

Draft Recommendation for
Space Data System Standards

|  |
| --- |
| CCSDS Streamlined Bundle Security Protocol Specification |

Draft Recommended Standard

April 2020

AUTHORITY

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Issue: | , Issue 2 |  |
|  | Date: | April 2020 |  |
|  | Location: | Not Applicable |  |
|  |  |  |  |

**(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)**

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the e-mail address below.

This document is published and maintained by:

CCSDS Secretariat

National Aeronautics and Space Administration

Washington, DC, USA

E-mail: secretariat@mailman.ccsds.org

STATEMENT OF INTENT

**(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF INTENT:)**

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

o Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.

o Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:

 -- The **standard** itself.

 -- The anticipated date of initial operational capability.

 -- The anticipated duration of operational service.

o Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

FOREWORD

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in the *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be sent to the CCSDS Secretariat at the e-mail address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

* Agenzia Spaziale Italiana (ASI)/Italy.
* Canadian Space Agency (CSA)/Canada.
* Centre National d’Etudes Spatiales (CNES)/France.
* China National Space Administration (CNSA)/People’s Republic of China.
* Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
* European Space Agency (ESA)/Europe.
* Federal Space Agency (FSA)/Russian Federation.
* Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
* Japan Aerospace Exploration Agency (JAXA)/Japan.
* National Aeronautics and Space Administration (NASA)/USA.
* UK Space Agency/United Kingdom.

Observer Agencies

* Austrian Space Agency (ASA)/Austria.
* Belgian Federal Science Policy Office (BFSPO)/Belgium.
* Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
* China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
* Chinese Academy of Sciences (CAS)/China.
* Chinese Academy of Space Technology (CAST)/China.
* Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
* Danish National Space Center (DNSC)/Denmark.
* Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
* Electronics and Telecommunications Research Institute (ETRI)/Korea.
* European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
* European Telecommunications Satellite Organization (EUTELSAT)/Europe.
* Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
* Hellenic National Space Committee (HNSC)/Greece.
* Indian Space Research Organization (ISRO)/India.
* Institute of Space Research (IKI)/Russian Federation.
* Korea Aerospace Research Institute (KARI)/Korea.
* Ministry of Communications (MOC)/Israel.
* Mohammed Bin Rashid Space Centre (MBRSC)/United Arab Emirates.
* National Institute of Information and Communications Technology (NICT)/Japan.
* National Oceanic and Atmospheric Administration (NOAA)/USA.
* National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
* National Space Organization (NSPO)/Chinese Taipei.
* Naval Center for Space Technology (NCST)/USA.
* Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
* Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
* South African National Space Agency (SANSA)/Republic of South Africa.
* Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
* Swedish Space Corporation (SSC)/Sweden.
* Swiss Space Office (SSO)/Switzerland.
* United States Geological Survey (USGS)/USA.

PREFACE

This document is a draft CCSDS Recommended Standard. Its ‘Red Book’ status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document’s technical content.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

DOCUMENT CONTROL

|  |  |  |  |
| --- | --- | --- | --- |
| **Document** | **Title** | **Date** | **Status** |
|  | , ,  |  | Current draft |
|  |  |  |  |
|  |  |  |  |

CONTENTS

Section Page

[1 Introduction 1-1](#_Toc39246834)

[1.1 PURPOSE 1-1](#_Toc39246835)

[1.2 SCOPE 1-1](#_Toc39246836)

[1.3 Applicability 1-1](#_Toc39246837)

[1.4 Rationale 1-2](#_Toc39246838)

[1.5 ORGANIZATION OF THIS RECOMMENDED STANDARD 1-2](#_Toc39246839)

[1.6 Definitions 1-3](#_Toc39246840)

[1.7 NOMENCLATURE 1-4](#_Toc39246841)

[1.8 References 1-5](#_Toc39246842)

[2 Overview 2-1](#_Toc39246843)

[2.1 Concept of the Streamlined Bundle Security Protocol 2-1](#_Toc39246844)

[2.2 Features of the Streamlined Bundle Security protocol 2-2](#_Toc39246845)

[3 Service Definition 3-1](#_Toc39246846)

[3.1 Overview 3-1](#_Toc39246847)

[3.2 Functions at Security Sources 3-1](#_Toc39246848)

[3.3 Functions at Security Verifiers 3-4](#_Toc39246849)

[3.4 Function at Security Acceptors 3-6](#_Toc39246850)

[4 Protocol Specification 4-1](#_Toc39246851)

[4.1 Protocol Data Units 4-1](#_Toc39246852)

[4.2 Security Protocol Procedures 4-6](#_Toc39246853)

[5 Managed Parameters 5-1](#_Toc39246854)

[5.1 Overview 5-1](#_Toc39246855)

[5.2 Requirements 5-1](#_Toc39246856)

[ANNEX A Protocol Implementation Conformance Statement (PICS) Proforma (normative) A-1](#_Toc39246857)

[ANNEX B Security, SANA, and Patent Considerations (Informative) B-1](#_Toc39246858)

[ANNEX C Informative References (Informative) C-1](#_Toc39246859)

[ANNEX D Abbreviations (Informative) D-1](#_Toc39246860)

[ANNEX E Baseline Mode (Informative) E-1](#_Toc39246861)

Figure

[Figure 2‑1: Bundle Nodes Sit at the Application Layer of the Internet Model 2-1](#_Toc39247067)

[Figure 2‑2 Security-aware BPAs operate on security blocks in a bundle. 2-2](#_Toc39247068)

[Figure 4‑1: The Generic Security Block Structure 4-2](#_Toc39247069)

[Figure 4‑2: Security Block Flags 4-3](#_Toc39247070)

[Figure 4‑3: The Canonical Form of the Primary Bundle Block 4-8](#_Toc39247071)

Table

[Table 5‑1 SBSP Managed Parameters 5-1](#_Toc39246961)

[Table B‑1 SBSP Security Block Types B-3](#_Toc39246962)

[Table B‑2 Security Context Identifier Values B-4](#_Toc39246963)

[Table E‑1 Security Context Parameters and Results E-5](#_Toc39246964)

# Introduction

## PURPOSE

The purpose of this Recommended Standard is to specify a Streamlined Bundle Security Protocol (SBSP) providing the minimal security service necessary to secure CCSDS Bundle Protocol (reference [1]) transactions across a Delay-Tolerant Network (DTN). The SBSP provides BP extension blocks, and associated procedures, that may be added to a BP bundle. These extension blocks provide a structured method for applying data integrity and/or data confidentiality to non-SBSP blocks within a bundle.

## SCOPE

This Recommended Standard defines the SBSP in terms of:

a) the protocol data units employed by the service provider; and

b) the procedures performed by the service provider.

It does not specify:

a) individual implementations or products;

b) the implementation of service interfaces within real systems;

c) the methods or technologies required to perform the procedures; or

d) the management activities required to configure and control the service.

This Recommended Standard does not mandate the operational use of any particular cryptographic algorithm with the SBSP. Reference [2] provides a listing of algorithms recommended by CCSDS; any organization should conduct a risk assessment before choosing to substitute other algorithms. Annex E (non-normative) defines baseline implementations suitable for a large range of space missions.

The protocol specified here applies only to the CCSDS Bundle Protocol and does not interact with other CCSDS protocols.

## Applicability

This Recommended Standard applies to the creation of Agency standards and for secure data communications over space networks between CCSDS Agencies in cross-support situations.

The Recommended Standard includes comprehensive specification of the service for inter-Agency cross-support. It is neither a specification of, nor a design for, real systems that may be implemented for existing or future missions.

The Recommended Standard specified in this document is to be invoked through the normal standards programs of each CCSDS Agency, and is applicable to those missions for which interoperability and cross-support based on capabilities described in this Recommended Standard is anticipated. Where mandatory capabilities are clearly indicated in sections of the Recommended Standard, they must be implemented when this document is used as a basis for interoperability and cross-support. Where options are allowed or implied, implementation of these options is subject to specific bilateral cross-support agreements between the Agencies involved.

NOTE – The SBSP adds extension blocks into a BP bundle. BP Agents that implement the SBSP may process these extension blocks in accordance with this Recommended Standard. BP agents that do not implement the SBSP may process these extension blocks as generic blocks in accordance with (reference [1]).

## Rationale

The goals of this Recommended Standard are to:

1. provide a standard method of applying block-specific security for bundle transport, independent of the underlying cryptographic algorithms employed by any particular space mission;
2. preserve compatibility with existing CCSDS BP version 6 formats so that, where appropriate, legacy bundle processing infrastructure may continue to be used without modification; and
3. facilitate the development of common commercial implementations to improve interoperability across agencies.

## ORGANIZATION OF THIS RECOMMENDED STANDARD

This Recommended Standard is organized as follows:

Section 1 presents the purpose, scope, applicability, and rationale of this Recommended Standard and lists the conventions, definitions, and references used throughout the document.

Section 2 (informative) provides an overview of the SBSP.

Section 3 (normative) defines the services provided by the protocol entity.

Section 4 (normative) specifies the protocol data units provided for these services and the procedures employed by the service provider.

Section 5 (normative) specifies the constraints associated with SBSP services.

Section 6 (normative) lists the managed parameters associated with these services

Section 7 (normative) specifies how to verify an implementation’s conformance with the SBSP.

Annex A (normative) provides a Protocol Implementation Conformance Statement (PICS) proforma for the SBSP.

Annex B (informative) provides an overview of security, SANA registry, and patent considerations related to this Recommended Standard

Annex C (informative) provides a glossary of abbreviations and acronyms that appear in the document.

Annex D (informative) provides a list of informative references.

Annex E (informative) defines baseline implementations suitable for a large range of space missions.

## Definitions

For the purposes of this document, the following definitions apply. For additional terms and definitions of some of the terms reference by these definitions, see [4, 5, 8].

NOTE – Generic definitions are presented in this section only if they have been augmented with an SBSP-specific definition.

**Block source**: The BPA that adds a block to a bundle. The block source of the primary block and the payload block are always considered to be the bundle source.

**Cipher suite**: a set of one or more algorithms providing integrity and/or confidentiality services. Cipher suites may define user parameters (e.g. secret keys to use) but do not provide values for those parameters. Additionally, cipher suites do not codify session information and context that may be required when operating in a DTN.

**Cipher suite result:** The set of outputs generated from one or more algorithms in a specific cipher suite as part of applying, verifying, or removing an integrity or confidentiality service.

**Plain text:** Data that is input to an encryption or integrity algorithm existing in a cipher suite; the block-type-specific contents of a target block prior to the application of a confidentiality service.

NOTE – Plain text is not a synonym for clear text. Clear text refers to user data that has never been encrypted, whereas plain text may refer either to clear text or to data which has been encrypted by an application prior to its inclusion in a BP block.

**Cipher text:** The enciphered plain text produced by a cipher suite; the union of the block-type-specific contents of a target block after the application of a confidentiality service with any overflow bytes stored in the corresponding SBSP confidentiality security block.

**Security acceptor:** The security acceptor of an SBSP security block is the BP agent required (by agent policy or configuration) to process and remove the SBSP security block from a bundle.

NOTE – The BP agent serving as the security acceptor of a SBSP extension block may not be known at the time the SBSP extension block is created at a bundle source. Therefore, determination of what BP agent serves as an acceptor is left to each BP agent in the network as it receives bundles. A bundle destination is considered the default security acceptor of any SBSP security blocks remaining in the bundle at the time it reaches its destination.

**Security-aware Bundle Protocol Agent (BPA)**: A security-aware BPA is any BPA that provides SBSP services. A BPA that can add SBSP security blocks to a bundle, verify security blocks in a bundle, or process security blocks at their security acceptor.

**Security block**: A single instance of a SBSP extension block in a bundle.

**Security context**: The set of assumptions, algorithms, configurations, and policies used to implement security services. The union of a cipher suite, cipher suite parameters, bundle-specific configuration, and node-specific configuration and policy.

**Security operation**: The application of a security service to a target block, notated as OP(security service, target block), e.g., OP(integrity, primary block) or OP(confidentiality, payload). A single security operation is implemented with a single security block.

**Security service**: A process that gives some protection to a target block. For example, this Recommended Standard defines security services for integrity and confidentiality.

**Security source:** The security source of an SBSP security block is the BPA that created the bundle containing the SBSP security block; the bundle source of the bundle containing the security block.

**Security verifier:** The security verifier of an SBSP security block is a security-aware BPA that verifies the integrity of the block prior to forwarding the block. A security verifier may not be the security block’s security source or security acceptor.

**Target block**: A block within a bundle that receives a security service as part of a security operation.

## NOMENCLATURE

### Normative Text

The following conventions apply for the normative specifications in this Recommended Standard:

1. the words ‘shall’ and ‘must’ imply a binding and verifiable specification;
2. the word ‘should’ implies an optional, but desirable, specification;
3. the word ‘may’ implies an optional specification;
4. the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.

### Informative Text

In the normative sections of this document, informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

* Overview;
* Background;
* Rationale;
* Discussion.

## References

The following publications contain provisions which, through reference in this text, constitute provisions of this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this Recommended Standard are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

[] *CCSDS Bundle Protocol Specification*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 734.2-B-1. Washington, D.C.: CCSDS, September 2015.

[] *CCSDS Cryptographic Algorithms*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 352.0-B-2. Washington, D.C.: CCSDS, August 2019.

[] *Space Missions Key Management Concept*. Issue 1. Report Concerning Space Data System Standards (Green Book), CCSDS 350.6-G-1. Washington, D.C.: CCSDS, November 2011.

[] *Information Security Glossary of Terms*. Issue 1. Report Concerning Space Data System Standards (Green Book), CCSDS 350.8-G-1. Washington, D.C.: CCSDS, November 2012.

[] CCSDS Terms. SANA Registry https://sanaregistry.org/r/terms.

[] W. Eddy and E. Davies. *Using Self-Delimiting Numeric Values in Protocols*. RFC 6256. Reston, Virginia: ISOC, May 2011.

[] T. Berners-Lee, R. Fielding, and L. Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. STD 66. Reston, Virginia: ISOC, January 2005.

 [8] R. Shirey. *Internet Security Glossary, Version 2*. RFC 4949. Reston, Virginia: ISOC, August 2007.

# Overview

## Concept of the Streamlined Bundle Security Protocol

The SBSP is a data processing method by which space missions can apply integrity or confidentiality to the individual blocks that comprise a BP bundle. The SBSP data units are codified as extension blocks within the BP and only exist in the context of a bundle.

This type of security service is needed because the stressed nature of the BP operating environment imposes unique conditions where usual transport security mechanisms may not be sufficient. For example, the store-carry-forward nature of the network may require protecting data at rest, preventing unauthorized consumption of critical resources such as storage space, and operating without regular contact with a centralized security oracle (such as a certificate authority).

Figure 2‑1 below illustrates the streamlined operational concepts associated with the SBSP. In this illustration, both security-aware (BN1, BN3, BN4) and non-security-aware (BN2) BPAs are encountered by bundles. The security-aware BPAs take on the roles of security source, verifier, and acceptor as they receive different bundles. A security-aware BPA may take on multiple roles, as evidenced by BN3, which serves as the security source for bundle 3, security verifier for bundle 2, and security acceptor for bundle 1.

Each security service is added to a bundle for a target block by a security source and removed from the bundle by a security acceptor. Some bundles may additionally encounter security verifiers, as bundle 2 in figure 2-1 does.



Figure 2‑1 : Bundle Nodes Sit at the Application Layer of the Internet Model

Figure 2-2 shows the operation of a security source, verifier, and acceptor on a security block that provides integrity for its target block in a bundle.

In this figure, steps 1 and 2 occur at the security source. At step 1, the security source identifies a target block in the bundle which is required by policy to have integrity applied before the bundle is transmitted. At step 2, the security source computes an integrity result and adds it to the bundle in the form of a SBSP security block.

Steps 3 and 4 occur at the security verifier node. Step 3 shows the identification of the node as the security block’s security verifier. The security verifier verifies the integrity of the target of the security block at step 4.

Steps 5, 6, and 7 occur at the security acceptor. Step 5 shows the identification of the node as the security block’s security acceptor. The block is processed in order to verify the integrity of its target block in step 6. The security block is removed from the bundle in step 7.



Figure 2‑2 Security-aware BPAs operate on security blocks in a bundle.

## Features of the Streamlined Bundle Security protocol

### General

The SBSP is “streamlined” because it provides the minimal set of security services to assure the confidentiality or integrity of blocks in a bundle for common operational scenarios [C1,C2].

Specifically, the SBSP provides the following capabilities.

1. A security block may be added to a bundle by the same BPA that added its target block to the bundle;
2. A security block may be added to a bundle if there is only one instance of the target block’s block type in the bundle;
3. A security block may be added to a bundle if its target block is not the target of any other security block in the bundle.
4. Multiple security blocks of the same block type may exist in a bundle.

NOTE – It is anticipated that future security specifications will define additional security extension blocks and processing associated with those blocks to provide more complex security services for more complex operating scenarios.

The CCSDS systems engineering area’s security working group (SEA-SEC) has developed a suite of documents to assist mission planners with planning & assessment, design, and implementation of security measures for CCSDS missions. In that context, this specification is an implementation document for which a companion design document (Green Book) is planned.

### Block-Level granularity

SBSP security blocks contain a target block identifier. The ability to “target” blocks within a bundle enables these different blocks to have different security services applied to them.

NOTE – Blocks within a bundle represent different types of information and may need different types of security applied to them. Applying a single level and type of security across an entire bundle fails to recognize that blocks in a bundle represent different types of information with different security needs.

### Deterministic security sources

The SBSP couples the creation of target blocks and the creation of security services for those blocks. Specifically, a security-aware BPA may act as the security source for any security operation that targets a block it has added to the bundle.

NOTE – Only the creator of block in a bundle may add a security service for that block. If some other node in the network were to apply a security service to a block after it had been transmitted, the contents of the block could have been altered between the time the bundle was transmitted and the time it was received by a downstream node.

NOTE – The Bundle Protocol allows extension blocks to be added to a bundle at any time during its existence in the DTN. Therefore, a downstream node may add a security service to an extension block that it adds to the bundle, even though that node is not the source of the bundle.

NOTE – The security source can be inferred in circumstances where the creator of its target block can be inferred; For example, the security source of the primary block and the payload block must be the bundle source because only the bundle source can add these blocks to the bundle.

### Adaptive security policy

SBSP blocks do not specify the BPAs that serve as their security verifiers or acceptors.

NOTES

1 The topology of a DTN is expected to evolve over time and the security capabilities of bundle nodes may change after a bundle has been created by, and transmitted from, its bundle source.

2 BPAs that implement the SBSP determine whether they should be processing security blocks as part of their local BPA configuration and policy. The determination of which nodes are security acceptors and which nodes are security verifies is made by the nodes themselves, and not by the SBSP.

### Multiple Security Contexts

SBSP defines security contexts as a way of differentiating multiple uses of cipher suites based on operational scenarios in a DTN.

NOTE – For example, one mission may use a 256-bit Advanced Encryption Standard (AES) cipher suite for confidentiality with a negotiated security association identifier (SAID). In this case, at most the SAID must be communicated to process security at a security acceptor. Another mission may use the same 256-bit AES cipher without the ability to negotiate (or sustain) a security association. In this case, cipher suite parameters may be encoded with a symmetric key and included in the security block. In both cases, the same cipher suite is used, but the security context surrounding the cipher suite is very different.

SBSP security blocks identify the security context used in the generation of their cryptographic materials.

NOTES

1 This feature allows different SBSP security blocks to use different security contexts. This may include contexts that utilize different cipher suites or the same cipher suites operating in different modes or with different configurations.

2 CCSDS recommended cipher suites as specified in [2] should be used in the definition of security contexts unless a specific reason exists that does not allow them.

# Service Definition

## Overview

This section provides the service definition for the SBSP.

The services that the SBSP provides to the Bundle Protocol are represented as functions and defined as follows.

(a) ApplyIntegrity

(b) ApplyConfidentiality

(c) VerifyIntegrity

(d) AcceptIntegrity

(e) AcceptConfidentiality

The definitions of these functions are logical and do not pre-suppose any specific BPA, security context definition, or cipher suite implementation. Function behaviors do not presuppose any configuration approach, policy approach, or method for generating cryptographic material.

This specification allows for the concurrent execution of these functions such that one function can be started prior to some other function completing, regardless of whether these functions operate on different blocks within a bundle or different bundles within the BPA.

The parameters for these functions are documented in an abstract sense and specify the information passed between the BPA entity that calls the function and the SBSP entity that executes the function. The way in which a specific implementation makes this information available is not constrained by this specification. In addition to the parameters specified in this section, an implementation may provide other parameters on the function interface (e.g., parameters for controlling the service, monitoring performance, diagnosis, and so on).

## Functions at Security Sources

### Overview

A SBSP security service is applied to a target block within a bundle by the BPA serving as the security source of that target block. These services are applied using two functions: ApplyIntegrity and ApplyConfidentiality.

#### ApplyIntegrity Function

The ApplyIntegrity Function adds an integrity mechanism to a target block within a bundle at the BPA inserting the target block into the bundle. It is called by the BPA after the target block has been added to the bundle.

The input parameters of this function include the bundle containing the block whose integrity is being protected, the identifier for the target block, the security context to be used to generate an integrity protection mechanism, and the set of parameters associated with that security context.

When called, this function provides all input parameters to the selected security context, captures the results generated by the security context, and inserts these results into the bundle in the form of a new security block.

When the ApplyIntegrity Function has completed the processing, it returns resulting data to the caller in the return parameter, the ApplyIntegrity Return.

#### ApplyConfidentiality Function

The ApplyConfidentiality Function generates cipher text which replaces the plain text contents of a target block within a bundle at the BPA inserting the target block into the bundle. It is called by the BPA after the target block has been added to the bundle.

The input parameters of this function include the bundle containing the target block whose contents are being protected, the identifier for the target block, the security context used to generate the appropriate cryptographic material, and the set of parameters associated with that security context.

When called, this function provides all input parameters to the selected security context, captures the results (including cipher text) generated by the security context, and inserts these results into the bundle.

NOTE – The selected security context must generate an integrity result for the created cipher text so that the downstream application of the AcceptConfidentiality Function can verify that the cipher text has not changed in transit.

When the ApplyConfidentiality Function has completed the processing, it returns resulting data to the caller in the return parameter, the ApplyConfidentiality Return.

#### Special Considerations

Because the ApplyConfidentiality function is required to provide integrity for its cipher text it would be redundant to call both the ApplyIntegrity and ApplyConfidentiality functions for the same target block in a bundle. Therefore, a security source for the target block must choose whether the target block receives plain text integrity or the combination of plain text confidentiality and cipher text integrity.

NOTE – The ApplyIntegrity and ApplyConfidentiality functions may be called multiple times for different target blocks in a bundle.

### Input Parameters

#### Overview

The ApplyIntegrity and ApplyConfidentiality functions take the same types of parameters. This section describes the parameters in general and notes, per parameter, if there is any difference in interpreting the parameter for integrity or confidentiality.

#### Target Bundle

The Target Bundle parameter shall consist of the bundle containing one or more blocks for which a SBSP service is being applied. The calling BPA must be the BPA inserting the target block into the target bundle.

NOTE – The bundle is also provided in the return value from the integrity and confidentiality functions because these functions modify the contents of the target bundle.

#### Target Block Type

The Target Block Type parameter uniquely identifies the target block within the target bundle that is being used by the integrity or confidentiality functions.

NOTE – The target block type must be the only block type of its kind in the bundle to avoid any ambiguity in identifying the proper target of the security block.

#### Security Context Identifier

The Security Context Identifier parameter shall identify the security context used to generate cryptographic material associated with the ApplyIntegrity and ApplyConfidentiality functions.

NOTE – In the context of ApplyConfidentiality, this parameter must reference an “authenticated encryption with associated data” (AEAD) cipher suite.

#### Security Context Parameters

The Security Context Parameters parameter shall consist of a set of parameters associated with the security context identified by the Security Context Identifier. These parameters must be provided as inputs to the algorithms used to produce cryptographic material for the bundle.

NOTE – In addition to being input to the suite of algorithms, some of these parameters may also be recorded in the SBSP security blocks added to the bundle for use by downstream security verifiers and security acceptors.

### Return Parameters

#### ApplyIntegrity Return

The ApplyIntegrity Return shall consist of a new security block to be added to the target bundle. This new security block shall include the outputs of the security context, to include results computed by the integrity mechanism. The calling BPA inserts the new security block after the primary block and before the payload block.

NOTE – The order in which the new security block returned by the ApplyIntegrity Return is added to the target bundle, as it relates to other extension blocks in the bundle, is not relevant to security processing.

#### ApplyConfidentiality Return

The ApplyConfidentiality Return shall consist of both a new security block added to the target bundle and an updated version of the target block.

The updated target block will have its block-type-specific data field replaced with the cipher text generated by the selected security context. If the length of the cipher text produced by the security context is larger than the length of the original target block’s block-type-specific data, then only a portion of the cipher text will be placed into the target block – specifically as much cipher text will be copied so as to not alter the length of the target block. Additional bytes of cipher text will be considered overflow and stored in the new security block added by the ApplyConfidentiality function.

The new security block shall include a subset of security context parameters used to generate cryptographic materials, any additional authenticated data produced by the security context (to include integrity check values), and any cipher text overflow.

The calling BPA inserts the new security block after the primary block and before the payload block. The calling BPA replaces the original target block with the updated target block provided by this Return.

NOTE – The order in which the new security block and the updated target block are added to the bundle, relative to other extension blocks in the bundle, is not relevant to security processing.

## Functions at Security Verifiers

### Overview

The SBSP integrity service may be verified by security-aware BPAs residing on waypoint nodes. This service is applied using the function VerifyIntegrity.

NOTE – Security verifiers check the plain text integrity of blocks in a bundle for the purpose of removing blocks (and bundles) from the network if they appear to have been corrupted or otherwise modified. Verifiers do not similarly check whether confidentiality is successfully applied as that usually requires decryption and decryption outside of a security acceptor is not supported by the SBSP.

NOTE – The security verifier role is distinct from the security acceptor role. For any given security block, a BPA may not be both a security verifier and security acceptor. If the BPA is the security acceptor, then the AcceptIntegrity function is used instead of the VerifyIntegrity function.

#### VerifyIntegrity Function

The VerifyIntegrity Function is defined for a security-aware BPA serving as the security verifier of a SBSP security block. The function determines whether the integrity result present in a security block created from a call to the ApplyIntegrity function and an integrity result computed from the target block as it exists at the security verifier are equivalent.

NOTE – Failure to verify integrity is not, itself, proof of target block corruption. Integrity results can fail to match if there is a change to the security block contents, the target block contents, the security verifier’s cipher suite implementation, or there is a difference in the security context.

The input parameters of this function include the block whose integrity is being verified, the security block holding integrity information, and any parameters defined local to the security verifier associated with the security context used to check integrity.

When called, this function provides all input parameters to the selected security context, captures the results generated by the security context, and determines if the integrity verification succeeded.

When the VerifyIntegrity Function has completed the processing, it returns resulting data to the caller in the return parameter, the VerifyIntegrity Return.

### Input Parameters

#### Target Block

The Target Block parameter contains the contents of the BP block for which integrity is being verified by this function.

#### Integrity Security Block

The Integrity Security Block parameter consists of the SBSP security block that contains the integrity result generated by the ApplyIntegrity function over the Target Block as it existed at the time of block creation. This block may also include the suite identifier, suite parameters, and additional results used by the ApplyIntegrity function

#### Local Security Context Parameters

The Local Security Context Parameters parameter shall consist of the information configured at the security verifier BPA for the security context specified in the security block identified in the Integrity Security Block parameter. These local security context parameters must be provided as inputs to the security context used to verify the integrity of the target block.

NOTE – The security block’s security context parameters and Local Security Context Parameters may provide conflicting values for the same parameter.

### Return Parameters

#### VerifyIntegrity Return

The VerifyIntegrity Return shall consist of a result code signifying either that the integrity of the target block was verified or was not verified. Implementations may choose to return additional result codes to document exactly why the integrity verification failed.

## Function at Security Acceptors

### Overview

Security acceptors are security-aware BPAs that serve as either the bundle destination or a bundle waypoint configured as the endpoint of one or more security blocks in a bundle. Upon reaching a security acceptor, a particular security service is no longer needed in a bundle and it must be removed using the functions AcceptIntegrity and AcceptConfidentiality, depending on whether the security service was integrity or confidentiality, respectively.

#### AcceptIntegrity Function

The AcceptIntegrity Function is defined for a security-aware BPA node accepting a security block constructed by the ApplyIntegrity function at the target block source.

Similar to the VerifyIntegrity function, AcceptIntegrity calculates a verification result code for the target block. Additionally, the AcceptIntegrity function must remove from the bundle the security block used to implement the removed integrity service.

The input parameters of this function include the bundle containing the block whose integrity is being checked and whose security block is being removed, the identifier of the target block, and any parameters defined local to the acceptor associated with the security context used to check integrity.

When the function is called, the security block generated by the ApplyIntegrity function at the target block source can be uniquely identified as the only integrity security block in the bundle whose target matches the input target block type. The contents of the target block, the security block, and local parameters are input to the selected security context and the result of the integrity verification is captured. Regardless of whether integrity is successfully verified, the security block is removed from the bundle.

When the AcceptIntegrity Function has completed processing, it returns resulting data to the caller in the return parameter, the AcceptIntegrity Return.

#### AcceptConfidentiality

The AcceptConfidentiality Function is defined for a security-aware BPA node configured as the acceptor of the ApplyConfidentiality function run at the target block source. This function removes confidentiality from a target block by replacing cipher text within the block with the original contents of the block. The AcceptConfidentiality function must remove from the bundle the security block representing the removed confidentiality service.

The input parameters of this function include the bundle containing the block whose confidentiality is being removed, the identifier of the block type, and any parameters defined local to the acceptor associated with the security context used to process confidentiality.

When the function is called, the security block generated by the ApplyConfidentiality function at the target block source can be uniquely identified as the only confidentiality security block in the bundle whose target matches the input target block type. The contents of the target block, the contents of the security block, and local parameters are input to the selected security context and the results of the security context are captured. Regardless of whether confidentiality is successfully removed, the security block is removed from the bundle.

When the AcceptConfidentiality Function has completed the processing, it returns resulting data to the caller in the return parameter, the AcceptConfidentiality Return.

### Input Parameters

#### Overview

The AcceptIntegrity and AcceptConfidentiality functions take the same types of parameters. This section describes the parameters in general and notes, per parameter, if there is any difference in interpreting the parameter for integrity or confidentiality.

#### Target Bundle

The Target Bundle parameter shall consist of the bundle containing both the security block being removed and the target block of that security block.

#### Target Block Type

The Target Block Type parameter uniquely identifies the block within the target bundle that is being used by the integrity or confidentiality function. This parameter also uniquely identifies the security block.

NOTE – When provided to the AcceptIntegrity function, the appropriate security block will be the only integrity block in the bundle whose target is the given target block type. When provided to the AcceptConfidentiality function, the appropriate security block will be the only confidentiality block in the bundle whose target is the given target block type.

#### Local Security Context Parameters

The Local Security Context Parameters parameter shall consist of those parameters configured at the security acceptor for the security context identified in the associated security block. These parameters must be provided as inputs to the suite of algorithms used to accept either integrity or confidentiality, depending on whether the function is the AcceptIntegrity or AcceptConfidentiality function, respectively.

NOTE – The security block’s security context parameters and Local Security Context Parameters may provide conflicting values for the same parameter.

### Return Parameters

#### AcceptIntegrity Return

The AcceptIntegrity Return shall consist of a result code and a modified bundle.

The result code will signify if the integrity of the target block was verified. Implementations may choose to return additional result codes to document exactly why the integrity verification failed.

The target bundle will be modified such that the security block created by the ApplyIntegrity function at the bundle source is removed from the bundle.

#### AcceptConfidentiality Return

The AcceptConfidentiality Return shall consist of a result code and a modified bundle.

The result code will signify if the confidentiality of the target block was successfully removed. Implementations may choose to return additional result codes to provide more detailed status.

The target bundle will be modified in two ways. First, the security block associated with the removed confidentiality service will be removed from the target bundle. Second, the target block of the confidentiality service will be modified based on whether confidentiality was successfully removed. If confidentiality removal was successful, the cipher text contents of the target block will be replaced by the plain text output from the security context. If confidentiality removal was not successful, the target block will be removed from the bundle.

#

# Protocol Specification

## Protocol Data Units

### General

Bundles that implement security services may include zero or more instances of the following security blocks subject to the constraints in this section.

1. Block Integrity Block (BIB);

NOTE – The BIB is used to ensure the integrity of its plain text target block from the security source, which calculates an integrity result, to the security acceptor, which verifies that integrity result and removes the BIB from the bundle.

1. Block Confidentiality Block (BCB).

NOTE – The BCB is used to ensure the confidentiality of its target block from the security source, which provides authenticated encipherment, to the security acceptor which provides verified decipherment and removes the BCB from the bundle.

A security operation must not be applied to a target block more than once in a bundle.

NOTE – For example, the two security operations: (integrity, payload) and (integrity, payload) are considered redundant and cannot appear together in a bundle. However, the two security operations (integrity, payload) and (integrity, extension\_block\_1) may both be present in the bundle. Also, the two security operations (integrity, extension\_block\_1) and (integrity, extension\_block\_2) are unique and may both appear in the same bundle.

The block type that is the target of a security operation must identify a singleton block; the combination of security block type and target block type uniquely identifies a security operation in the bundle.

The security source of every security block in a bundle must be the BPA that added the target block to the bundle.

Security blocks will be present only after the primary block and before the payload block in the bundle.

### Canonicalization

Each security block must use the Canonical Bundle Block Format as defined in section 4.5.2 of the Bundle Protocol Specification RFC (reference[1]).

NOTE – Each security block thus comprises the following elements, in the order which they are presented here:

* Block Type Code;
* Block Processing Control Flags;
* Block EID Reference List (optional);
* Block Data Length;
* Block-Type-Specific Data Fields.

### Generic Security Block Structure

The structures of the BIB and BCB block-type-specific data fields are identical and shall follow the Generic Security Block Structure (GSBS) as illustrated in figure 4‑1.

|  |
| --- |
| Target Block Type(SDNV) |
| Security Context ID (SDNV) | Security Block Flags (SDNV) |
|  \*Security Context Parameters Length (SDNV) |  \*Security Context Parameters Data (Compound) |
| Security Result Length (SDNV) | Security Result Data (Compound) |

Figure 4‑1 : The Generic Security Block Structure

NOTES

1. Although the diagram hints at a fixed-format layout, this is purely for the purpose of exposition. All fields are variable in length using SDNVs. In this figure, field names prefaced with ‘\*’ are optional, and their inclusion in the block is indicated by the Security Block Flags field.
2. The format of compound fields is defined below in 4.1.3.3.

The block fields are defined as follows.

Target Block Type (SDNV)—The singleton block type that uniquely identifies the target block of the associated security operation.

Security Context ID (SDNV)—The security context used to implement the security service represented by this block and applied to the target block.

Security Block Flags (SDNV)—These flags define which optional security block fields and other features are present in the block. The structure of the Security Block Flags field shall be as shown in figure 4‑2.

1. bits 7–1 are reserved for future use.
2. bit 0, param, indicates whether or not the Security Context Parameters Length and Security Context Parameters Data fields are present.



Figure 4‑2 : Security Block Flags

 (Optional) Security Context Parameters—If indicated by a value of 1 in the param bit (bit 0) of the Security Block Flags, the following two fields shall be present.

Security Context Parameters Length (SDNV)—The length of the next field, which is the security context parameters data field.

Security Context Parameters Data (Compound)—Parameters to be used with the security context in use, e.g., a key identifier or initialization vector (IV).

NOTE – Subsection 4.1.7 provides a list of security context parameters and their types. The particular set of parameters that is included in this field, and the encoding of those parameters, are defined as part of a security context specification.

NOTE – Examples of security context parameters include key identifiers, initialization vectors (IVs), and salt values.

Security Result Length (SDNV)—The length of the security result data field.

Security Result Data (Compound)—The results of the appropriate security context specific calculation.

NOTE – Examples of security results data include signatures, Message Authentication Codes [MACs], or integrity check values.

The format of a compound field is defined as follows.

Each item in any compound field shall be represented as a type-length-value tuple.

Type shall be a single byte indicating the item being specified as defined in a security context specification.

Length shall be the count of data content bytes to follow, represented as an SDNV-codified integer.

Value shall be the data content of the item, whose encoding is a function of the item type.

### Block Integrity Block

The block-type code value for Block Integrity Blocks must be as specified in Table *B‑1*.

The block processing control flags value can be set to the values required by local policy.

The target block for a BIB may only have as its target block the primary block, payload block, or a non-security extension block.

The security result must contain the result of applying the security context calculation (e.g., the MAC or signature) to the relevant parts of the target block.

If the security context does not protect the entire, original target block’s block-type-specific data, the security context parameters must specify which bytes of the target block are protected.

### Block Confidentiality Block

The block-type code value for Block Confidentiality Blocks must be as specified in Table *B‑1*.

The Block Processing Control flags value of a BCB can be set to whatever values are required by local policy, except that this block must have the ‘replicate in every fragment’ flag set if the target of the BCB is the payload block.

NOTE – The presence of a BCB in each fragment indicates to a receiving node that the payload portion of each fragment represents cipher text.

A BCB target block may only have as its target block the payload block or a non-security extension block.

The security context ID must be documented as a context which includes a confidentiality cipher that provides authenticated encryption with associated data (AEAD).

Additional information created by a security context (such as an authentication tag) can be placed either in a security result field or in the generated cipher text. The determination of where to place this information is a function of the security context used.

NOTES

1. A BCB must not alter the size of its target block’s block-type-specific data.
2. The generated cipher text will never be smaller than the provided plain text. This implies that there must always be a security result held in the BCB – either the authentication information from the security context or the overflow caused by including such information in the cipher text itself.

A BCB must have at least one security result.

NOTES

1. The security result compound field normally contains fields such as an encrypted bundle encryption key, authentication tag (integrity check value), or overflow cipher text.
2. The BCB modifies the contents of its target block. When confidentiality is applied, the target block’s block-type-specific data is encrypted ‘in-place’. Following encryption, the target block’s block-type-specific data contains cipher text, not plain text. Other target block fields (such as type, processing control flags, and length) remain unmodified.

If the security context does not protect the entire, original target block’s block-type-specific data, the BCB for that target block must specify, as part of the security context parameters, which bytes of the target block’s block-type-specific data are protected.

The BCB’s ‘Discard if block cannot be processed’ flag may be set independently from its target block’s ‘Discard if block cannot be processed’ flag.

NOTES

1. Whether or not the BCB’s ‘discard’ flag is set is an implementation/policy decision for the enciphering node.
2. Fragmentation, reassembly, and custody transfer are adversely affected by a change in size of the payload due to ambiguity about what byte range of the block is actually in any particular fragment. The requirements for ‘in-place’ encryption allow fragmentation, reassembly, and custody transfer to operate without knowledge of whether or not encryption has occurred.

### Block Interactions

A BIB and BCB must not share a target block. Therefore, the following rules must be followed.

A BIB must not be added for a target block that is already the target of a BCB.

A BCB must not be added for a target block that is already the target of a BIB.

NOTE – A BPA does not need to add both a BIB and a BCB for a target block because the BCB provides an integrity protection mechanism for its target block in the security results of the BCB.

### Security Context parameter and result Identification

Security contexts must be formally specified and registered. A security context specification must define its own context parameters and results. Each defined parameter and result is represented as the tuple of an identifier and a value. Identifiers are always represented as an SDNV-encoded integer. The encoding of values is as defined in the security context specification.

NOTE – Identifiers must be unique for a given security context but do not need to be globally unique.

NOTE – An example of a security context specification is provided in Annex E.

## Security Protocol Procedures

### Canonicalization algorithms

Security services require consistency and determinism in how information is presented to a security context at the security source and at a receiving node. For example, integrity services require that the same target information (e.g., the same bits in the same order) is provided to the security context when generating an original signature and when validating a signature. Canonicalization algorithms are used to construct a stable, end-to-end bit representation of a target block.

Security contexts may define their own canonicalization algorithms and require the use of those algorithms instead of the ones provided in this specification.

The three types of blocks that may undergo block canonicalization are the primary block, the payload block, and extension blocks.

NOTE – The canonical forms of these blocks are not transmitted.

#### Self-Delimiting Numerical Values (SDNVs)

Where a block contains SDNVs, the canonical form of any SDNV shall be an eight-byte fixed-width integer field in network byte order representing the ordinal number of the SDNV.

NOTE – The size of eight bytes is chosen because implementations may treat larger SDNV values as invalid, as noted in the BP specification (reference [1]).

#### Endpoint Identifiers (EIDs)

Where a block contains an endpoint ID (EID), the canonical form of the EID shall be represented based on whether the EID uses CBHE or not.

If the EID uses CBHE, the canonical form of the EID shall be the canonical form of the two SDNVs comprising the EID scheme name offset and scheme specific part offset, respectively.

If the EID does not use CBHE, the canonical form of the EID shall be the URI copied from the relevant parts(s) of the dictionary block and the scheme name offset and scheme specific part offset shall not appear in the canonicalization.

NOTE – The URI encoding will cause errors if any node rewrites the dictionary content (e.g., changing the DNS part of an HTTP URL from lower case to upper case). This could happen transparently when a bundle is synched to disk using one set of software and then read from disk and forwarded by a second set of software. Because there are no general rules for canonicalizing URIs (or IRIs), this problem may be an unavoidable source of integrity failures.

When canonicalizing a dictionary entry, the ‘null-terminators’ shall not be included in the length or the canonicalization of URIs.

NOTES

1 The URI encoding used for canonicalization does not preserve the null-termination convention from the dictionary field, nor is the scheme and Scheme-Specific Part (SSP) canonicalized separately. Instead, the byte array < scheme name > : < SSP > is used in the canonicalization.

2 A separator token: “,” shall be used when multiple EIDs are canonicalized together to determine where one EID ends and another begins.

#### The Primary Block

The canonical form of the primary block shall be as shown in figure 4‑3. The fields of the primary block are presented in the order in which they must appear.

|  |  |
| --- | --- |
| Version | Processing flags (incl. COS and SRR) |
| Canonical primary block length |
| Destination endpoint ID length |
| Destination endpoint ID |
| Source endpoint ID length |
| Source endpoint ID |
| Report-to endpoint ID length |
| Report-to endpoint ID |
| Creation Timestamp (2 x SDNV) |
| Lifetime |

Figure 4‑3 : The Canonical Form of the Primary Bundle Block

The values in the canonical form of the primary block are as follows.

The Version value shall be a single-byte value identifying the version of the bundle.

NOTE – The SBSP is only designed to be used with BP version 6.

The Processing flags value in the primary block shall be an SDNV that includes the Class Of Service (COS) and Status Report Request (SRR) fields. For purposes of canonicalization, the unpacked SDNV shall be ANDed with mask 0x0000 0000 0007 C1BE.

NOTE – The mask above sets all reserved bits and the ‘bundle is a fragment’ bit to zero.

The Canonical primary block length value shall be a four-byte value containing the length (in bytes) of this structure, in network byte order.

The Destination endpoint ID length shall be the length (represented as a four-byte value in network byte order) from the primary bundle block.

The destination endpoint ID value shall be a canonicalized EID.

The source endpoint ID length shall be the length (represented as a four-byte value in network byte order) from the primary bundle block.

The source endpoint ID value shall be a canonicalized EID.

The report-to endpoint ID length shall be the length (represented as a four-byte value in network byte order) from the primary bundle block.

The report-to endpoint ID value shall be a canonicalized EID.

The creation timestamp shall be copied from the primary block, composed of an SDNV representing a DTN time and an SDNV for the creation time’s sequence number.

The lifetime shall be copied from the primary block as an unpacked SDNV.

The fragment offset and total application data unit lengths shall be ignored during canonicalization.

#### The Payload Block

When canonicalizing the payload block, the block processing control flags value used for canonicalization shall be the unpacked SDNV value with reserved and mutable bits masked to zero. The unpacked value shall be ANDed with mask 0x0000 0000 0000 0077.

NOTE – The mask above sets to zero the reserved bits and the ‘last block’ bit.

A Payload block shall be canonicalized as-is, except where only a portion of the payload data is to be protected by a security operation.

If only a portion of the payload block data is protected by the security operation, only those bytes of the payload that are protected shall be included in the canonical form.

NOTE – In this case, additional security context parameters are required to specify which part of the payload is protected.

#### Extension Blocks

When canonicalizing an extension block, the block processing control flags value used for canonicalization shall be the unpacked SDNV value with reserved and mutable bits masked to zero. The unpacked value shall be ANDed with mask 0x0000 0000 0000 0057.

NOTE – The mask above sets to zero the reserved bits, the ‘last block’ flag, and the ‘Block was forwarded without being processed’ bit.

The ‘Block was forwarded without being processed’ flag shall be ignored during canonicalization.

NOTE – This flag is ignored because the bundle may pass through nodes that do not understand that extension block; this would cause the flag to be set.

The block length shall be canonicalized as its unpacked SDNV value. If the data to be canonicalized is less than the complete, original block data, this field shall contain the size of the data being canonicalized (the ‘effective block’) rather than the actual size of the block.

#### Discussion

Reserved flags are omitted from canonicalization because it cannot be determined if they will change in transit. The masks used to omit these flags must be revised if additional flags are defined and require protection by SBSP security services.

### Sending Procedures

#### A security source will apply a security operation to a bundle only when required by security policy.

#### A security operation may only be applied to a bundle if all of the following conditions are met.

##### The node in possession of the bundle must be identified by policy as the security source for the operation.

###### The security source must be the node which adds the target block of the security operation to the bundle.

###### The security source for the payload block and primary block must be the bundle source.

NOTES

1 The bundle source may apply either a BIB or BCB to the payload block.

2 The bundle source may apply a BIB to the primary block.

1. If a BIB targeting the primary block is added at the bundle source, the primary block must not be modified after it is transmitted by the bundle source.

##### The target block of the security operation must not be the target of an existing security block in the bundle.

NOTE – A BIB and BCB may not share a target block. It is recommended that a BCB is used if both integrity and confidentiality services are desired for the same target block. A BCB uses an AEAD cipher to provide confidentiality, generating an integrity protection mechanism as a product of encryption.

##### The target block of the security operation must be uniquely identified by its block type.

###### If the target block type of the security operation identifies multiple blocks in the bundle, the security operation cannot be applied.

NOTES

1 If a node adds a block to a bundle which results in the bundle containing multiple blocks of the same block type, that node may not act as the security source for that block.

2 In cases where all blocks of the same block type are removed from a bundle before a node adds a new instance of the block, that node may act as the security source for the new block as it can be uniquely identified by its block type.

#### It is recommended that bundle-in-bundle encapsulation is used in situations where the above conditions are not met if the security operation must still be applied to the bundle.

### Receiving Procedures

* + - 1. If the relationship between a security block and its target block is ambiguous, the security service will fail according to security policy.

NOTES

1 The scenario presented in 4.2.3.1 will not occur when every node that a bundle encounters during transmission is a security-aware BPA. However, a security block’s target block may be ambiguous if the bundle encounters a non-security-aware BPA during transmission which adds a duplicate block type to the bundle.

2 Network designers must consider the implications of allowing both security-aware and non-security-aware BPAs to be active in the network at one time.

#### Receiving BCB Blocks

* + - * 1. If the receiving node is the destination of the bundle, that node must decrypt any BCBs remaining in the bundle in accordance with the security context specification.

NOTE – In this case, the bundle destination is functioning as the default security acceptor for all BCBs in the bundle.

* + - * 1. A receiving node that is not the bundle destination may decrypt a BCB if security policy identifies it as the security acceptor for that BCB.
				2. When a BCB is decrypted, the recovered plain text must replace the cipher text in the target block’s block-type-specific data.
				3. If decryption of the BCB target block fails, the target shall be processed according to security policy.

######  If the relevant parts of an encrypted payload block cannot be decrypted (i.e., the decryption key cannot be deduced or decryption fails), then the bundle must be discarded and processed no further.

###### If an encrypted target block other than the payload block cannot be decrypted, the BCB and its associated target block must be discarded and processed no further.

###### If any BCB failure results in the removal of a block in a bundle or the bundle itself, then requested status reports may be generated to reflect bundle or block deletion.

* + - * 1. When the security operation for a BCB has been removed, the BCB must be removed from the bundle.

#### Receiving BIB Blocks

* + - * 1. If the receiving node is the destination of the bundle, the node must process any BIBs remaining in the bundle in accordance with the security context specification.

NOTE – In this case, the bundle destination is functioning as the default security acceptor for all BIBs in the bundle.

* + - * 1. A receiving node that is not the bundle destination may verify the integrity of the target block of a BIB and remove the BIB from the bundle if security policy identifies it as the security acceptor for that BIB.
				2. A receiving node that is not the bundle destination or security acceptor may verify the integrity of the target block of a BIB if security policy identifies it as a security verifier for that BIB.
				3. If integrity verification of the BIB target block fails, the target shall be processed according to security policy.

###### If a payload integrity check fails at a security verifier that is in possession of the correct cryptographic key, it is recommended that the bundle be processed in the same way as if the check had failed at the security acceptor.

###### If the target block is the payload or primary block and is removed by the security verifier or security acceptor as a function of security policy, the bundle must be discarded.

###### If the integrity verification of the BIB target block fails, a bundle status report indicating the failure may be generated.

* + - * 1. When the security operation for a BIB has been removed, the BIB must be removed from the bundle.

### Bundle Fragmentation

* + - 1. If a node fragments a bundle that has security services applied, the fragmentation rules described in the BP specification (reference [1]) must be followed.

NOTE – Only the payload block may be fragmented; security blocks, like all extension blocks, can never be fragmented.

* + - 1. Integrity and confidentiality operations are not to be applied to a bundle representing a fragment (i.e., a bundle whose ‘bundle is a fragment’ flag is set in the Bundle Processing Control Flags field).

NOTE – Specifically, a BCB or BIB must not be added to a bundle fragment, even if the target block of the security block is not the payload.

# Managed Parameters

## Overview

Managed parameters are those parameters provided by management rather than included in SBSP security blocks. Because BP bundles may be stored in a network for extended periods of time, parameters that identify the state of the network must be provided by management as part of the policy or technical configuration of security-aware BPAs in the network.

NOTE – BP networks cannot assume timely, reliable, end-to-end data exchange. Parameters associated with a specific instance of an integrity or confidentiality function input or output must be provided inline in an SBSP security block.

The detailed specification of some managed parameters must occur in the context of a specific security context.

NOTE – SBSP security blocks may specify different security contexts. These contexts may have different input parameters and some of these input parameters may be managed parameters.

## Requirements

### SBSP Managed Parameters

SBSP managed parameters are those parameters that are independent of the security contexts used to generate cryptographic materials. The managed parameters used for the SBSP shall be those listed in table 5.1.

NOTE – These parameters are defined in an abstract sense, and are not intended to imply any particular implementation of a management system.

Table 5‑1 SBSP Managed Parameters

|  |  |  |
| --- | --- | --- |
| Managed Parameter | Allowed Values | Defined in Reference |
| Local security-aware BPA parameters used to process SBSP security blocks. |
| Bundle Node Security Roles | Security SourceSecurity VerifierSecurity Acceptor | This document |
| Supported Suites | Integer, Agency-Specific |  |
| Expected security blocks | Block Type(s) |  |
| Known Suite Keys | Algorithm-specific, Agency-Specific |  |

### Security Context Managed Parameters

Security context managed parameters are those parameters associated with the security context of a SBSP security block, but managed and provided by the BPAs comprising the security source, security verifier (is appropriate) and security source of the security block as part of their local security context parameters.

The managed parameters of a specific security context definition must be provided in the specification of the security context.

1. Protocol Implementation Conformance
Statement (PICS) Proforma

(normative)
	1. INTRODUCTION
		1. OVERVIEW

This annex provides the Implementation Conformance Statement (ICS) Requirements List (RL) for an implementation of [Specification]. The ICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation claiming conformance must satisfy the mandatory requirements referenced in the RL.

* + 1. ABBREVIATIONS AND CONVENTIONS

The RL consists of information in tabular form. The status of features is indicated using the abbreviations and conventions described below.

Item Column

The item column contains sequential numbers for items in the table.

Feature Column

The feature column contains a brief descriptive name for a feature. It implicitly means ‘Is this feature supported by the implementation?’

Status Column

The status column uses the following notations:

* M mandatory;
* O optional;
* C conditional;
* X prohibited;
* I out of scope;
* N/A not applicable.

Support Column Symbols

The support column is to be used by the implementer to state whether a feature is supported by entering Y, N, or N/A, indicating:

Y Yes, supported by the implementation.

N No, not supported by the implementation.

N/A Not applicable.

The support column should also be used, when appropriate, to enter values supported for a given capability.

* + 1. INSTRUCTIONS FOR COMPLETING THE RL

An implementer shows the extent of compliance to the Recommended Standard by completing the RL; that is, the state of compliance with all mandatory requirements and the options supported are shown. The resulting completed RL is called an ICS. The implementer shall complete the RL by entering appropriate responses in the support or values supported column, using the notation described in A1.2. If a conditional requirement is inapplicable, N/A should be used. If a mandatory requirement is not satisfied, exception information must be supplied by entering a reference X*i*, where *i* is a unique identifier, to an accompanying rationale for the noncompliance.

* 1. PICS PROFORMA FOR CCSDS Streamlined Bundle Security Protocol
		1. GENERAL INFORMATION
			1. Identification of ICS

|  |  |
| --- | --- |
| Date of Statement (DD/MM/YYYY) |  |
| ICS serial number |  |
| System Conformance statement cross-reference |  |

* + - 1. Identification of Implementation Under Test

|  |  |
| --- | --- |
| Implementation Name |  |
| Implementation Version |  |
| Special Configuration |  |
| Other Information |  |

* + - 1. Identification of Supplier

|  |  |
| --- | --- |
| Supplier |  |
| Contact Point for Queries |  |
| Implementation Name(s) and Versions |  |
| Other information necessary for full identification, e.g., name(s) and version(s) for machines and/or operating systems;System Name(s) |  |

* + - 1. Identification of Specification

|  |
| --- |
| [CCSDS Document Number] |
| Have any exceptions been required?NOTE – A YES answer means that the implementation does not conform to the Recommended Standard. Non-supported mandatory capabilities are to be identified in the ICS, with an explanation of why the implementation is non-conforming. | Yes [  ]      No [  ] |

* + 1. REQUIREMENTS LIST

|  |
| --- |
| Classes |
| Item | Description | Reference | Status | Support |
| 1 | SBSP Support | 4.1.1.1 | M |  |

|  |
| --- |
| PDUs |
| Item | PDU | Ref. | Sender End-System | Receiver End-System | Relay |
|  |  |  | Status | Support | Status | Support | Status | Support |
| 2 | CBF | 4.1.2 | M |  | M |  |  |  |
| 3 | GSBS | 4.1.3 | M |  | M |  | C1 |  |
| 4 | BIB | 4.1.4 | O.1 |  | O.1 |  | O |  |
| 5 | BCB | 4.1.5 | O.1 |  | O.1 |  | N/A |  |

O.1: At least one of these options must be supported in order to claim compliance with this specification.

C1: If intermediate BIB verification is supported on a relay then the relay must be able to parse the Generic Security Block Structure.

|  |
| --- |
| Parameters of GSBS-PDU |
| Item | Parameter | Ref. | Status | Support | Values |
|  |  |  |  |  | Allowed | Supported |
| 6 | Number of security targets | 4.1.3.2.1 | M |  | >= 1 |  |
| 7 | Optional cipher suite parameters | 4.1.3.2.5,4.1.8 | M |  | 0-1,3-5,7-8 |  |

|  |
| --- |
| Security Processing |
| Item | Description | Reference | Status | Support |
| 8 | Canonicalization | 4.2.2 | M |  |
| 9 | BCB Decryption | 4.2.3.2.1, 4.2.3.2.2, 4.2.3.2.3, 4.2.3.2.4, 4.2.3.2.6, 4.2.3.2.7 | M |  |
| 10 | BCB Status Reports | 4.2.3.2.5 | O |  |
| 11 | BIB Reception | 4.2.3.3.1, 4.2.3.3.3, 4.2.3.3.4, 4.2.3.3.6, 4.2.3.3.8 | M |  |
| 12 | BIB Optional Checking | 4.2.3.3.2 | O |  |
| 13 | BIB Discard if no Primary Block Left | 4.2.3.3.5 | O |  |
| 14 | BIB Send Status Report on Failure | 4.2.3.3.7 | O |  |

|  |
| --- |
| Fragmentation and Reassembly |
| Item | Description | Reference | Status | Support |
| 15 | Fragmentation | 4.2.4.1, 4.2.4.2 | M |  |

|  |
| --- |
| Payload-Level Security |
| Item | Description | Reference | Status | Support |
| 16 | Use of CMS on payload | 4.2.5.1 | O |  |
| 16.1 | CMS format and processing | 4.2.5.3, 4.2.5.4, 4.2.5.5, 4.2.5.6, 4.2.5.7 | c:m |  |

|  |
| --- |
| Policy Considerations |
| Item | Description | Reference | Status | Options | Treatment |
| 17 | Dealing with violations | 5.2.1 | M | Discard bundle; discard affected block; favor one security option (specify method of determination). |  |
| 18 | Security operations to apply | 5.2.2 | M | Specify security operations to apply and criteria for determining them. |  |
| 19 | Conflict resolution | 5.2.3 | M | Discard the bundle; discard the target block; replace the security operation; other (specify) |  |

1. Security, SANA, and Patent Considerations

(Informative)

* 1. Security Considerations
		1. Scope

The SBSP addresses the security of blocks within a bundle and does not address the security of the underlying network. It may be necessary to apply security services at multiple layers within the protocol stack, to account for distributed processing and cross-support, to account for different classes of data or end users, or to account for protection of data during unprotected portions of the complete end-to-end transmission (e.g., across ground networks). The specification of security services at other layers is outside the scope of this document.

The SBSP does not addresses the fitness of externally defined cryptographic methods nor the security of their implementation. It is the responsibility of the SBSP implementer that appropriate algorithms and methods be chosen. It is the responsibility of the SBSP implementer to ensure that any cryptographic material, including shared secret or private keys, is protected against access within both memory and storage devices.

* + 1. security concerns
			1. General

Security concerns informing the design of the SBSP are addressed in more detail in reference [C5]. References [C1] and [C6] contain information regarding the choice of services and where they can be implemented. Reference [2] contains information regarding the choice of particular cryptographic algorithms.

This section discusses security threats that SBSP will face and describes how SBSP can mitigate these threats. The threat model described here is assumed to have a set of capabilities identical to those described by the Internet Threat Model (reference [C3]), but the SBSP threat model is scoped to illustrate threats specific to SBSP operating within DTN environments and therefore focuses on Man-In-The-Middle (MITM) attackers. Reference [C4] describes security threats against space missions.

* + - 1. Data Privacy

Malicious nodes may examine the contents of a bundle and attempt to recover protected data or cryptographic keying material from the blocks contained within. The SBSP BCB protects against this action by enciphering the contents of its target block thereby providing data privacy via a confidentiality service.

NOTE – Malicious nodes may continue to examine bundles offline in an attempt to recover encrypted data. The security contexts used by the BCB should be selected to provide suitable protection over the useful lifetime of the information being protected.

NOTE – To provide verifiable integrity checks, the security contexts used by the BCB should utilize encryption schemes that are “indistinguishable under adaptive chosen cipher text attack” (IND-CCA2) secure. Such schemes guard against signature substitution.

NOTE – Irrespective of whether the SBSP is used, traffic analysis will be possible.

* + - 1. Data Integrity

Malicious nodes may modify blocks within a bundle, to include replacing existing blocks, adding new blocks, and removing blocks. The SBSP can detect these activities using both the BIB and BCB blocks, depending on whether plain text or cipher text integrity is required.

NOTE – The integrity mechanisms used by the BIB and BCB should be strong against collision attacks and malicious nodes should be prevented from accessing the cryptographic material used by the security source. If these conditions can be met, malicious nodes will be unable to modify the target block without being detected.

The SBSP does not support an in-bundle mechanism to detect (or correct) cases where a malicious nodes removes a block from a bundle. If a target block is removed, then any security block associated with that target block will fail to validate. If a security block is removed from the bundle, some other policy must be in place at the security verifier or security acceptor to note that a security block was expected to exist in the bundle.

NOTE – The implementation of SBSP must be combined with a policy configuration at security-aware BPAs which describes the expected and required security operations that must be applied to, or expected to be present for, blocks in bundles processed by the BPA.

* + - 1. Authentication of Communicating Entities

The SBSP does not provide a service for authentication of communicating entities. This authentication can be accomplished by combining SBSP services with other components, such as encapsulation.

* + - 1. Control of Access to Resources

Resource access controls are not directly addressed by this specification.

* + - 1. Availability of Resources

No mechanisms are defined in this specification to verify or assist with the verification of availability of resources.

* + - 1. Auditing of Resource Usage

No mechanisms are defined in this specification to audit or assist with the auditing of resource usage by the protocol.

* + 1. Consequences of not Using the SBSp

If the SBSP is not used, bundle delivery must rely on security measures provided by the convergence layer adapter(s) and/or lower layers. For space applications these alternative security measures may be non-existent or shared across a large group of applications and application domains.

The consequence of not using SBSP is that bundle exchange has no consistent, end-to-end security. Lower layer security may differ link-by-link and application layer security may differ application-by-application.

* + 1. Sana Considerations
			1. General

This recommendations of this document request SANA to create or update the registries in this section.

* + - 1. Bundle Block Types Registry

The SBSP defines two new extension blocks whose block types must be registered with SANA. The new block types are defined as follows.

Table B‑1 SBSP Security Block Types

|  |  |  |
| --- | --- | --- |
| Value | Description | Reference |
| TBD | SBSP BIB Block Type | This document |
| TBD | SBSP BCB Block Type | This document |

* + - 1. Security Context Identifiers Registry

The purpose of this registry is to document the enumeration of security contexts defined for use with DTN security protocols. This registry is described as follows.

1. **Name**: Security Context Identifiers Registry
2. **Purpose**: Enumerate security contexts defined for use with DTN security protocols.
3. **Structure**: 3 Columns Table: Value; Description; Reference
4. **Data Types**:
	1. Value: 16-bit unsigned integer representing the enumeration of a security context.
	2. Description: A string of text describing the security context.
	3. Reference: The formal specification of the security context.
5. **Category**: WG/Local
6. **Review Authority**: SIS Area or designee
7. **Registration Procedure**: New values of this registry requires an official representative of a space agency member of the CCSDS, an approved, published specification from a recognized standards organization, and expert CESG review.

The initial registry should be filled with the following values.

Table B‑2 Security Context Identifier Values

|  |  |  |
| --- | --- | --- |
| Value | Description | Reference |
| 0 | Reserved for Testing | This document |
| 1 | SBSP Baseline Context | This document |

* + 1. Patent Considerations

There are no known patents covering the Streamlined Bundle Security Protocol as described in this document and its normative references.

1. Informative References

(Informative)

[C] *Security Architecture for Space Data Systems*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 351.0-M-1. Washington, D.C.: CCSDS, November 2012.

[C] *Security Guide for Mission Planners*. Issue 1. Report Concerning Space Data System Standards (Green Book) CCSDS 350.7-G-1. Washington, D.C.: CCSDS October 2011.

[C] E. Rescorla and B. Korver. *Guidelines for Writing RFC Text on Security Considerations*. RFC 3552. Reston, Virginia: ISOC, July 2003.

[] *Security Threats against Space Missions*. Issue 2. Report Concerning Space Data System Standards (Green Book) CCSDS 350.1-G-2. Washington, D.C.: CCSDS, December 2015.

[C] *Information Processing Systems—Open Systems Interconnection—Basic Reference Model—Part 2: Security Architecture*. International Standard, ISO 7498-2:1989. Geneva: ISO, 1989. 2015.

[C] *The Application of CCSDS Protocols to Secure Systems. Issue 2*. Report Concerning Space Data System Standards (Green Book) CCSDS 350.0-G-2. Washington, D.C.: CCSDS, January 2006

1. Abbreviations

(Informative)

Term Meaning

BCB Block confidentiality block

BER Basic Encoding Rules

BIB Block integrity block

BP Bundle Protocol

BPA Bundle Protocol agent

CBF Canonical Bundle Block Format

CMS cryptographic message syntax

COS class of service

DER Distinguished Encoding Rules

DTN Delay-Tolerant Networking

EID endpoint identifier

GSBS generic security block structure

HTTP Hypertext Transfer Protocol

ICV integrity check value

IETF Internet Engineering Task Force

IRI Internationalized Resource Identifier

IV initialization vector

MITM man in the middle

RFC Request for Comments

RL requirements list

SBSP Streamlined Bundle Security Protocol

SDNV self-delimiting numeric value

SRR status report request

SSP scheme-specific part

TLV type-length-value

URI Uniform Resource Identifier

URL Uniform Resource Locator

1. Baseline Mode

(Informative)
	1. OVERVIEW

This baseline implementation is specified to ensure interoperability and to allow representative interoperability testing. The baseline is intended to provide implementers with a standard way to implement the SBSP that can still be modified to accommodate individual missions.

* + 1. Algorithms
			1. Confidentiality

The baseline implementation to be used for interoperability testing of confidentiality is authenticated encryption over the target block’s block-type-specific data, using the Advanced Encryption Standard (AES) algorithm in Galois/Counter Mode (GCM). In addition:

1. the key is 256 bits in total length;
2. the key is symmetric;
3. the initialization vector is between 8-16 bytes, with 12 bytes being the recommended length;
4. the output MAC is 16 bytes in total length.
	* + 1. Integrity

The baseline implementation to be used for interoperability testing of integrity is a keyed hash over the target block’s block-type-specific data, using the Secure Hash Algorithm (SHA) combined with the keyed hash message authentication code (HMAC). In addition:

1. the key is 256 bits in total length;
2. the key is symmetric;
3. the truncation length is 256 bits;
4. the output MAC is 256 bits in total length.
	* 1. Key Information
			1. Confidentiality

The baseline implementation uses a symmetric key of 256 bits in length.

NOTES

1. It is assumed that the decrypting node knows the symmetric key used for encryption.
2. The configuration, storage, and exchange of keys are outside the scope of this recommendation.
	* + 1. Integrity

The baseline implementation uses a symmetric HMAC key of 256 bits in length.

NOTES

1. It is assumed that the node that verifies integrity knows the HMAC key used when generating the original keyed hash.
2. The configuration, storage, and exchange of HMAC keys are outside the scope of this recommendation.
	* 1. Canonicalization
			1. Confidentiality

The baseline implementation uses the canonicalization algorithms defined in section 4.1.2 with the following exceptions:

1. The plain text used during encryption must be calculated as a single, definite length byte array representing the block-type-specific data of the BCB target block;
2. The cipher text used during decryption must be calculated as a single, definite length byte array representing the block-type-specific data of the BCB target block;
3. All other fields of the BCB target block must not be considered as part of encryption or decryption.
	* + 1. Integrity

The baseline implementation uses the canonicalization algorithms defined in section 4.1.2 with the following exceptions:

1. The keyed hash must be calculated as a single, definite length byte array over the block-type-specific data of the BIB target block;
2. All other fields of the BIB target block must not be included in the calculation of the keyed hash.
	* 1. Processing
			1. Confidentiality
				1. Encryption at a Security Source

The baseline implementation performs encryption using the target block plain text, a unique initialization vector (IV), and a key as inputs to the AES-GCM-256 algorithm defined in E1.1.1.

NOTES

1. The IV is provided as a security context parameter, as seen in Table E‑1.
2. For every encryption performed with the same key, a unique IV must be used. The same key and IV combination must not be used more than once.
3. Key management is outside of the scope of this recommendation.

The cipher text calculated as a result of encrypting the target block must replace the target block’s plain text block-type-specific data. The cipher text must be the same length as the plain text to be replaced.

NOTE – The length field of the target block must not be modified as the generated cipher text will always be the same length as the plain text it is replacing.

The authentication tag generated by the security context must be added as a security result of the BCB.

* + - * 1. Decrypting at a Security Acceptor

The baseline implementation performs decryption using the target block cipher text, an IV, key, and authentication tag as inputs to the AES-GCM-256 algorithm defined in E1.1.1.

NOTES

1. The IV is provided as a security context parameter, as seen in Table E‑1.
2. The authentication tag is provided as a BCB security result, as seen in Table E‑1.

The plain text calculated as a result of decrypting the target block must replace the target block’s cipher text block-type-specific data. The plain text must be the same length as the cipher text to be replaced.

NOTE – The length field of the target block must not be modified as the generated plain text will always be the same length as the cipher text it is replacing.

* + - 1. Integrity
				1. Keyed Hash Generation at a Security Source

The baseline implementation generates a keyed hash using the target block’s block-type-specific data and an HMAC key as input to the HMAC-SHA-256 algorithm defined in E1.1.2.

NOTE – Key management is outside of the scope of this recommendation.

The generated hash must be stored as the Integrity Signature security result in the BIB.

* + - * 1. Keyed Hash Verification at a Security Verifier or Acceptor

The baseline implementation generates a keyed hash using the target block’s block-type-specific data and an HMAC key as inputs to the HMAC-SHA-256 algorithm defined in E1.1.2.

The generated hash shall be compared to the Integrity Signature security result of the BIB, seen in Table E‑1. If the hashes are identical, integrity verification is successful. Otherwise, integrity of the target block cannot be verified.

* + 1. Parameter and Result Definitions

The security context parameters and results for both integrity and confidentiality services shall be as defined in Table E‑1.

Table E‑1 Security Context Parameters and Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Name** | **Description** | **Field** |
| 0 | Reserved |  |  |
| 1 | Initialization Vector (IV) | A random value, typically eight to sixteen bytes. | Security Context Parameters |
| 2 | Reserved |  |  |
| 3 | Key Information | Material encoded or protected by the key management system and used to transport an ephemeral key protected by a long-term key. | Security Context Parameters |
| 4 | Reserved |  |  |
| 5 | Integrity Signature | Output from the HMAC calculation over the BIB target block at the security source. | Security Results |
| 6 | Reserved |  |  |
| 7 | Reserved |  |  |
| 8 | Authentication Tag | Output from the AES-GCM-256 cipher. This result is used at the security acceptor to verify that the protected data has not been modified. | Security Results |
| 9-255 | Reserved |  |  |