption Block Count field, resulting in the required number of bits (128). To generate the ICB, the Encryption Block Count portion is set to a value of 1 (i.e., 32 bits, with 31 zeros and a 1 in the least significant position).

The value in the Counter Block shall never be repeated for a given encryption key.

Note: The Counter Block is defined in the NIST 38 D GCM document.

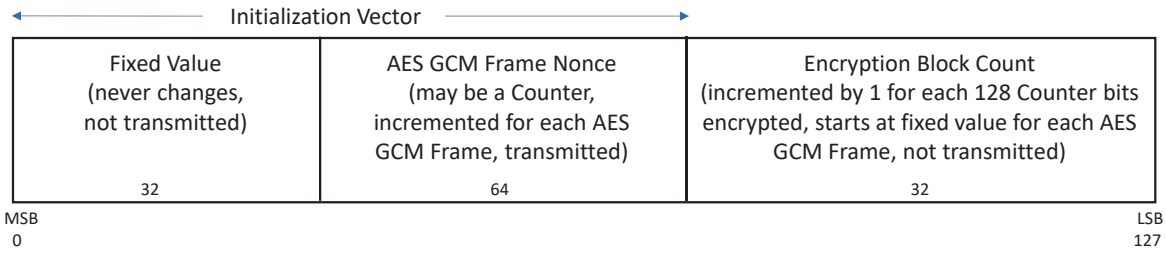

Figure 3. Example Counter Block

8.2.3 The Fixed Value and Frame Nonce

- The fixed value of the Counter Block shall be programmable but fixed per project.
- The fixed value of the Counter Block shall not be transmitted.
- The fixed value of the Counter Block shall be 32 bits in length.
- The encryption frame Nonce shall be the value that gets placed in the Counter Block Frame Nonce field transmitted with each Encryption Frame.

8.2.4 The Encryption Block Count

- The encryption count shall be incremented after each use within a given Encryption Frame. The Encryption Block Count field increments for each block of 128 bits that are encrypted in a given Encryption Frame. In combination of this field and the Nonce field,

the result is that the counter block never repeats, and if an incrementing count is used for the nonce, its value only increases.

8.2.5 The Encryption Block Count Length

- The encryption block count length shall be 32 bits.

9. SECURITY STRUCTURE

9.1 GCM

Encryption, Authentication, and Authenticated Encryption are defined in NIST SP 800-38D GCM Nov 2007.

9.2 Frame Error Control Field (FECF)

The CCSDS FECF (if used) is a CRC internal to the transfer frame and calculated prior to the security being applied. At the receive end, when the PT CLTU is received by the C&DH or Mission Processor, the FECF can be checked. This order allows detection of errors that occur in moving the data from the earlier stages after the FECF is applied at the sending end or after security is removed at the receive end.

10. ERROR CORRECTION CODING

10.1 Background

There are three codes available for up (forward) link error control coding (stated in bit values), the BCH (63,56), LDPC (128, 64) and LDPC (512, 256) codes, all described in CCSDS 231.0-B-4. The LDPC (512, 256) code is mentioned in several places in this document but the details for its uses are not fully defined in the initial version of this document.

After applying security, one of the three ECCs mentioned above shall be applied.

Only one of the codes shall be used for a given mission phase.

In general, it is important to not use a differential data format like NRZ-M after encoding as it will cause a doubling of code symbol errors upon converting back to NRZ-L. This is particularly important when attempting to use the BCH decoder in the single error correction mode.

Uncoded Secured Encryption Frames are also allowed but at the link designer risk.

Code rates are defined as the ratio of the input bits to the output bits where “n” is used for the number of output bits (called code symbols) and “k” is the number of input (message) bits. Code rate is a measure of k/n. The BCH is a rate 56/63 code and the LDPC codes are both rate 1/2 codes. For convenience and octet alignment, a spare bit is added to the output of the BCH code

making it a 56/64 = 7/8 code. The output bits are code symbols, but we are often sloppy here and loosely call them bits.

10.2 BCH Code

The BCH is a short code that can be used for single bit error correction, or up to 3-bit error detection. It has a high undetectable error rate. For near Earth missions with short round trip time, it is common to not use the code for single bit error correction but instead to just report the detection of an error via the telemetry (Appendix C) and have the command resent. For deep space missions with longer round-trip times, it is more common to use the code for single error correction, double error detection. When using the BCH code for single error correction, the NRZ-M data format shall not be used.

10.2.1 63,56 BCH Code

When using a heritage CLTU, the BCH codewords shall segment the transfer frame in units of 8 bytes (7 bytes of message, 1 byte of parity in each 8-byte codeword) as described in CCSDS 231.0-B-4.

Note: The 63,56 BCH code with an added spare bit shall be the BCH code defined in CCSDS 231.0-B-4, hence it becomes a rate 7/8 code.

When using this code, for every 7 message octets, the output is 8 code octets, so the coded message is expanded by 8/7 over the original uncoded message. With this code, the information message rate is not exactly 7/8 of the channel rate due to the synch marker and possible other fields.

10.3 LDPC Code

The LDPC codes are more powerful than the BCH code but also more complex to implement. The LDPC (128, 64) code is a rate 1/2 code so the output code symbols expand the message by a factor of 2. The LDPC (512, 256) code is also a rate 1/2 code and hence it has the same expansion factor of 2. With this code, for a given physical channel code symbol rate, the information message rate would be half of the channel rate. It is common to double the physical channel code symbol rate so as to not cut the message rate in half. (This is similar to how a rate 1/2 convolutional code is used.)

Both codes are transparent, but at the receiving end, the inversion sense of the CSM must be resolved. If the CSM is found in the inverted sense, the LDPC decoding will work but the data must be put in the proper sense prior to the security function.

10.4 Coding and Randomization

The Coded Data shall consist of a set of codewords defined in the latest version of CCSDS 231.0-B-.

The coded data shall use the randomizer defined in CCSDS 231.0-B-.

Note: The BCH encoding procedure is described in section 3 and the LDPC encoding procedure is in section 4 of that document. This security standard differs from how CCSDS specifies coding on a CLTU in that the coding here is applied to the entire secured Encryption Frame after the security. A heritage CLTU will also have the internal randomization on the transfer frame and will have the BCH code. All three of these codes are systematic (the original bits are not changed by the coding process; parity symbols are added to allow for error correction). Once coded, the original bits and the parity bits are referred to as code symbols and are randomized.

10.5 Service Providers and Space Link Extension

When the BCH code is used, the MOC shall apply it to the Encryption Frame, even if there is a BCH code inside the heritage CLTU.

Note: If the DSN is used as the service provider, either the DSN or the MOC can apply the selected LDPC code to the Encryption Frame. In all cases, the service provider usually performs the NRZ-L to M conversion (if selected) prior to modulation and transmission.

This standard is intended to be similar to techniques currently used on several NASA/GSFC satellites.

There are some important points to remember including:

1. GCM encryption is indicated instead of CCM.
2. The Size field is the length of the encrypted data. Padding will be added to the secured Encryption Frame so that the length is an integer multiple of the selected code message size, k. For the BCH code k = 56 bits (7 octets), and for the LDPC codes k = 64 bits (8 octets) or 256 bits (32 octets), depending on which LDPC code is selected.
3. At the receiving end, the Tail is used to determine the coded message length and the Size is used to determine the amount of padding. See example below.
4. When BCH coded, the Encryption Frame length plus the code padding, will increase by a factor of 8/7.
5. When LDPC coded, the Encryption Frame length will be increase by a factor of 2 and the transmission code symbol rate will be double the uncoded data rate resulting in approximately the same message rate as when not coded. The added synch marker causes the rate to not be exactly the same as the uncoded rate.

ERROR CORRECTION CODING Summary:

Error Correction Coding (when used) is applied after Encryption and authentication (CSM, Size, Nonce, Message Authentication Code (MAC) not encrypted, EB90 in a HCLTU is encrypted). A CSM synch marker and Idle are not coded. Randomization applied after coding, restarts after each CSM synch marker.

11. SIZE AND NONCE

The values in the Size Field and Nonce Field are used by the encryption, coding, randomization, de-randomization, decoding, decryption, and authentication functions; hence they cannot be encrypted, but they do get error correction coded (ECC).

11.1 Size Field

- The Size Field as shown in Figure 4, shall follow immediately after the synchronization marker with no gap between them.
- The nonce shall be placed after the Size Field and prior to the CT segment with no gap between them.

  Note: When used with a heritage CLTU, the nonce will be prior to the EB90 Start Sequence. When used with a transfer frame (TC, AOS or USLP) the nonce will be placed prior to the secured transfer frame or section thereof.
- The size field shall be 16-bits in length.
- The value in the Size field plus 1, shall be the number of PT bytes to be encrypted in the current Encryption Frame.

  Note: At both the sending and receiving end, this number will be used to calculate the length of the coding and decoding related functions and hence indicate where the MAC and ciphertext is to be found. The Tail indicates the end of the coded data but since there can be padding for the coding the Size field is still needed.

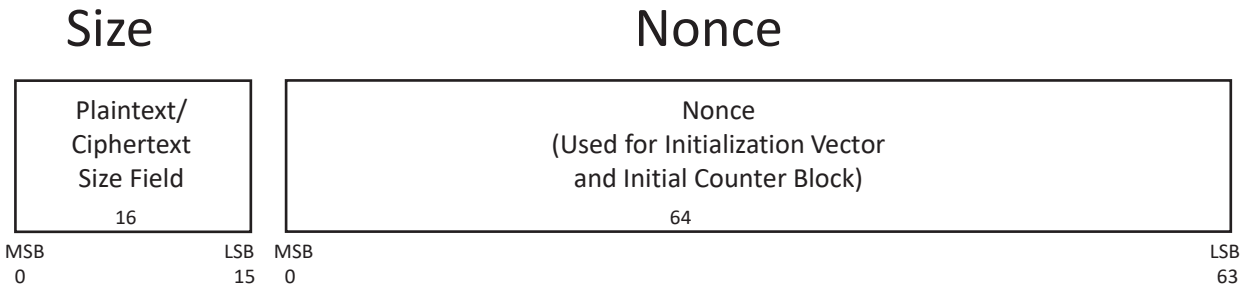

Figure 4. Size and Nonce Fields

11.1.1 Determining the Length of Items to be Secured, Coded, Decoded, and Decrypted

At the sending end, the size field is used to define the length of the data to be encrypted. After encryption, calculation of the MAC and attachment of the MAC, the error correction encoding process uses the Size field to determine the length of the data to be error correction coded and to determine if codeword padding is needed. This length is calculated by using the value in the Size Field and the known lengths of the managed parameters. For example, the length to be coded is the length of the Size field, plus the length of the Nonce field, plus the length of the encrypted data (the value in the Size field plus 1), plus the length of the MAC, plus the number of bytes

needed to result in an integer multiple of the code message length, k. The Size and Nonce are part of the authenticated data and are referred to as Additional Authenticated Data (AAD).

At the receiving end, either the Tail or Size field is needed for determining the coded and randomized data length. When a Tail is not used, the size field provides that information. The type of coding and use of a Tail are managed parameters. so Upon finding a Synch Marker, de-randomization is performed and the first codeword is decoded. Upon decoding and reading the Size Field, the length of the Coded data and Encryption Frame is calculated, the de-randomization process continues, and the decoder acts on the remaining codewords. The size field also provides the information needed by the authentication and decryption process. When a Tail is used, the length of the data is found after de-randomization and decoding, and the Tail is found. The Tail is defined to fail the decoding process.

The authentication process requires that all of the codewords in the current Encryption Frame be decoded and the code parity removed. This does not apply to the BCH code internal to a heritage CLTU which is not decoded at this stage.

11.1.2 Padding

The GCM encryption (GCTR function) handles data lengths that are not multiples of the encryption block length and therefore padding is not required for the encryption function. Details are in the Cipher section.

At the sending end, prior to the error correction coding, when the combined length of the Size, Nonce, ciphertext, and MAC is not a multiple of the codeword message size, padding shall be added to make it so.

The pattern of the padding shall be 0x55 per byte.

Padding example with 180 bytes of data:

After encryption, the Size + Nonce fields (10 bytes), Data (180 bytes) and the MAC (16 bytes) field sizes are added prior to the error correction coding, resulting in 206 bytes to code per Encryption Frame. If the BCH code with a k = 7 bytes is being used, the number of codewords will be 206/7 = 29.43 requiring 4 additional bytes to be added to make a multiple of 7, 210/7 = 30. After reading the Size field at the receive end, this same calculation (add length of Size, Nonce and MAC then divide by k) is done and the last 4 bytes of padding are removed (or ignored) prior to authentication and decryption.

For the LDPC (128,64) bit code, (16,8) bytes, a similar calculation is done but here 8 is the divisor. In this example a pad of 2 bytes would be required.

11.1.3 Error Correction Encoding and Decoding

After encryption and authentication, the Size, Nonce, ciphertext, MAC, and padding are coded, the tail is then added and all of the above are randomized. At the receiving end, a similar calculation to what was done on the sending end is done to determine the length of the data to be decoded. Any additional pad bytes added for the error correction coding are discarded prior to sending the Size/Nonce, Ciphertext, and MAC for authentication, then decryption.

11.2 Nonce

The Nonce contains a value (commonly a counter, Figure 4) used as part of the IV which is used by the encryption, decryption, and authentication functions. The purpose of the nonce is for anti-replay protection.

- The nonce field shall be 64 bits.
- The sender shall maintain a stored value of the nonce.
- The nonce shall not be encrypted.
- For the life of a given key, the value of the nonce shall not be repeated.
- The nonce shall be different from any other counter used elsewhere in the system.
- The IV shall be composed of a user defined and managed static field followed by the nonce, resulting in a length of 96 bits, Figure 3.
- The value of the nonce shall be strictly incrementing over time.

Note: The Nonce value and associated key index may be sent to the ground in the spacecraft health and safety telemetry at some limited project determined interval (e.g., once a week).

To provide anti-replay protection, the nonce must be changed from Encryption Frame to Encryption Frame according to an established rule. That rule can be as simple as incrementing its value by 1 for each Encryption Frame or by following a more complicated algorithm.

Note: Missions may define a mission-specific nonce-acceptance window to support long round-trip-time (RTT) operations or Delay-Tolerant Networking environments, where limited out-of-order delivery may occur, provided replay protection is preserved.

The nonce value shall be set using a method which avoids reuse for a given key. This includes avoiding reuse after a power failure. The following are two example methods that could accomplish this.

At the sending end:

1. If an incrementing counter (not necessarily by 1) is used, the security function stores the most recently used nonce value.
    a. An Encryption Frame may contain several blocks of data.
    b. For each Encryption Frame of data that contains a Nonce, the stored nonce value is incremented by at least 1 upon invocation of the authenticated encryption function. (upon the success of an authenticated encryption.) See section 8 of the NIST 800 38D doc for details.
    c. The nonce values can be recorded in a non-volatile memory, at some project determined interval. It is suggested that the interval be 100 or greater. The interval depends on the life cycle of the non-volatile memory being used. Should there be a power failure, the next nonce value used is the stored value plus the interval value or greater.
    d. When using a simple increment with the requirement that each nonce has a greater value than the previous, a window might be used to limit how much greater a following nonce may be.

2. A time epoch-based approach may be used where the first nonce for a given second is equal to the number of seconds since epoch*128. The 128 factor allows for up to 128 commands per second before rolling into the next second. The advantage of this approach is that there is no need to store nonces and share them across different operation centers.

At the receiving end:

1. The value of the nonce most recently used for a successful authentication and decryption is stored.
2. Upon initial power up or equipment cold reset, the nonce value for each key is set to zero.
3. Prior to the next authentication and decryption operation, the security function checks that the value in the received nonce is at least 1 greater than the stored value.
4. When an Encryption Frame of data is successfully authenticated and decrypted, the stored nonce value is replaced by the current one. If there is an authentication failure, the stored nonce value is not incremented and the Encryption Frame with the failed authentication is deleted.
5. The spacecraft will receive and process a command with a nonce counter that is at least 1 greater than the stored value.
6. Section 9 of NIST 38D specifies, “When power is restored, neither the preceding IV nor any other previous IV shall immediately be repeated for the key”. After an AES or power reset that causes the stored Nonce to default to zero, any nonce value greater than the previously used value can be used for the default key. The ground system maintains the most recent nonce count for each key, or the maximum value used among all of the keys and may use that to recover from a security reset.

The GCM standard allows for AAD that does not get encrypted but is used in the authentication process. This includes the Size and Nonce fields.

12. SYNCHRONIZATION

12.1 Start Sequence and Code Synchronization Marker

The term Start Sequence has several meanings.

When using bulk security with a heritage CLTU, the EB90 Start Sequence shall remain in the CLTU as specified in CCSDS 231.0-B-4 section 5.2.2.2. This Start Sequence does not function as the Code Synchronization Marker as it did in a heritage CLTU. This Start Sequence synchronization marker may be used by the C&DH at the receiving end after the error correcting coding and security is processed. In this standard, where a heritage CLTU has security applies and is then coded, the Size and Nonce fields precede the EB90 Start Sequence, requiring a synchronization marker that precedes the Size and Nonce fields. Since the first use of this synchronization marker is to perform the derandomizing and decoding, we use the term CSM. When using a secured and coded heritage CLTU, keeping the Start Sequence ensures that the resulting PT heritage CLTU will be identical to an unsecured CLTU and can be processed using heritage software/hardware at the sending and receiving end.

When **not** using a heritage CLTU, the EB90 is not required. In alignment with CCSDS 231.0-B- specifies, a 64 bit synchronization maker is used and CCSDS also calles this a Start Sequence. In this standard the term is CSM to distinguish it from the heritage CLTU Start Sequence, ever if the pattern used for the CSM contains EB90.

In order to identify the coded Encryption Frame at the receiving end, the sending end shall place a synchronization marker (CSM) prior to the Size field with no gap between them.

The synchronization marker shall be the 64-bit value, 034776C7272895B0 used by CCSDS or 555555555555EB90. In a binary representation the CCSDS 64-bit synchronization pattern is as shown in Figure 5 (spaces not to be included):

00000011 01000111 01110110 11000111 00100111 00101000 10010101 10110000
↑ ↑
BIT 0 BIT 63

Figure 5. CCSDS 64 Bit Synchronization Marker

To accommodate the selection of a CSM, the ground system shall be programmable.

Note: Since this synchronization marker is first used to delimit the coded secured Encryption Frame for the decoding process, the synch marker is called a CSM

Note: The CCSDS patten is recommended in CCSDS 131.0-B-4 section 8.2.2.1 and 9.3.5, for use with the three lower rate LDPC codes, and in CCSDS 231.0-B-4 section 5.2.2.3. In CCSDS 231.0-B-4 the value is shown as 0347 76C7 2728 95B0. The blank spaces between characters are not to be included).

Note: On several previous missions, the Synchronization marker is called a Barker code and has a default value of 034776C727289580. This is different from what is stated in CCSDS 131.0-B-4 section 8.2.2.1 and 9.3.5 and CCSDS 231.0-B-4 section 5.2.2.2 in that the second to last character is an "8" rather than the CCSDS specified "B". We suspect that this was a typographical error and was intended to be the same as the CCSDS pattern.

Note: Spacecraft synchronization marker search should allow a tolerance of several bits. For near Earth, a tolerance of 2 is common, for deep space when using LDPC, the value should be higher, closer to 6.

12.2 Transfer Frame Synchronization Marker

At the MOC, a project shall put a synchronization marker preceding each TF unless that project designs the spacecraft to delimit the plaintext TF by some other means (e.g., a spacecraft design where the security and data protocol layer are aligned, and one Large Encryption Frame (LEF) per Transfer Frame is used instead of several SEFs).

Note: when a CLTU is composed of several Small Encryption Frames (SEF), and an error occurs in one of the SEFs, that SEF is deleted at the Decoding or Security layer. The higher data protocol layer will receive a partial CLTU, and loss of synchronization can occur. The Transfer Frame Synchronization Marker is needed to re-synchronize. With Long Encryption Frames (LEF) an RF or security error is not a problem, since the entire CLTU will be discarded. However, even if the TF Length field is used to determine the Length of the variable length TF, the data protocol layer will need the TF synch marker to delimit received TFs. Formally speaking, we can say that this is not within the scope of this standard, it is at a higher layer of the protocol. A HCLTU will have the EB90 to function as the TF synch marker. Either that or the ASM 1ACFFC1D is recommended.

13. MESSAGE AUTHENTICATION CODE

Any bit pattern of the intended length can be processed by the decryption routine, and an output will result. The decryption process has no way to know if the input bit pattern came from an intended reliable source or has been corrupted in the RF transmission link. Give the decryptor an input and it will generate an output. An authentication process is used to add confidence that the message came from an authorized source and that it is not a duplicate of a previous message sent by that source or by an unauthorized source. If a command is not accepted, this does not preclude re-sending that command, but it does require that the Nonce be incremented from the value used for the last accepted command for each duplicate command sent.

In this standard, the message authentication code (MAC, also called Tag) algorithm is the one defined in the NIST 800 SP 38D GCM document. The AAD consists of the Size and Nonce fields and has a fixed length. Some of the details are listed here but the NIST 800 SP 38D GCM is the governing document.

13.1 Secured Data

The Encryption Frame shall include a MAC.

13.2 Fields Covered

The MAC shall be calculated after the data is encrypted, starting from the Size and Nonce to the end of the encrypted data.

13.3 MAC Position

The MAC shall be the last field in each Encryption Frame after the last bit of the encrypted data with no gap.

13.4 MAC Length

- The length of the MAC shall be 128 bits.
- The full 128 bits of the MAC shall be transmitted.

13.5 MAC Calculation

The calculation for the MAC (Tag) shall be as defined in the NIST 38 D GCM document.

Note: The result of that calculation ($GCTR_K(J_o,S)$) is 128 bits.

Note: To authenticate the data at the receiving end, a MAC' (MAC prime) is calculated on the same parts of the received data and compared to the MAC transmitted with the data. Since at the sending end, the MAC was calculated after the encryption, at the receiving end it must be calculated prior to decryption as shown in Figure 6. *The MAC is not encrypted.*

14. ORDER OF PROCESSING

Order of Processing Required as shown in Figure 6:

Send Side: Encrypt, Authenticate, Code, Randomize
Receive Side: De-randomization, Decode, Authenticate and Decrypt

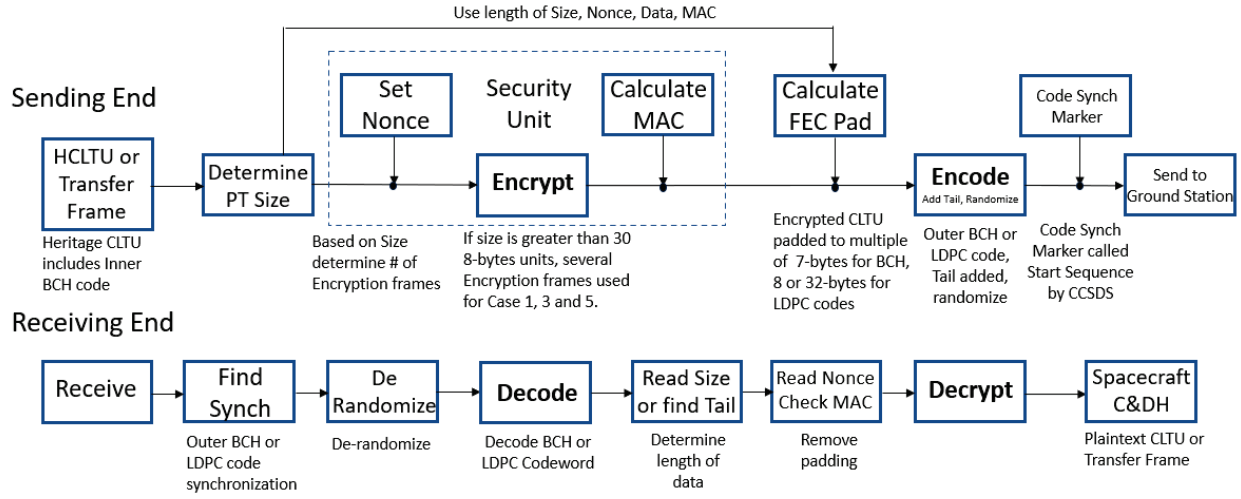

Figure 6. Function Flow for Security and Coding

Notes:

1. *Since bulk security delivers a transfer frame with the nonce assigned after the ground FOP, all of the functions related to COP-1 are intact at the receiving end. See appendix C for detail on COP-1 and bulk security.*
2. *Artemis uses 128-bit Authentication Tag (MAC) truncated to 64 bits.*

15. SECURITY AND CODING STRUCTURE

There are two general security approaches: The use of either one or more Small Encryption Frames or a single Large Encryption Frame for a total of 7 Use-cases that are covered in this section of the standard. If it turns out that the use of a Large Encryption Frame (covered in Use Cases 2, 4, 6, and 7 below) is technically impractical, it will be discarded in a later version of this standard. It is expected that a project will use only one of these use cases for a given mission phase.

1) Small Encryption Frame (SEF)

When CLTUs are encrypted in one or more Encryption Frames where each Encryption Frame does not contain more than 240 encrypted bytes, the method is referred to as using Small Encryption Frames. When using this method where each Encryption Frame is limited to a maximum of 240 encrypted bytes, plus the Size, Nonce and MAC fields, the maximum length will be 266 bytes. This is shown later in this section in Use Cases 1, 3, and 5.

2) Large Encryption Frame (LEF)

When the method used includes CLTUs that require more than 240 encrypted bytes and a single Encryption Frame is used, the method is referred to as using Large Encryption Frames. The frames are of variable length and are shown later in this section in Use Cases 2, 4, 6, and 7.

Note: For USLP implementations, the designer should consider the amount of memory and processing speed required for de-randomization, decoding, authentication and decryption on-board the spacecraft when selecting the transfer frame length and which approach to use.

Upon receipt of a PT heritage CLTU or transfer frame, the length of the Small/Large Encryption Frame shall be determined as follows:

- If SEFs are being used, the data length shall be used to determine how many Encryption Frames are needed and how to set the Size field in each individual Encryption Frame.

  Note: The maximum length heritage CLTU, which requires 5 Small Encryption Frames, contains 147 BCH internal code words plus the start and tail sequences.

- If LEFs are being used, the data length shall be used to set the Size field in the single Large Encryption Frame.

  Note: The maximum size of the transfer frame, either AOS or USLP, is 65,536 bytes. The length is the value in the size field plus 1.

15.1 Heritage CLTU with 63,56 BCH Code both Inside the CLTU and after Encryption (Use Cases 1 and 2)

In a heritage CLTU, the BCH codewords and Tail, are in units of 8 bytes, leaving only the two byte EB90 Start Sequence that is not 8 bytes in length. The length of the heritage CLTU to be

encrypted is arbitrary and usually greater than 16 bytes. The AES algorithm encrypts in units of 128 bits (16 bytes) so when the CLTU length is not a multiple of 16 bytes the last section of the encrypted counter block is truncated. See Cipher section for details. For Use Case 1, see Figure 7 and Figure 8. For Use Case 2, see Figure 7 and Figure 9.

The encrypted CT has the same number of bits as the PT.

Note: Reminder, the Synch marker, Size/Nonce, and MAC fields are not encrypted.

15.2 Encryption Frame

The secured CLTU may be in a single Encryption Frame or broken into up to 5 Small Encryption Frames. In all cases, each Encryption Frame has one Size, one Nonce and one MAC field and is preceded by a 64-bit Synchronization marker called a CSM.

With bulk security, when not using a heritage CLTU, the EB90 Start Sequence and Tail sequences, which are a kind of "wrapper" around the transfer frame are eliminated. The Size field indicates the number of encrypted bytes in a given Encryption Frame and the transfer frame length field allows the C&DH to calculate how many of the Encryption Frames are needed to assemble the transfer frame.

The use of either one or more Encryption Frames or a single Encryption Frame is a managed parameter.

15.3 Use Cases

The Following use cases are defined in this Standard and shown in the following figures. The term "CLTU" is defined in section 7.2. In all Cases, a Tail is the last field in the CLTU or each of the SEFs.

- Use Case 1 (SEF): Heritage CLTU with BCH code internal to the CLTU, secured in up to a maximum of 5 (Small) Encryption Frames, then with BCH coding applied to each Encryption Frame and preceded by a 64 bit CSM. The purpose of this Use case is to allow existing ground CLTU equipment and spacecraft CLTU avionics to continue to be used.
- Use Case 2 (LEF): Heritage CLTU with BCH code internal to the CLTU, secured in a single Large Encryption Frame, then with BCH coding applied to the Encryption Frame and preceded by a 64 bit CSM. This case also allows existing ground CLTU software and spacecraft software to be used.
- Use Case 3 (SEF): Command Transfer Frame broken into up to 5 (Small) Encryption Frames, BCH coded and each preceded by a 64 bit CSM. In this case, the CLTU is the 1 to 5 (Small) Encryption Frames, each BCH coded, and preceded by a 64 bit CSM. No internal BCH.
- Use Case 4 (LEF): Command Transfer Frame secured in a single Large Encryption Frame, BCH coded and preceded by a 64 bit CSM. In this case, the CLTU is the single Large Encryption Frame, BCH coded, and preceded by a 64 bit CSM.
- Use Case 5 (SEF): Command Transfer Frame broken into up to 5 (Small) Encryption Frames, LDPC (128, 64) coded and each preceded by a 64 bit CSM. In this case, the

CLTU is the 1 to 5 (Small) Encryption Frames, each LDPC coded and preceded by a 64 bit CSM.

- Use Case 6 (LEF): Command Transfer Frame secured in a single Large Encryption Frame, then LDPC (128, 64) coded and preceded by a 64 bit CSM. In this case, the CLTU is the single Large Encryption Frame, LDPC coded, and preceded by a 64 bit CSM.
- Use Case 7 (LEF): Command Transfer Frame secured in a single Large Encryption Frame, then LDPC (512, 256) coded and preceded by a 64 bit CSM. In this case, the CLTU is the single Large Encryption Frame, LDPC coded and preceded by a 64 bit CSM.

Table 1. Configurations Covered in this Standard

Use Case	Approach	Data Structure	Encryption Frame Method	Code	Figure Reference
1	SEF	H CLTU	$1 \leq n \leq 5$ EFs	BCH	7,8
2	LEF	H CLTU	1 EF	BCH	7, 9
3	SEF	TC TF	$1 \leq n \leq 5$ EFs	BCH	10, 11
4	LEF	TC TF	1 EF	BCH	10, 12
5	SEF	TC TF	$1 \leq n \leq 5$ EFs	LDPC (128, 64)	13, 14
6	LEF	TC TF	1 EF	LDPC (128, 64)	13, 15
7	LEF	TC TF	1 EF	LDPC (512, 256)	

15.4 Small Encryption Frame Length

Prior to the FEC coding that gets applied after the security, the Small Encryption Frames shall have a maximum length of 266 bytes.

Note: At the receiving end, the decoding and decryption function does not know if a given Encryption Frame is the first, middle or last Encryption Frame of a given CLTU or transfer frame. It does not need to know, because it simply decodes and decrypts the data and sends it to the C&DH, Mission Processor or some other higher layer protocol handler. The C&DH will see a flow of bytes as if the data was not secured and process them.

15.5 Steps for Security and Coding

When sending a transfer frame that is not in a heritage CLTU, the transfer frame shall have a CCSDS Synchronization Marker preceding the transfer frame with no gap between them. Since this synch marker is not used until after the error correction decoding, it is not a CSM. The short 16 bit EB90 or the 32-bit ASM is recommended.

Note: This is similar to the SMTF used in CCSDS 131.0-B-, and it is to allow the next higher processing layer to delimit a transfer frame.

Note: The details of how a user delimits the transfer frame are not within the scope of this standard.

Figure 6 in section 14 depicts the functional flow when either a heritage CLTU is used with outer BCH coding; or a transfer frame is used with BCH coding or with LDPC Coding. The following text describes the security and coding in detail.

Parameters used in the following (units are bytes):

- C length of the ciphertext (CT) variable
- P length of the plaintext (PT) = C, variable
- S length of the Size Field 2 bytes (Size Field contains the value C)
- M length of the MAC 16 bytes
- N length of the nonce 8 bytes
- F length of an Encryption Frame: variable, F = S + N + C + M (64-bit synchronization marker is not included) (this is length prior to coding)
- k length of the code message size k = 7 bytes for the BCH and 8 bytes for the 128,64 bit = 16,8 byte LDPC code
- n length of codeword
- D length of padding in bytes to make F + D an integer multiple of k
- E length prior to coding, of data that gets coded: variable E = S + N + C + M + D = F + D
- L number of codewords L = E/k
- G length of codewords (after coding) G = (n/k) E = nL

15.5.1 At the Sending End

At the sending end, after security, code padding, and coding and immediately following the padding, a Tail shall be applied.

Randomization shall then be applied to the coded encryption frame, padding and Tail.

Fields that are added shall be immediately before the following or after the preceding structure with no intervening bits.

Upon receiving a data item to be secured and coded, the length of the data item shall be determined, which is referred to as the total PT size, P_T in bytes.

Note: If SEFs are to be used (limited size Encryption Frames), determine the number of Encryption Frames that will be needed. Since each Encryption Frame is limited to 240 bytes of User data, the number of Encryption frames is $P_T/240$ rounded to the next higher integer. The Size value for each of the required Encryption Frames will be determined at this point. For a maximum length CLTU of 1186 bytes, this results in 5 Encryption Frames with $P_1 = 240$, $P_2 = 240$, $P_3 = 240$, $P_4 = 240$, $P_5 = 226$ bytes. After encrypting, the ciphertext components, C_i, will have the same length as prior to the encryption.

At the Sending End Use-Case 1 Heritage CLTU

1. Upon receiving data to be secured and coded, there are several fields that need to be populated and assembled in the following order.
   a. Size, Nonce, Ciphertext, MAC, Pad
   b. The Size, Nonce and MAC fields have fixed and defined lengths but the length of the Plaintext that becomes the Ciphertext is a variable length. C = P
2. Set the value in the Size Field equal to the number of PT bytes P to be placed in that Encryption Frame, for each Encryption Frame.
3. Set the value in the nonce field (using the stored value).
4. Encrypt the PT.
5. Calculate the MAC (details in NIST 38 D doc).
6. Increment the stored nonce value (assumes a simple incremented nonce is being used).
7. Send the string of Size, Nonce, Ciphertext, MAC to the coding processor.
8. Determine the size of this data F = S + N + C + M for each Encryption Frame.
9. Determine length of padding needed, calculate F/k, and determine the number of bytes needed to round to the next higher integer. For BCH code, k = 7, for LDPC (16,8) bytes, k = 8.bytes.
10. Attach those bytes of padding with value 01010101 after the MAC with no gap.
11. Code the string of length E = S + N + C + M + D bytes. With BCH code the length will increase by a factor of 8/7, with the 16,8-byte LDPC the length will increase by a factor of 2.
12. Randomize the coded data and the Tail.
13. Add the code synch marker at the beginning of the coded data string.
14. Add 8 bytes of idle at the end of the string. (8 bytes of 10101010 = 0xAA).
15. Send to ground station for transmission.

Example: given a CLTU of length 802 bytes, using BCH code on secured data. Note that this is not for a minimum length transfer frame that is shown in Figure 7.

At the sending end for the first of 4 SEFs:

1. If using the SEF approach with a maximum PT size of 240 bytes per Encryption Frame, 4 Encryption Frames will be needed, with the following lengths of PT, 240, 240, 240, 82. We will continue the example with one of the Encryption Frames where P = 240.
2. S = 240
3. Set nonce
4. Encrypt
5. MAC
6. Increment nonce
7. Send to coding function (which includes adding padding as necessary)
8. F = S+N+C+M = 2+8+240+16 = 266 bytes
9. 266/7 = 38 D = 0 (no padding for external BCH necessary)
10. E = S+N+C+M+D = 2+8+240+16+0 = 266 bytes prior to coding. E = 38x7 This will become 38 CW if BCH is being used.

11. After coding, the string length will become **G =** 266x8/7 = 304 bytes
12. Add Tail then Randomize
13. Add code synch marker
14. Add trailing idle
15. Transmit

15.5.2 At the Receiving End

1. At the symbol synchronizer or de-randomizer, convert data to the true sense if the CSM is found inverted.
2. Find synch marker. There is a continuous search for the CSM independent of other processing.
3. Begin de-randomization on at least the number of bytes needed to cover the first codeword. For BCH this will be 8 bytes, for 16,8 byte LDPC it will be 16 bytes.
4. Decode the first codeword (which contains the Size field whose value is the encryption size prior to coding.). *Note: the following assumes that the coded length will be found by using the Size field. If instead the implementation is such that the Tail is used, skip to line 10.*
5. Return k bytes. For (8,7) byte BCH k = 7 bytes, for (16,8) byte LDPC k = 8 bytes
6. Read first 2 bytes which are Size field that contains the value C, the length of the ciphertext. (C = P for the AES cypher)
7. Calculate the length of the coded data prior to padding and coding, F = S + N + C + M
8. Divide this length by the k of the code and round up to determine the number of codewords, E= F/k rounded up to the next integer, E=⌈F/k⌉. (ceiling function)
9. The length of the data that is coded, G, is the number of codeword times n of the code.
10. Complete de-randomizing the rest of the coded data without re-initializing.
11. Decode all codewords (Decode before Decrypt).
12. Delete or ignore any padding used to make an integer number of codewords D = E - F
13. Read the nonce value and construct the IV (ICB and additional CBs as needed).
14. Check the MAC. Do same calculation that was done on the sending end, across the Size, Nonce and Ciphertext data to generate MAC'.
15. Compare to the received MAC. If there is a failure, delete the entire Encryption Frame and report the authentication security failure via telemetry.
16. If pass, decrypt ciphertext to obtain the PT. If fail, report decryption failure via telemetry.
17. Send only the PT to the next higher layer of processing (Synch, Size, Nonce, MAC and Idle are removed).
18. For Use Cases 3 and higher, if the idle is used to indicate the end of command, it is suggested that the command from the MOC contains the idle as part of the PT that gets encrypted. This is to avoid problems during testing where the data is processed differently from when operational.
19. After stringing the pieces of PT together, the next higher layer will see the original data as if it was never secured and coded.

Example: continuing with example from above, using BCH code on secured data

At the Receiving end:

1. Check for inverted synch marker
2. If inverted form of Synch marker is found, invert all following data to put it in proper sense.
3. Begin derandomize on at least 8 bytes, 1 CW
4. Decode first codeword
5. Return 7 bytes
6. Read Size = 240 bytes (PT size prior to encryption in first SEF)
7. F = S + N + C + M = 2+8+240+16 = 266
8. F/k = 266/7 = 38 Number of codewords after rounding up = 38
9. E = 38x7 = 266
10. Derandomize to length of 266 bytes
11. Decode the remaining 37 codewords, total of 38
12. Delete padding if any. None in this example with a result of a total of 266 bytes
13. Read nonce
14. Calculate MAC' (MAC prime)
15. Compare to received MAC
16. Decrypt
17. Send PT to next higher processing layer.

15.6 Use Case 1: Heritage CLTU with 1 to 5 Small Encryption Frames - BCH

This case covers a Heritage-CLTU with the BCH code internal to the CLTU, bulk secured in up to 5 Small Encryption Frames, then BCH coded on the Encryption Frame(s). The limit of 5 Encryption Frames is based on the length of a single maximum length command transfer frame. This use case allows existing ground CLTU software and spacecraft software to be used. (This case might be retired in the next 5 to 15 years.)

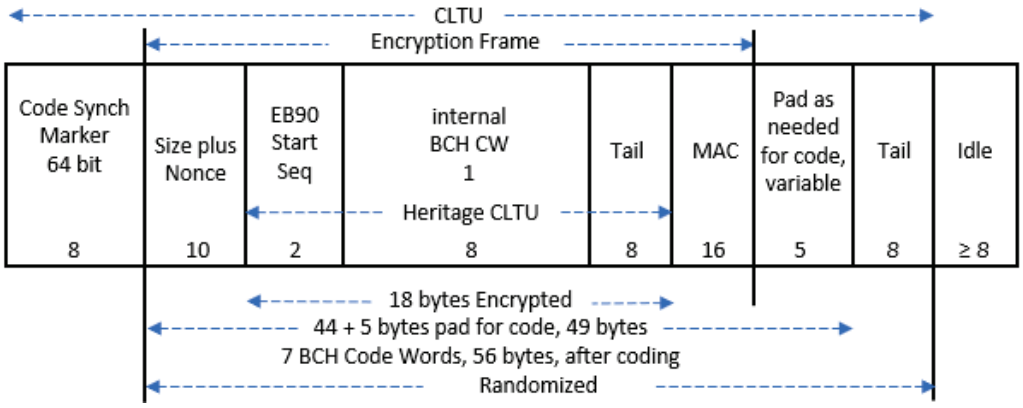

Figure 7. Use Cases 1 and 2: HCLTU with Min Length Transfer Frame

No padding is required for the encryption at the level of this document, the GCM algorithm incorporates what is needed. The result of encryption does not change the length of the encrypted section but after encryption a 16 byte MAC (Tag) is added for authentication. See Figure 7.

After encryption and authentication, an outer BCH code is applied. It covers the Size, Nonce, encrypted data and the MAC which in aggregate has a length of 44 bytes. Since the BCH coding is in units of 7-bytes, 5 bytes of pad needs to be added. Each 7-byte unit is coded into an 8 byte BCH codeword, resulting in 56 bytes after encoding. An 8 byte code synch mark is prepended to the BCH coded data, an 8-byte Tail is added at the end and at least 8 bytes of idle are placed after the Tail. The idle is not BCH coded.

A CLTU contains only one transfer frame. When the length of the CLTU is larger than 240-bytes, up to 5 secured and coded Encryption Frames are used. For a transfer frame of 224 bytes or less, the resulting HCLTU will fit in a single small Encryption Frame where 234 bytes are encrypted. The Size plus Nonce and MAC complete the small Encryption Frame to 260 bytes prior to encoding. A 6 byte pad is needed for the outer BCH coding.

For larger transfer frame data, additional small Encryption Frames are used. See Figure 8. The Maximum length CCSDS command transfer frame has a length of 1024 bytes and after the internal (inner) BCH code is applied, has a length of 147 x 8 = 1176 bytes, 5 small Encryption Frames are required for a single maximum length transfer frame. The CLTU is the combination of all of the SEFs.

After encryption and authentication, an outer BCH code is applied. It covers the Size, Nonce, the encrypted data and the MAC which in aggregrate has a maximum length of 266 bytes prior to coding, for each Encryption Frame. Since the BCH code is in units of 7-bytes, padding will often be needed but not for 266 bytes. Each 7-byte unit gets coded into an 8 byte BCH codeword. For a maximum length Encryption Frame, there will be 38 codewords and 304 bytes after encoding. Each Encryption Frame will have a CSM and a Tail. The CLTU is the combination of the Small Encryption Frames.

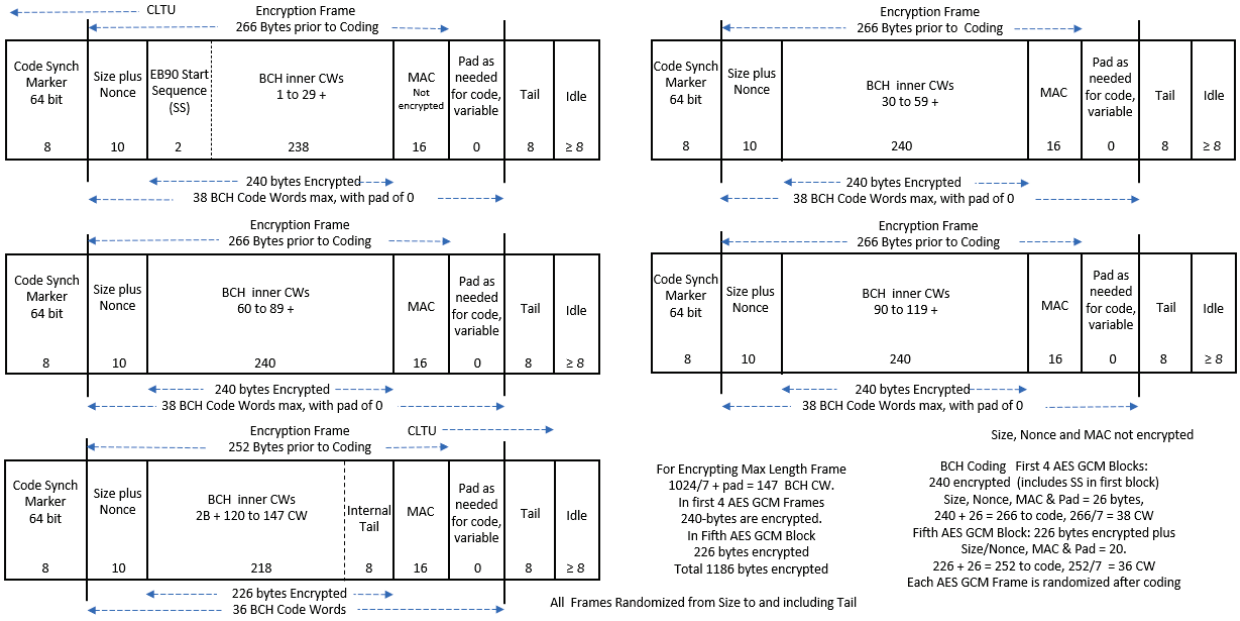

Figure 8. Use Case 1: HCLTU Max Length Transfer Frame in 5 Small Frames

Note that in these 5 frames the CLTU with the inner BCH, there are partial BCH CWs in each frame. The intent is that at the receiving end, after decryption, either the PT data is passed to the next higher processing layer which will assemble the PT byte stream from each frame, find the EB90 Start Sequence and do the required processing, OR store the PT until an internal tail is found, then send to the next higher processing layer. In this example the slicing of the CLTU is as follows:

First frame	2 byte Start Sequence + 29 BCH CWs + 6 bytes of CW 30
Second frame	2 bytes of CW 30 + 29 BCH CWs + 6 bytes of CW 60
Third frame	2 bytes of CW 60 + 29 BCH CWs + 6 bytes of CW 90
Fourth frame	2 bytes of CW 90 + 29 BCH CWs + 6 bytes of CW 120
Fifth frame	2 bytes of CW 120 + 27 BCH CWs + 8 byte Tail

15.7 Use Case 2: Heritage CLTU with Single Large Encryption Frame – BCH Coded

This case covers a heritage CLTU with BCH code internal to the HCLTU, bulk secured in a single Large Encryption Frame, then with BCH code on the encryption frame.

Figure 7 and the text associated with it applies for a minimum length HCLTU and transfer frame. Figure 9 shows the case when a Large Encryption Frame is used.

The Maximum length command transfer frame has a length of 1024 bytes and after the internal BCH code is applied, is made up of 147 BCH 8-byte codewords and has a length of 147 x 8 = 1176 bytes.

The Size, Nonce and MAC fields fill the Large Encryption Frame prior to coding. See Figure 9.

After encryption and authentication, a BCH code is applied. It covers the Size, Nonce, encrypted data and MAC resulting in a maximum length of 1212 bytes. Since the BCH coding is in units of 7-bytes, padding of 6 bytes is needed. Each 7-byte unit gets coded into an 8 byte BCH codeword, resulting in up to 174 codewords and 1392 bytes after coding. A coded large encryption frame has a CSM placed at the beginning, an 8-byte Tail at the end, followed by at least 8 bytes of idle.

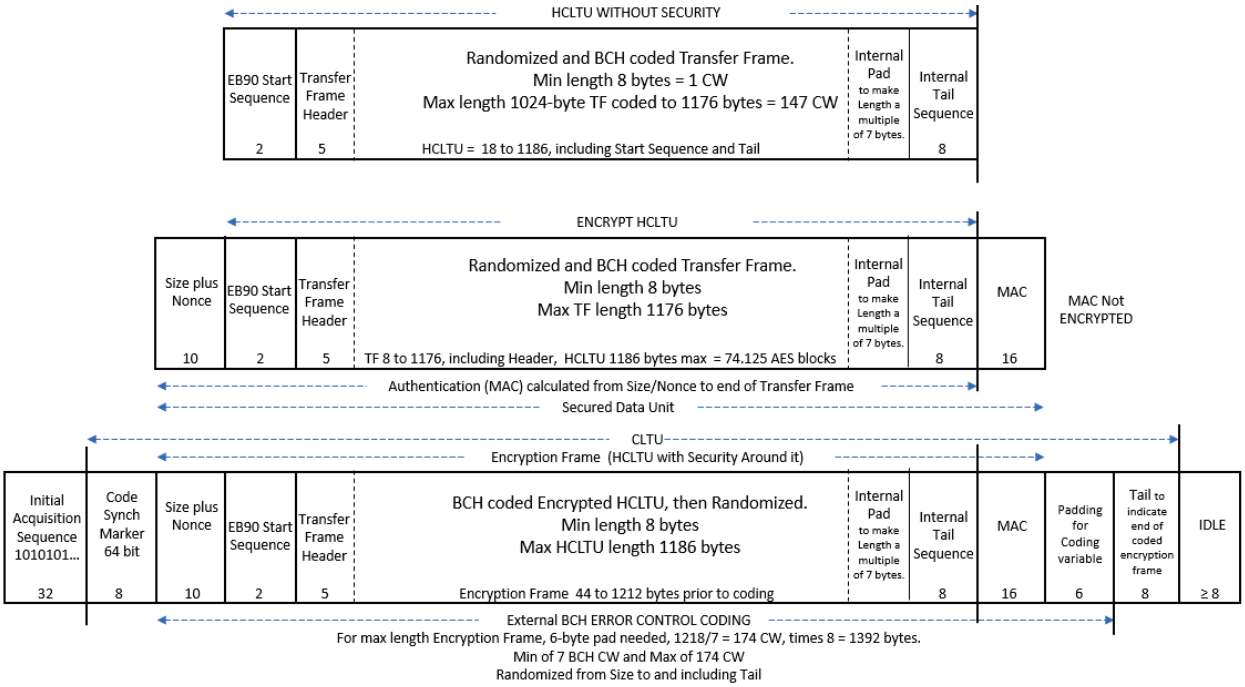

Figure 9. Use Case 2: CLTU Max Length Transfer Frame in single Encryption Frame

15.8 Use Case 3: Transfer Frame contains 1 to 5 Small Encryption Frames – BCH Coded

This case covers a transfer frame, bulk secured in up to 5 Small Encryption Frames, then BCH coded on the Encryption Frame(s).

For Use Case 3, the transfer frame shall have a CCSDS transfer frame synch marker prepended to the transfer frame.

Note: The Size field in the transfer frame primary header can be used instead of the Tail Sequence. This case allows existing ground software and spacecraft software to process the BCH code but in this case, it is applied after the security. There is no internal BCH code as in a heritage CLTU.

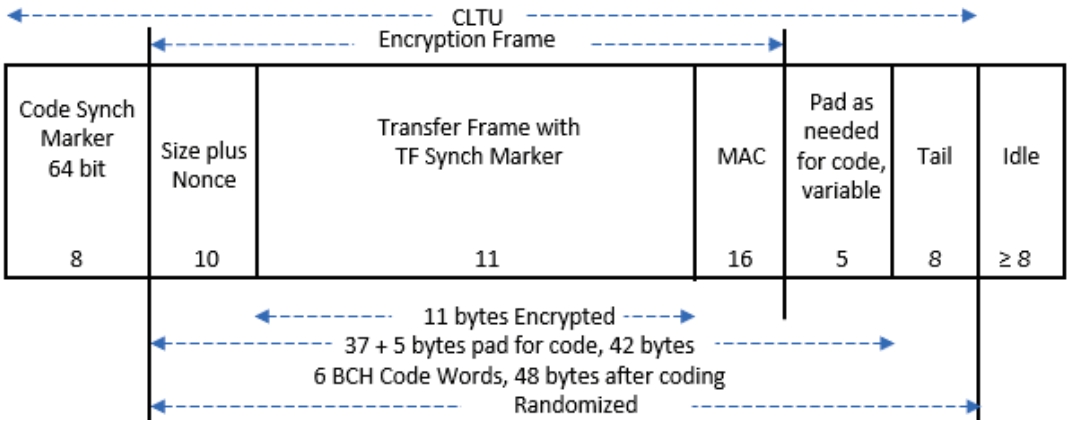

Figure 10. Use Cases 3 and 4: Transfer Frame Minimum Length – BCH Coded

For minimum length transfer frame there is one Encryption Frame with a length of 11 bytes to be encrypted. No padding is required for the encryption. The result of encryption does not change the length of the encrypted section but after encryption, a Size, Nonce and MAC (Tag) fields are added. See Figure 10.

After encryption and authentication, a BCH code is applied. It covers the Size, Nonce, encrypted data, MAC and code padding fields resulting in a length of 42 bytes. Since the BCH coding is in units of 7-bytes, 5 bytes of pad need to be added to accommodate the coding. Each 7-byte unit get coded into an 8-byte BCH codeword, resulting in 56 bytes after coding. An 8-byte code synch mark is prepended to the coded data, an 8-byte Tail is appended and at least 8 bytes of idle are placed at the end of the Encryption Frame.

For transfer frames that are larger than the minimum length, up to 5 secured and coded Encryption Frames are used. The Maximum length transfer frame with a 4 byte CCSDS transfer frame synch marker has a length of 1028 bytes. This and following examples use a 4 byte CCSDS transfer frame synch marker, but 2 bytes would be sufficient.

For a transfer frame with a CCSDS synch marker of 240 bytes or less, the resulting coded transfer frame will fit in a single Encryption Frame where 240 bytes max are encrypted. The Size, Nonce and MAC fields fill the Encryption Frame to 266 bytes max prior to coding. For larger transfer frames, additional Encryption Frames are used. See Figure 11.

After encryption and authentication, a BCH code is applied. It covers the Size, Nonce, encrypted data and MAC fields which have a maximum length of 266 bytes for each Encryption Frame. Since the BCH coding is in units of 7-bytes, padding will often be needed. Each 7-byte unit get coded into an 8-byte BCH codeword. For a maximum length Encryption Frame, there will be 304 bytes after coding. An 8-byte code synch mark is prepended to the coded data, an 8-byte Tail is appended and at least 8 bytes of idle are placed at the end of the Frame.

At the receive end, for determining the length of the encrypted and coded variable length data, an implementation using the Tail field is simpler than using the TF Size field.

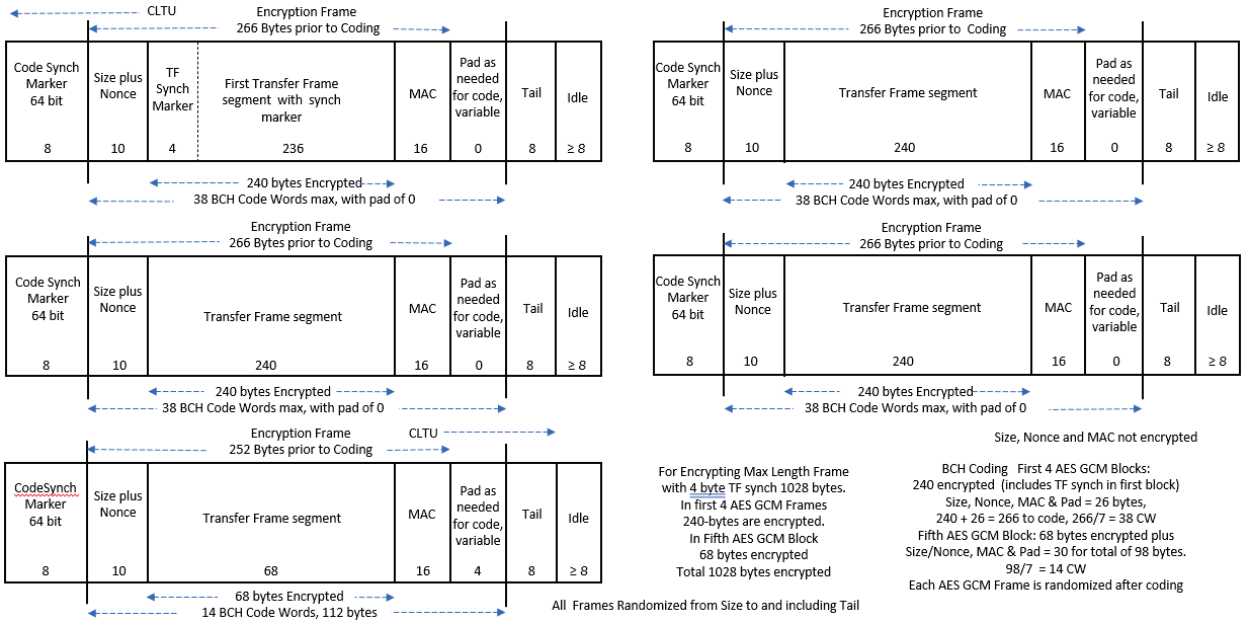

Figure 11. Use Case 3: Transfer Frame Max Length in 5 Encryption Frames, BCH Coded

15.9 Use Case 4: Transfer Frame – Single Large Encryption Frame – BCH Coded

This case covers a transfer frame bulk secured in a single Large Encryption Frame, then BCH coded on the encryption frame.

For Use Case 4, the transfer frame shall have a CCSDS transfer frame synch marker prepended to the Transfer Frame.

Note: Figure 10 and the text associated with it applies for a minimum length transfer frame. Figure 12 shows the case when a single Large Encryption Frame is used where the frame is not limited to 266 bytes after encryption and authentication.

No matter the length of the transfer frame, a single secured and coded Encryption Frame is used. The Maximum length transfer frame with a 4 byte CCSDS synch marker has a length of 1028 bytes. The Size, Nonce and MAC fields fill the Encryption Frame prior to coding. See Figure 12.

After encryption and authentication, a BCH code is applied. It covers the Size/Nonce, encrypted data MAC fields resulting in a maximum length of 1054 bytes. Since the BCH coding is in units of 7-bytes, padding will often be needed. Each 7-byte unit get coded into an 8-byte BCH codeword, resulting in up to 1208 bytes after coding. An 8-byte code synch mark is prepended to the coded data, an 8-byte tail is appended at the end and at least 8 bytes of idle are placed at the end of the Encryption Frame.

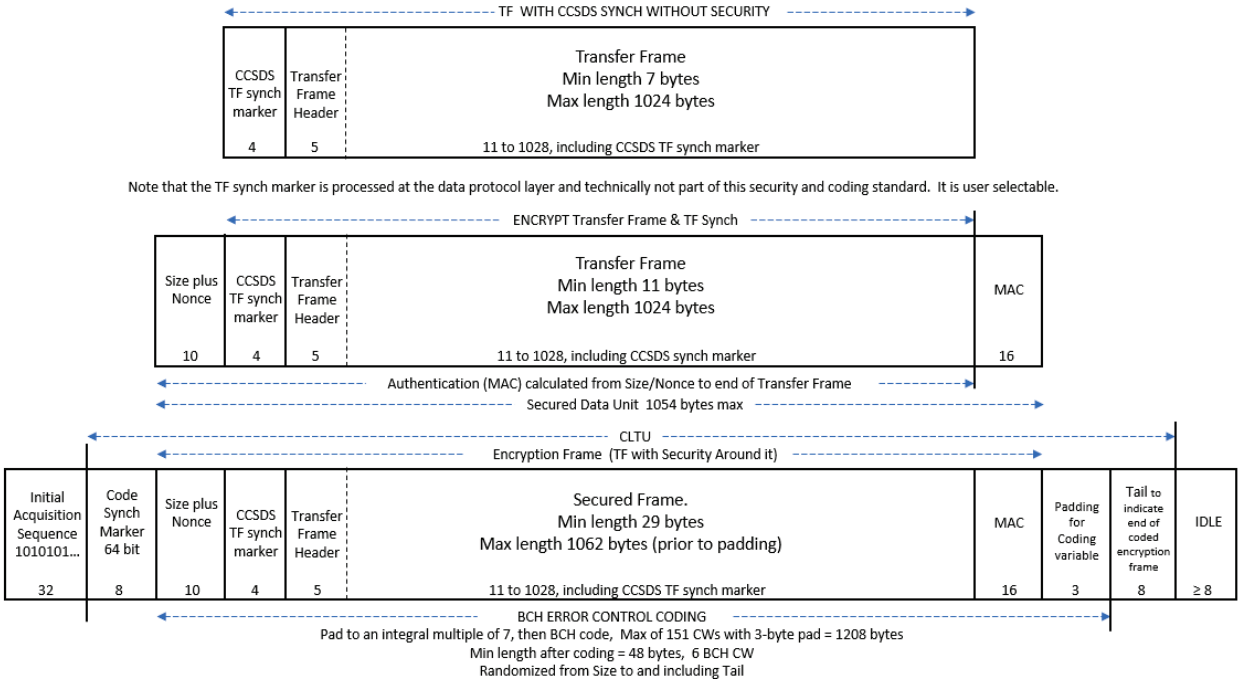

Figure 12. Use Case 4: Transfer Frame Max Length in a Single Large Encryption Frame, BCH Coded

15.10 Use Case 5: Transfer Frame – 1 to 5 Small Encryption Frames – LDPC (128, 64)

This case covers a transfer frame, bulk secured in up to 5 Encryption Frames, then LDPC (128, 64) bit coded.

This case makes use of an LDPC code that is more powerful than the BCH code.

For Use Case 5, the transfer frame shall have a CCSDS synchronization marker prepended to the Transfer Frame.

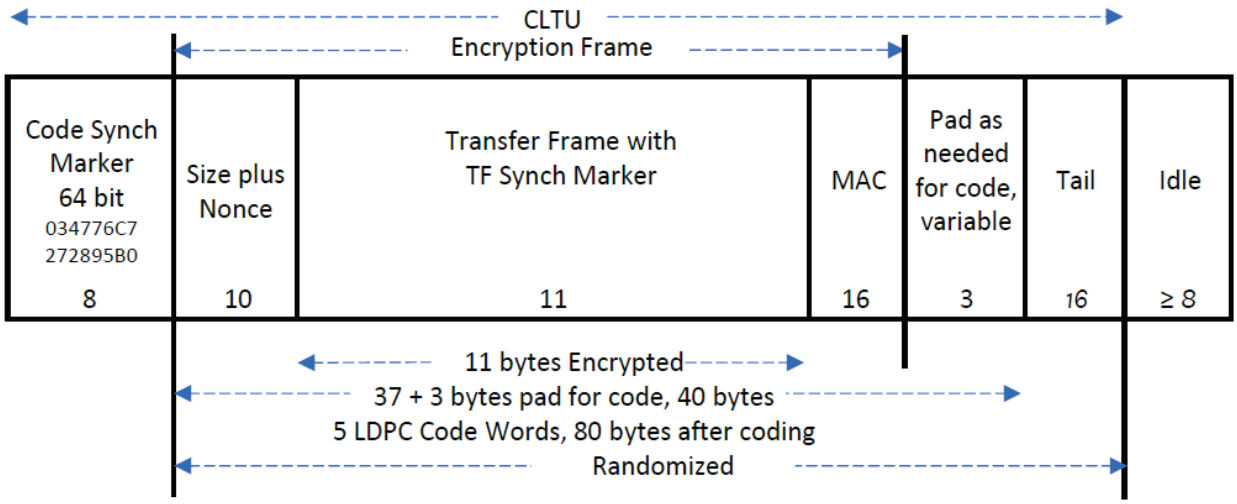

Figure 13. Use Cases 5 and 6: Transfer Frame - Minimum Length, LDPC Coded

For minimum length transfer frame with 4 byte CCSDS transfer frame synch marker, there is one Encryption Frame with a length is 11 bytes to be encrypted. No padding is required for the encryption. The result of encryption does not change the length of the encrypted section but after encryption, a Size, Nonce and MAC (Tag) fields are added. See Figure 13.

After encryption and authentication, an LDPC code is applied. It covers the Size, Nonce, encrypted data and MAC which has a length of 37 bytes. Since the LDPC coding is in units of 8-bytes, three bytes of padding are needed. Each 8-byte unit get coded into a 16-byte LDPC codeword, resulting in 80 bytes after coding. An 8-byte code synch mark is prepended to the coded data, a 16-byte Tail is appended and 8 bytes of idle are placed at the end of the Frame.

For transfer frames that are larger than the minimum length, up to 5 secured and coded Small Encryption Frames are used. The Maximum length transfer frame with a 4 byte CCSDS synch marker has a length of 1028 bytes.

For the small PT data where a transfer frame with CCSDS transfer frame synch marker of 240 bytes or less, the encrypted frame will fit into a single Small Encryption Frame. The Size, Nonce and MAC fields fill the Encryption Frame to 266 bytes max prior to coding. For large PT data, additional Encryption Frames are used. See Figure 14.

After encryption and authentication, an LDPC code is applied. It covers the Size, Nonce, encrypted data and MAC fields which has a maximum length of 266 bytes for each Encryption Frame. Since the LDPC coding is in units of 8-bytes, 6 bytes of padding are needed, resulting in 272 bytes to be coded. Each 8-byte unit get coded into a 16- byte LDPC codeword. For a maximum length Encryption Frame, there will be 544 bytes after coding. An 8-byte synch marker is prepended to the coded data and a 16 byte Tail and 8 bytes of idle are placed at the end of each Encryption Frame.

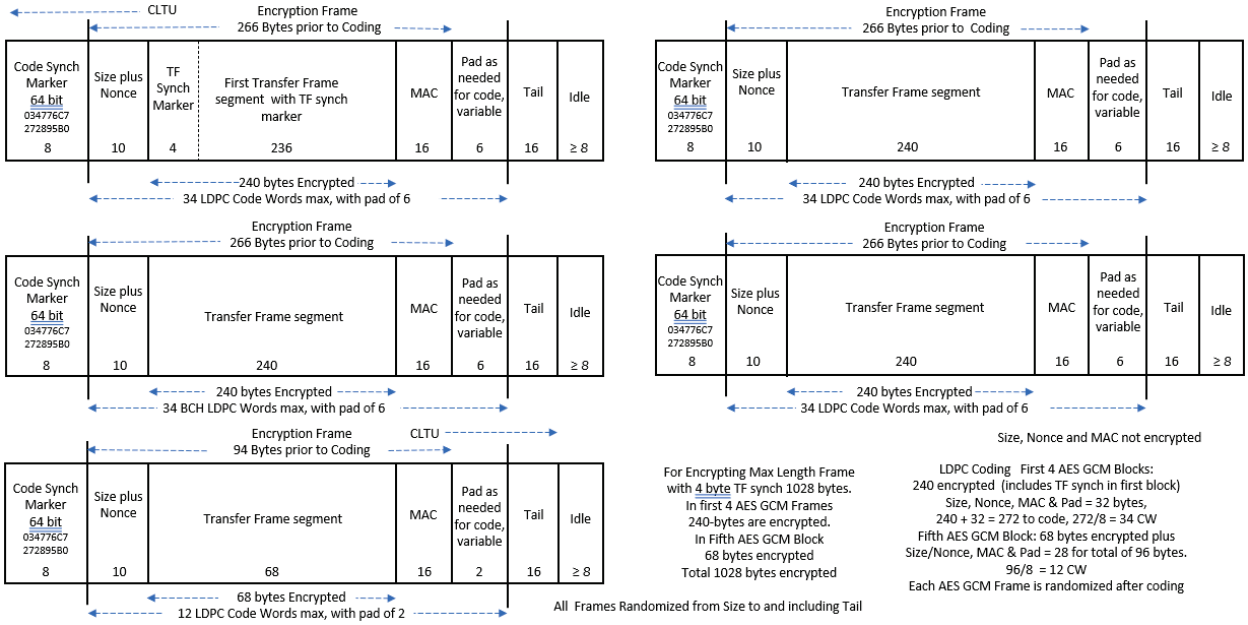

Figure 14. Use Case 5: Transfer Frame Maximum Length in 5 Small Encryption Frames, LDPC Coded

15.11 Use Case 6: Transfer Frame – Single Large Encryption Frame – LDPC (128, 64) Coded

This case covers a transfer frame bulk secured in a single large Encryption Frame, then LDPC (128, 64) coded.

For Use Case 6, the transfer frame shall have a CCSDS transfer frame synch marker prepended to the Transfer Frame.

Note: Figure 13 and the text associated with it applies for a minimum length transfer frame. Figure 15 shows the case when a large Encryption Frame is used which does not require that the Encryption Frame be limited to 266 bytes after encryption and authentication.

No matter the length of the transfer frame, a single secured and coded frame (byte stream) is used. The Maximum length transfer frame with a 4 byte CCSDS transfer frame synch marker has a length of 1028 bytes. The Size/Nonce and MAC fields are added to the Large Encryption Frame prior to coding. See Figure 15.

After encryption and authentication, an LDPC code is applied. It covers the Size, Nonce, encrypted data and MAC fields resulting in a maximum length of 1054 bytes. Padding may be required prior to coding to make the length a multiple of the code message length, k=8. For a maximum length transfer frame, padding of 2 bytes are needed, with the resulting length of 1056 bytes. Each 8-byte unit gets coded into a 16-byte LDPC codeword, resulting in up to 2112 bytes

after coding for a max length Transfer Frame. An 8-byte code synch mark is prepended to the coded data and at least 8 bytes of idle are placed at the end of the coded frame.

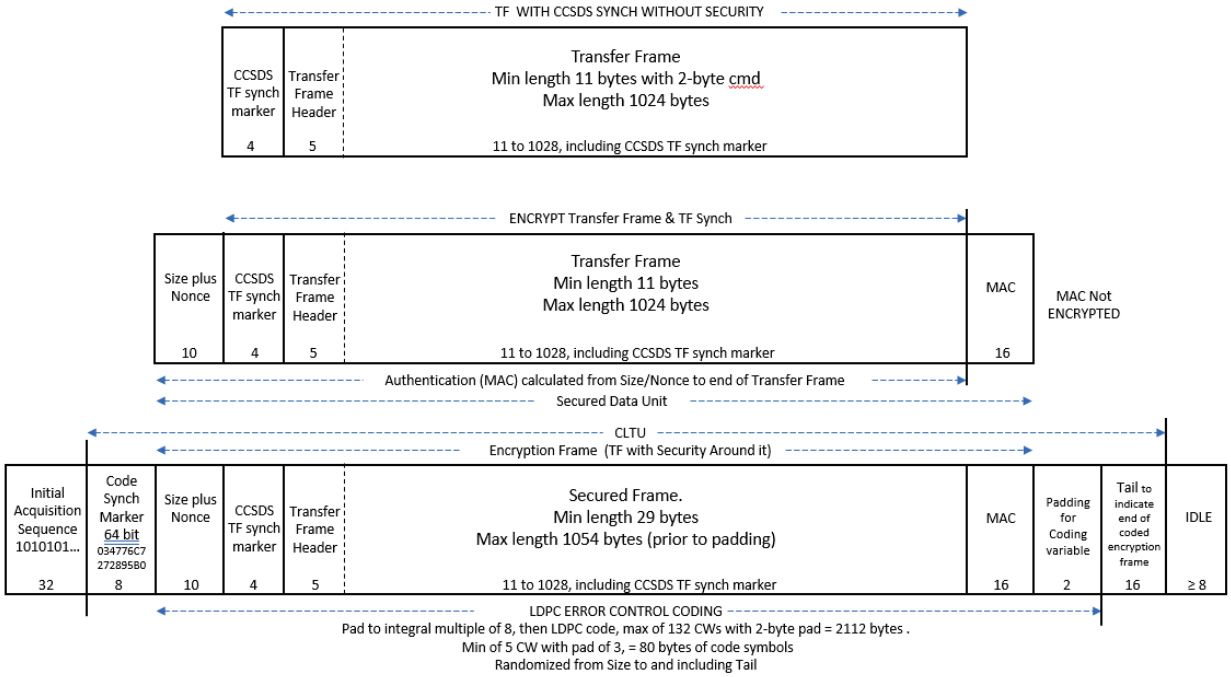

Figure 15. Use Case 6 - Transfer Frame Maximum Length in single Large Encryption Frame, LDPC Coded

15.12 Use Case 7: Transfer Frame – Single Large Encryption Frame LDPC (512, 256)

Use Case 7 not shown but similar to Use Case 6 with the longer (512, 256) LDPC code.

APPENDIX A – Number of Bits are Required for the Counter

We consider mission duration, the maximum data rate, and the shortest command size to calculate the maximum number of commands in the spacecraft lifetime. The shortest command CLTU would be a hardware command transfer frame of 7+1 bytes (1 BCH CW) preceded by a 2 byte Start Sequence and with an 8-byte Tail following the transfer frame for 18 bytes total. The CLTU would be preceded by a synch mark of 8 bytes and a Size + Nonce of 10 bytes. At the end of the CLTU is a MAC of 16 bytes and Tail of 8 bytes. We will use 64 bytes for the size after the ECC.

We will consider:

Authentication	16 bytes = 128 bits
Size of CLTU	64 bytes = 512 bits.
Maximum data rate	1.0 Mbps frames/sec = 1.0 Mbps/(512 b/frame) = 1953 frames/sec
Mission duration	10 years

Number of commands assuming continuous commanding

Number of frames = 1.0 Mbps x 10 years / (512 bits per frame)
= 1.0 Mbps x 10 x 365 x 24 x 3600 sec / (512 bpf)
= 6 x10^{11} frames

2^{40} = 1.1 x 10^{12} so 40 bits would be enough to cover a counter for 10 years at 1 Mbps.

To be conservative we have allocated 64 bits for the Nonce counter, and 16 bits for the Size field. The Size plus Nonce fields could have been limited to a total of 64 bits. With a Size field of 10 bits, a Nonce of 54 bit is more than enough.

APPENDIX B - Difference Between this Standard and CCSDS 231.0-B Section 5.2.1

In this bulk standard, the 64-bit synchronization marker common to all Cases is called a CSM. In the CCSDS 231.0-B- book, the synch marker is called a Start Sequence, for both the 16-bit EB90 and the 64-bit 034776C7272895B0 synchronization marker. See figures below from the CCSDS 231.0-B book. An alternate 64-bit CSM is also available in section 12.

In this bulk standard, coded secured Encryption Frames are considered, so the first use of the synch marker is to delimit the first codeword, hence the term CSM. The same CSM is used for the BCH and the LDPC codes. In the CCSDS book, the Start Sequence is different for the two code types.

In a heritage CLTU there is a Tail sequence that immediately follows the BCH coded transfer frame. In this standard that Tail remains and is internal to the security and coding. In this standard, there is a tail that follows the secured and coded data which may be a heritage CLTU or a transfer frame. The figure below is taken from the CCSDS 231.0-B- document and maintains that documents figure numbers.

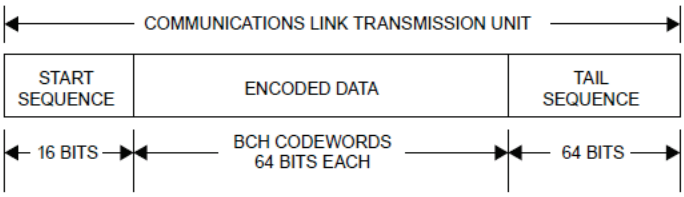

Figure 5-1: Components of the CLTU when BCH Coding Is Used

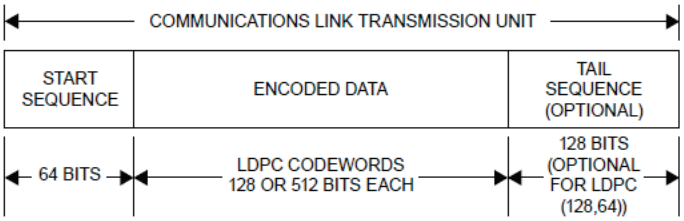

Figure 5-2: Components of the CLTU when LDPC Coding Is Used

In this standard, there is a Size and Nonce field at the beginning of the coded Data. In the CCSDS case, the size and nonce are part of a modified transfer frame.

APPENDIX C - Communication Operation Procedure-1, COP-1

This appendix states details of the COP-1 but the important point of this appendix is that COP-1 is completely independent of the Bulk security and functions at a higher protocol layer.

The COP-1 procedure [4] is a Go-Back-N ARQ procedure which, if used, consists of two synchronized procedures, a Frame Operating Procedure (FOP) operating on the ground and a Frame Acceptance and Reporting Mechanism (FARM) operating on the spacecraft (there is one COP-1 procedure per VCID for software commands only). Since the FOP is designed to automatically re-transmit sequence-controlled TC frames if they are not received on the spacecraft as well as send expedited frames which bypass the ARQ mechanism, its' operation may change the order of frames transmitted relative to the order they are input to the FOP. For example, if a series of sequence-controlled frames (type A) with frame sequence numbers #1 to #10 was transmitted followed by an expedited frame (type B) but the sequence-controlled frame #6 was not received, the FARM would discard frames #7 to #10 and request re-transmission beginning with frame #6. The FOP would follow transmission of the expedited frame with re-transmission of the sequence-controlled frames #6 to #10. The order of frames output by the FARM in this example would then be sequence-controlled frames #1 to #5, then the expedited frame, followed by the sequence-controlled frames #6 to #10.

Notice that if the nonce values for the frames in this scenario were assigned by the security function (for this example starting at 1) prior to being input to the FOP, the output of the FARM to the security function on the spacecraft would result in nonce values incrementing from 1 to 5, then jumping to 11, and then going back to 6 and incrementing to 10. The bypass frame would set the last received nonce value in the security function on the spacecraft at 11 which would result in frames #6 to #10 failing authentication. For this reason, nonce values must be assigned after the FOP procedure on the ground and not before. TC frames presented to the sending end of GSFC 8012 in figure 6 should have already been processed by the FOP. Similarly frames output at the receiving end of figure 6 should then be processed by the FARM. This order of functions ensures that the nonce values increment with the order of transmission and monotonically increase with reception by the spacecraft (a missing frame would result in the nonce value increasing by 2 instead of 1 but this still satisfies the rule that the nonce value must increase) to avoid false authentication failures.

APPENDIX D - Clear Mode

The NASA Center of Record frowns on having a clear mode for flight. When the primary command link is encrypted, any backup/contingency command link shall include authentication at minimum per NASA-STD-1006 SSPR-2. Clear-and-unauthenticated commanding is not permitted in operations except under TA-approved emergency waivers with documented risk acceptance and reversion plans.

When using Bulk security, implementing the option of a clear mode in addition to a secure mode is different from how it is done with CCSDS security. Some NASA centers do not have a clear mode for many (most, all) of the bulk secured satellites. If a center wants a clear mode, they will have to have the logic to bypass the security (encryption, authentication, or both). When in secure mode, an encrypted command to change to clear mode would be required. When in clear mode, “clearly” a clear mode command to change to secure mode would be required. In the case of a spacecraft emergency, the system designer might have the logic set to go to clear mode. This would allow shorter commands with no security overhead but the coding for error correction would likely be retained. The logic might even change from one ECC to another when in clear mode.

For several missions, with no clear mode, in an emergency the security changes to a default key. As the emergency is recovered, an encrypted command to change to one of the other operational keys is sent.

APPENDIX E - LDPC Details

***NOTE: Appendix E is essentially section 4.4 and 4.5 from CCSDS 231.0-B-4, reference [2]**

Fill Data

E1 If the Transfer Frame(s) and AAD to be transmitted in a CLTU do not fit exactly within an integral number of BCH or LDPC codewords, then 'fill' bits shall be appended to the (last) Transfer Frame to be transmitted in the CLTU until an integral number of BCH or LDPC codewords is completed.

E2 The pattern of the fill shall consist of a sequence of alternating 'ones' and 'zeros', starting with a 'zero'.

E3 The fill data shall be added before encoding. It shall be encoded and randomized with the Transfer Frame or CLTU.

NOTE – The Synchronization and Channel Coding Sublayer may require the introduction of these fill data in the encoding process; they are not removed by the decoding process. Removal of fill is the responsibility of the sublayer above, which delimits the end of the Transfer Frame(s) and discards extraneous bits (e.g., fill).

Decoding Procedure

E4 An LDPC code should be decoded using 'soft symbols', rather than the binary 'hard symbols' typically used for a BCH code.

NOTE – This provides a performance improvement of about 2 dB but depends on a Receiver symbol synchronizer that can produce soft outputs. This modification is not mandatory, however, since a belief propagation decoder can also operate on binary symbols.

APPENDIX F - Options for Compatibility with CCSDS SDLS

This standard contains two fields, the Size and Nonce, that come immediately after the synchronization marker and before either the heritage CLTU or TC transfer frame, that are authenticated but not encrypted. In this standard these two fields take up 10 bytes. As an option, a mission may decide to define a larger number of unencrypted bytes between the synchronization marker and the beginning of the encrypted portion. Should a mission choose this option, the number of unencrypted bytes **shall** be a managed parameter that is fixed for that mission or mission phase. This option makes this standard just as 'interoperable' as CCSDS SDLS but also gives missions that don't require 'interoperability' the option not to use it.

There are several use cases that require a mission to need to have initial fields of the CCSDS Transfer Frame unencrypted. Some examples are:

1. To allow a receiving node to demultiplex and distribute messages to equipment belonging to different space agencies or different projects without having to decrypt the messages.

2. To allow each of several satellites in a multi satellite beam to examine the destination SCID in the CCSDS Transfer Frame header and determine which messages are for them (avoiding a security failure and excessive reporting by attempting to process those that are not intended for that spacecraft).

Note: Each space agency equipment or satellite will likely be using different encryption keys. In the GCM protocol, the authentication HASH key is based on the encryption key so authentication cannot be done without some of the security information. In both examples above, the error correction decoding would allow the unencrypted fields to be transferred essentially error free and allow demultiplexing, but the authentication would likely not be done until the message arrives at its destination.

When transferring a heritage CLTU, when the mission decides to not encrypt part of the Transfer Frame header, it makes sense to also not encrypt the CCSDS Start Sequence that immediately follows the Nonce and occurs prior to the CCSDS Transfer Frame header.

For a TC or AOS Transfer Frame, if the mission decides to keep the SCID unencrypted, then the first two bytes of that Transfer Frame header are not encrypted. Instead of the maximum 240 encrypted bytes in a Small Encryption Frame, the number of encrypted bytes will become 238 bytes (minus two for the SCID).

For those missions which decide to keep the entire TC Transfer Frame header unencrypted, i.e., 5 bytes, then the maximum number of unencrypted bytes would be 7. Similarly, for USLP to do its configurable Transfer Frame header, the maximum number of unencrypted bytes would be 16.

Authentication is unchanged, starting at the end of the synchronization marker, but there is now an increase in the "AAD" and a reduction in the number of encrypted bytes.

Note: Leaving the SCID and possibly the VCID unencrypted while retaining the ability of the intended user to authenticate it provides several benefits. Intermediate nodes in a network, for example nodes providing international cross-support, can use that information to route the encrypted frame without having access to the encryption keys of the user. In addition, multiple users sharing a physical channel (such as when we have multiple satellites per aperture, MSPA), each with their own unique encryption keys, can receive an Encryption Frame intended for a different user and discard it based on the SCID without reporting it as a security incident. The user with the SCID matching the SCID in the frame, accepts only those frames which pass authentication and reports as a security incident, frames which fail authentication.

APPENDIX G – Considerations for Telemetry Encryption

G1 Introduction

This appendix is an outline only.
There is a need to secure telemetry data in order to protect spacecraft state of health, command-echo information, as well as simply to add security measures to deny any information that NASA wants to protect. It may become useful to secure at least some of the telemetry data. This section outlines a method. It is assumed that all frames have the same length, USLP not considered.

Encrypting

On some satellites the command security is done in the transponder. When the only telemetry path is through the transponder, the telemetry encryption can also be done in the transponder, which would simplify security boundaries and key management. But often telemetry is multiplexed with science and operational data on a separate link that is otherwise un-secured.

When only a portion of the telemetry needs to be secured, such as engineering and housekeeping data, there must be a way to distinguish the secured data from the plaintext data. If the chosen method is to use the VCID, the VCID must be unencrypted. In this case encryption would start with the third byte of the AOS transfer frame header, leaving the first two bytes of the header unencrypted. In addition to the 6-bit VCID, this would expose the 2-bit version number and the least significant 8 bits of the spacecraft ID. This partially defeats the value of bulk security, but it still has value since no internal change to the AOS transfer frame is required.

Pros for encrypting the entire downlink

Easiest to implement

All science and housekeeping data is protected

Cons for encrypting the entire downlink.

Not necessary to encrypt science date that will be made public.

Encrypting data that is made public could aid cryptanalysis.

G1.1 Considerations for Securing Telemetry

1. The spacecraft should maintain a count (used as a nonce) of secured frames in non-volatile memory.
2. That count should be incremented by 1 every time a secured frame is generated and transmitted.
3. The spacecraft should generate a counter block based on the nonce with an additional fixed portion.
4. The fixed portion should be maintained in a non-volatile memory.
5. The Nonce and fixed portion of the counter block should be settable via secured command.
6. The spacecraft should be capable of sending either unsecured or a secured telemetry.
7. The selection of security on a given channel should be controllable via a secured command.
8. Keys used for telemetry should be separate from the keys used for command.

9. The expected cypher is AES-256 but this GSFC standard allows other cyphers to be used in the future as the cryptology field develops.
10. The Insert Zone (if used) is compatible with fixed length transfer frames and should not be encrypted.

Note: when the Insert Zone is used for real time information like voice or time transfer, short or fixed latency algorithms may be required, or plaintext may be needed. See CCSDS 732.0-B-5 AOS Space Data Link Protocol Oct 2025.pdf

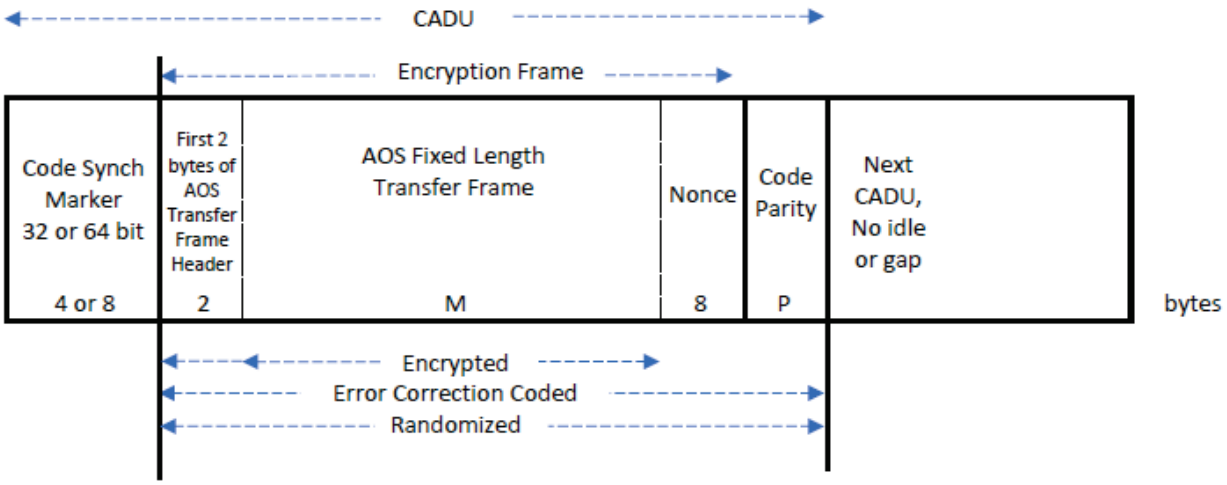

Figure 16. Bulk Secured and Coded AOS Telemetry

The nonce is placed at tail of AOS transfer frame to allow existing ground receivers to do virtual channel sorting with no change to the ground receivers. Decryption of encrypted frames would be done after the data is transferred to the operations center.