The Consultative Committee for Space Data Systems

Draft Recommendation for

Space Data System Standards

CCSDS BUNDLE PROTOCOL SECURITY SPECIFICATION

DRAFT RECOMMENDED STANDARD

CCSDS ???.?-R-1

RED BOOK
October 2021

AUTHORITY

	Issue:	Red Book, Issue 1	
	Date:	October 2021	
	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:

- 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.
- 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.
- 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
	CCSDS Bundle Protocol Security Specification, Draft Recommended Standard, Issue 1	October 2021	Current draft

CONTENTS

Figure

Table

INTRODUCTION

1.1 PURPOSE

This document defines a Recommended Standard for the CCSDS Bundle Protocol Security Protocol (BPSec), based on the Bundle Protocol Security Protocol of RFC 9172 (reference [11]). BPSec defines Bundle Protocol version 7 (RFC9171) (reference [1]) extension blocks with associated procedures that may be used with BPv7 bundles. These extension blocks provide a structured method for applying data integrity and/or data confidentiality to blocks within a bundle.

1.2 SCOPE

This Recommended Standard defines BPSec 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 BPSec. Reference [2] provides a listing of algorithms recommended by CCSDS; any organization should conduct a risk assessment before choosing to substitute other algorithms. Annex D (non-normative) defines baseline implementations suitable for a large range of space missions.

The protocol specified here applies only to the Bundle Protocol version 7 and does not interact with other CCSDS protocols. BPSec applies to the Session and Presentation layers of the OSI model as it relates to the Bundle Protocol, as seen in Figure 3-1 of [9].

1.3 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.

BPv7 requires that inter-bundle security services (as opposed to the security services provided by overlying application protocols or underlying convergence-layer protocols) be provided in accordance with the BPSec Recommended Standard. BPv7 also requires that any BPv7 Agent (BPA) which sources, cryptographically verifies, and/or accepts a bundle must implement support for the BPSec Recommended Standard. The use of this Recommended Standard for any particular BPv7 session is optional.

1.4 RATIONALE

The goals of this Recommended Standard are to:

a) provide a standard method of applying block-specific security for bundle transport, independent of the underlying cryptographic algorithms employed by any particular space mission; and

b) facilitate the development of common commercial implementations to improve interoperability across agencies.

1.5 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 BPSec.

Section 3 (normative) provides a profile of BPSec and specifies the constraints associated with BPSec services.

Section 4 (normative) defines the services provided by the protocol entity.

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

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

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

Annex D (normative) defines baseline implementations suitable for a large range of space missions.

Annex E (normative) defines BPSec managed parameters.

Annex F (informative) provides policy considerations for BPSec.

Annex G (informative) is a service specification for BPSec policy.

1.6 DEFINITIONS

This section provides terms and definitions necessary for understanding this Recommended Standard. In cases where necessary terms are defined in other documents, they are repeated in this section for clarity and convenience.

Additional terms and definitions related to this Recommended Standard can be found in references [1, 4, 5, 8, 11].

1.6.1 DEFINITIONS

The Space Assigned Numbers Authority (SANA) registry of CCSS Terms [REF] and the Internet Security Glossary Version 2 (reference [8]) define common security terms as they relate to the deployment of security systems.

In some cases the input data to an encryption function may itself be encrypted. This is referred to as superencryption.

BP may be deployed in scenarios that prohibit establishing relationships between two or more entities prior to exchanging information. BPSec defines a new term, security context, to refer to the practice of annotating a bundle with contextual information that would otherwise be used to establish a security association.

Security associations may still be used with BPSec under the auspices of a BPSec security context which allows them.

1.6.2 DEFINITIONS FROM BPV7

This Recommended Standard also makes use of a number of terms defined in reference [1]. These terms, and their definitions, are repeated in this section for convenience.

Bundle: A bundle is a protocol data unit of BP, so named because negotiation of the parameters of a data exchange may be impractical in a delay-tolerant network: it is often better practice to "bundle" with a unit of application data all metadata that might be needed in order to make the data immediately usable when delivered to the application. Each bundle comprises a sequence of two or more "blocks" of protocol data, which serve various purposes.

Block: A bundle protocol block is one of the protocol data structures that together constitute a well-formed bundle.

Bundle node: A bundle node (or, in the context of this document, simply a "node") is any entity that can send and/or receive bundles.

Bundle protocol agent: The bundle protocol agent (BPA) of a node is the node component that offers the BP services and executes the procedures of the bundle protocol.

Bundle endpoint: A bundle endpoint (or simply "endpoint") is a set of zero or more bundle nodes that all identify themselves for BP purposes by some common identifier, called a "bundle endpoint ID"

Endpoint Identifier (EID): A Uniform Resource Identifier (URI; reference [7]) used to identify a bundle endpoint.

> NOTE – This definition is taken from the text of BPv7, but not provided in the terminology portion of [1].

Node ID: An EID that identifies the administrative endpoint of a node. The EID of a node's administrative endpoint uniquely identifies that node.

> NOTE – This definition is taken from the text of BPv7, but not provided in the terminology portion of [1].

1.6.3 DEFINITIONS FROM BPSEC

This Recommended Standard makes use of a number of terms defined in reference [11]. Terms from the BPSec standard that are relevant to this Recommended Standard have been repeated in this section for convenience.

Bundle Destination: The node which receives a bundle and delivers the payload of the bundle to an application. Also, the Node ID of the Bundle Protocol Agent (BPA) receiving the bundle. The bundle destination acts as the security acceptor for every security target in every security block in every bundle it receives.

Bundle Source: The node which originates a bundle. Also, the Node ID of the BPA originating the bundle.

Security acceptor: A bundle node that processes and dispositions one or more security blocks in a bundle. Security acceptors act as the endpoint of a security service represented in a security block. They remove the security blocks they act upon as part of processing and disposition. Also, the Node ID of that node.

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

Security block: A BPSec extension block in a bundle.

Security context: The set of assumptions, algorithms, configurations, and policies used to implement security services.

> NOTE – A security context identifies cipher suites, annotative data, and other parameters needed for a security verifier or security acceptor to process the security results in a security block. Security contexts differ from security associations; a security context can be defined which uses security associations, in which case parameters such as a security association identifier would be defined and used with a security-association aware security context.

Security operation: The application of a given security service to a security target, notated as OP(security service, security target). For example, OP(bcb-confidentiality, payload). Every security operation in a bundle must be unique, meaning that a given security service can only be applied to a security target once in a bundle. A security operation is implemented by a security block.

Security service: A process that gives some protection to a security target.

> NOTE – BPSec defines two security services. One service is defined for plain text integrity (bib-integrity). The other services is defined for authenticated plain text confidentiality with additional authenticated data (bcb-confidentiality).

Security source: A bundle node that adds a security block to a bundle. Also, the Node ID of that node.

Security target: The block within a bundle that receives a security service as part of a security operation.

NOTE – To avoid confusion with the term Security Target which can mean a set of requirements and specifications used to evaluate information technology products or systems, the term "Target Block" is used in this Recommended Specification.

Security verifier: A bundle node that verifies the correctness of one or more security blocks in a bundle. Unlike security acceptors, security verifiers do not act as the endpoint of a security service and do not remove verified security blocks. Also, the Node ID of that node.

1.6.4 DEFINITIONS FROM THIS STANDARD

In some cases, this Recommended Standard provides a specific definition for a general security-related term. These definitions will be considered normative in the context of this standard, but may have differing definitions in other contexts.

Cipher suite: A set of one or more cryptographic 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.

Target block: A BPSec security target; a block that receives a security service as part of a security operation.

1.7 NOMENCLATURE

1.7.1 NORMATIVE TEXT

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

a) the words 'shall' and 'must' imply a binding and verifiable specification;

b) the word 'should' implies an optional, but desirable, specification;

c) the word 'may' implies an optional specification;

d) 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.

1.7.2 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.

1.8 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.

[1] Burleigh, S., Fall, K., Birrane, E., "Bundle Protocol Version 7" RFC9171. Reston, Virginia, ISOC, January 2022.

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

[3] *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.

[4] *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.

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

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

[7] 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.

[9] *The Application of Security to CCSDS Protocols*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 350.0-G-3. Washington, D.C.: CCSDS, March 2019.

[10] *CCSDS Symmetric Key Management*. Issue 1. Draft Recommendation for Space Data System Practices (Red Book), CCSDS 354.0-R-1. Washington, D.C.: CCSDS, June 2018.

[11] Birrane, E., McKeever, K., "Bundle Protocol Security Specification" RFCXXX

2 OVERVIEW

2.1 SECURITY NEEDS FOR BUNDLE PROTOCOL VERSION 7

The Bundle Protocol (BP) provides end-to-end communications across many networking environments, to include delay/disruption tolerant networks (DTNs). The BPv7 specification refers to a DTN as "a networking architecture providing communications in and/or through highly stressed environments" (reference [1]). In this context, the term "highly stressed environment" can refer to multiple challenging conditions including intermittent connectivity, large and/or variable delays, asymmetric data rates, and high bit error rates. Whether deployed in a DTN or some other networking environment, "BP may be viewed as sitting at the application layer of some number of constituent networks, forming a store-carry-forward overlay network" (reference [1]).

BP is presumed to be deployed in circumstances where the networking environment is not trusted, which requires consideration of usual security challenges related to confidentiality and integrity.

Networks may be untrusted for a variety of reasons. Different nodes in a network may cross multiple administrative boundaries (such as on the Internet) such that not every administrative entity is given the same level of trust. Within a given administrative boundary such as an enterprise intranet, nodes may have different physical or logical access controls and policies which may affect trust. Similarly, individual nodes may incorporate components (convergence layers, virtual machines, open source libraries) with differing levels of trust.

The stressed nature of certain networks where BP may be deployed may impose constraints that break or otherwise impede the use of usual transport security mechanisms. BPSec specifies unique security features inherent in securing a store-and-forward based transport protocol, to include 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" (reference [11]).

Because usual transport security mechanisms that require multiple handshakes may not be feasible in all environments where BP operates, a new end-to-end security service is needed that can secure bundles in all operational environments.

2.2 CONCEPT OF BUNDLE PROTOCOL SECURITY

BPSec is a data processing method by which space missions can apply security services to the individual blocks that comprise a BPv7 bundle. BPSec data units are codified as extension blocks within the BP and only exist in the context of a bundle.

2.2.1 BPSec Security Services

BPSec provides two security services for blocks within a BPv7 bundle:

1. **bib-integrity**. This service provides an integrity mechanism over the plain text block-type-specific data of a target block. This service also provides the option to include additional data beyond the plain text of the block-type-specific data of the target block. Integrity mechanisms (absent confidentiality mechanisms) detecting changes resulting from processing errors, environmental conditions, or intentional manipulation. When the integrity mechanism incorporates signatures, this service can also provide authentication for the relevant data.

2. **bcb-confidentiality**. This service provides authenticated confidentiality over the plain text block-type-specific data of a target block. This service also provides the option to include additional authenticated data beyond the plain text of the block-type-specific data of the target block.

BPSec does not provide a mechanism for hop-by-hop authentication for two reasons:

1. The term hop-by-hop is ambiguous in a BP overlay; BPA that are adjacent at the BP layer may not be adjacent in physical connectivity. This condition is difficult or impossible to detect and may give a false impression of the transport of data between BPAs. This is particularly true in a DTN where the topology of the network may change over time while bundle are stored pending future transmission.

2. Authenticated integrity and/or confidentiality is already defined between endpoints in a system. These endpoints can be topologically adjacent BPAs without implying that the BPAs are (or remain) adjacent as the topology of the network evolves.

2.2.2 BPSec Security operations and Blocks

BPSec security services can be applied to individual blocks within a bundle, allowing for a very fine-grained approach to security within a bundle.

Blocks in a bundle may carry different types of information. Payload blocks carry application data, whereas extension blocks may carry network information, annotative information related to the payload, or special processing instructions for the bundle itself. This combination of network-focused, bundle-focused, and application-data-focused information often requires different types of security services with different shared secrets (keys), different policies, and different accesses.

BPSec defines a security operation as the application of a security service to a particular target block within a bundle. This operation is notated as *OP(service, target)*. For example, applying the security service of bib-integrity to a bundle's primary block could be notated as:

OP(bib-integrity, primary block)

BPSec requires that all security operations in a bundle be unique, which means that the same security service cannot be applies to the same target block multiple times. This could create ambiguity in the order of verifying integrity and how to handle cases where some integrity mechanisms succeed and others fail their verifications.

BPSec security services (such as *bib-integrity*) use security contexts to define cipher suites, parameters, and results. It is possible to define a security context which would calculate multiple integrity results over a single target block. In this case, there is still a single operation, *OP(bib-integrity, target block)*, with that operation carrying multiple security results from the generating security context. This avoids the ambiguity of having multiple security services, because this single security service generates all integrity results at the same time, processes all integrity results at the same time, and provides a single mechanism to disposition multiple integrity results that pass/fail verifications.

Security operations are instantiated in a bundle by their inclusion in a security block. Every security block contains at least one, and possible many, security operations.

When multiple security operations share certain common attributes (e.g., the same security source, security service, and parameters) they may be aggregated into a single security block. In this case, there is a one-to-many relationship between a security block and the security operations it carries.

The relationship between security blocks and target blocks is illustrated in Figure 2-1. This figure illustrates a single bundle containing 6 blocks: the required primary and payload block, and four (4) extension blocks. Security block (1) providing a security service for target block (1) and security block (2) providing a security service for target block (2).

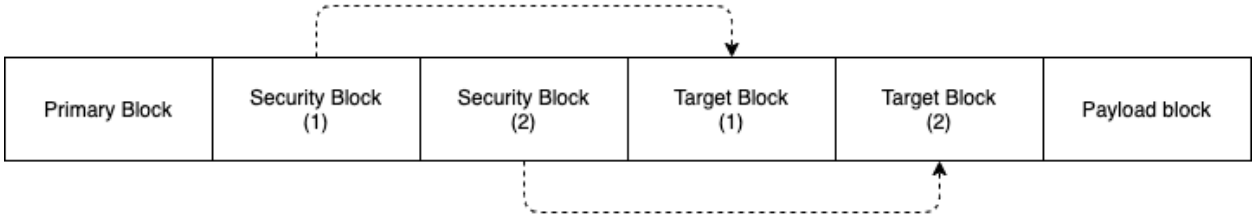

Figure 2-1: BPSec security blocks target other bundle blocks.

2.2.3 BPSec Roles and Responsibilities

BPSec defines three roles associated with the processing of security operations within security blocks: the security source, security verifier, and security acceptor.

NOTE – These roles are defined *per security operation*. This means that different security operations in the same bundle might have different security sources, verifiers, and acceptors.

The security source of a security operation refers to the BPA that added the security block representing that security operation to the bundle. This source is identified as the Node ID of that BPA. Security sources evaluate security operations prior to their inclusion in a security block to ensure that adding that operation would not violate any constraints imposed by BPSec.

NOTE – When a security block contains multiple security operations, all such operations share the same security source.

The security verifier of a security operation refers to one or more BPAs which verify a security operation as a bundle transits through the BPA. If the verification succeeds, the security service remains in the bundle. If the verification fails, the security block, and bundle, might be processed differently in accordance with local BPA security policy.

A given BPA might be identified as a security verifier for some, but not all, of the security operations in a security block. For example, if a security block contains two security operations: *OP(bib-integrity, primary block)* and *OP(bib-integrity, payload block)*, BPAs might be configured as security verifiers only for integrity results over the primary block and not the payload block.

A common use of security verifiers is to verify the signed integrity of blocks in a bundle prior to the delivery of that bundle at its destination. Detecting corrupted data quickly (and removing corrupted bundles early) helps reduce congestion and resource utilization within the network.

The security acceptor of a security operation refers to the single BPA that both verifies and removes a security operation from a bundle. When the last security operation is removed from a security block, the security block is removed from the bundle.

A security acceptor does not have to be the bundle destination. Security operations may be terminated prior to a bundle reaching a destination, particularly when some security results are processed at administrative boundaries in a network.

The bundle destination is considered the default security acceptor for any security operations remaining in a bundle when it reaches its destination.

Figure 2-2 illustrates the operational concepts associated with BPSec. In this figure, bundles are created at BN1 and BN2, and received by BN2 and BN3. The BPAs at these nodes take on the roles of security source, verifier, and acceptor as they receive different bundles. These roles are determined by local security policy. A BPA may take on multiple roles, as evidenced by BN2, 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 during transmission, as bundle 2 in Figure 2-2 does at BN2.

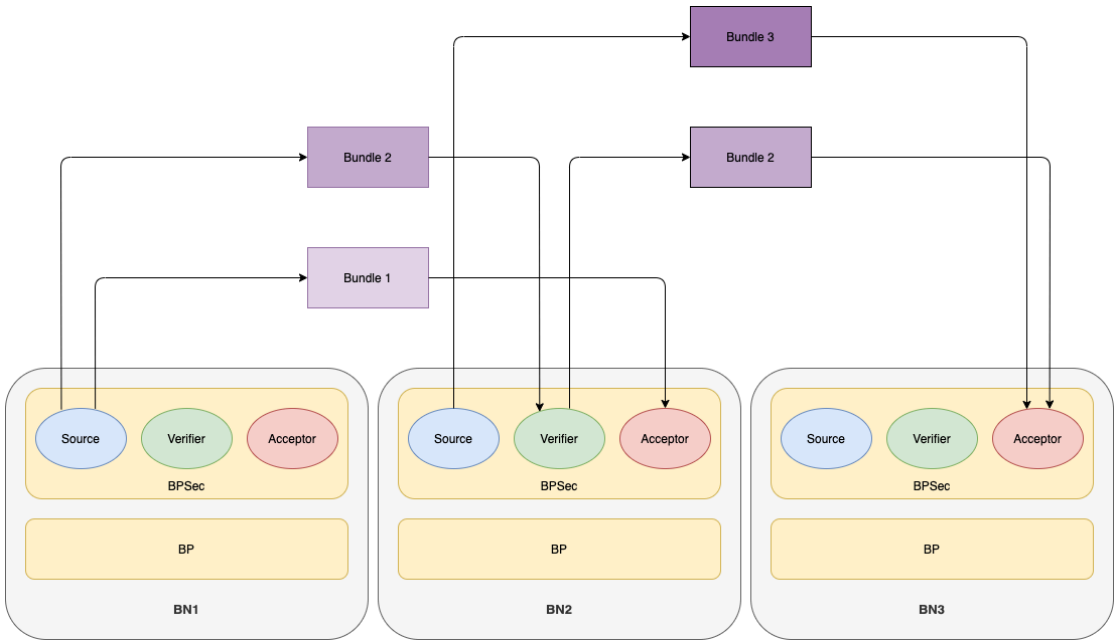

Figure 2-2: BPAs have multiple security roles [C10]

Figure 2-3 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 a BPA identified as a security source by policy. 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 BPSec block.

Steps 3 and 4 occur at a BPA identified as a security verifier by policy. 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 a BPA identified as the security acceptor for the security block by policy. 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.

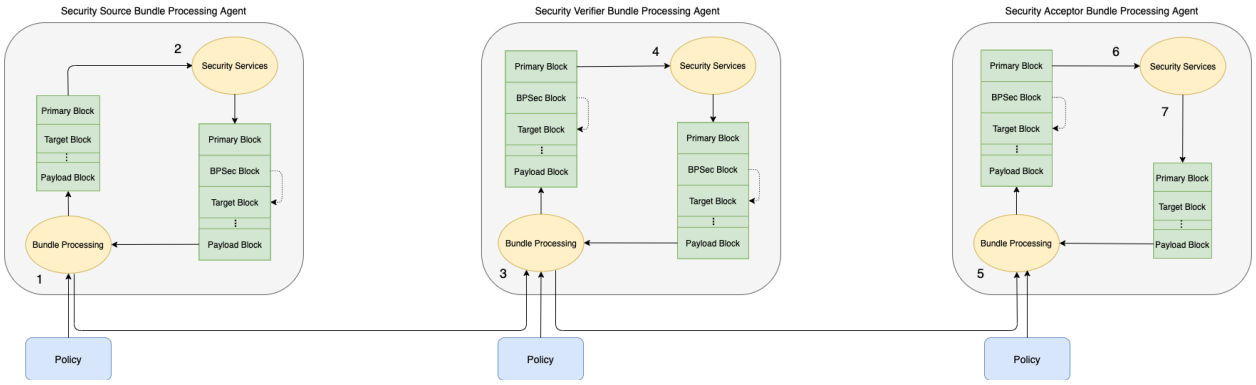

Figure 2-3: BPAs process security operations within security blocks. [C10]

2.3 NOTABLE FEATURES OF BPSEC

This section describes those features of BPSec that distinguish it from other transport layer security mechanisms.

2.3.1 GENERAL

BPSec provides security services to assure the confidentiality or integrity of the contents of blocks in a bundle.

Specifically, BPSec provides the following capabilities.

a) Security blocks may be added to a bundle by the bundle source BPA and/or downstream BPAs;

b) Security blocks may be modified or removed from a bundle by BPAs acting as security verifiers and security acceptors for security operations within those blocks;

c) A security block may represent either a collection of data integrity or a collection of confidentiality security operations;

d) A security operation may be added to a security block in a bundle if it represents a unique security operation;

e) Security results associated with a security operation are generated using a security context that specifies the cipher suite, user parameters for the cipher suite, and any special processing rules related to the use of the cipher suite; and

f) Multiple security blocks of the same block type may exist in a bundle.

Future security specifications might define additional security extension blocks and processing associated with those blocks to provide more complex security services for more complex operating scenarios. BPSec specifies the manner in which new security blocks can be made interoperable with existing BPSec security blocks.

It is expected that multiple security contexts will be defined for use with BPSec.

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 Recommended Specification is an implementation document for which a companion design document (Green Book) is planned.

2.3.2 BLOCK-LEVEL GRANULARITY

The target of any security operation represented in a security block is another block in the bundle. The ability to "target" blocks within a bundle enables BPSec to secure different information with a bundle differently.

Securing target blocks separately is a unique requirement derived from the unique BPv7 feature of allowing different types of information to be added in different types of extension blocks.

Some blocks carry information about the bundle itself, while other blocks carry application data or annotations related to the application data, and other blocks might carry information relevant to the state of the network itself. Applying a single integrity or confidentiality service to all of this information would fail to recognize that blocks in a bundle represent different types of information with different security needs.

2.3.3 MULTIPLE SECURITY SOURCES

BPSec allows a bundle to contain multiple security blocks which may have different security sources.

This is a unique ability because blocks might be added to a bundle after the creation of the bundle and these blocks may need security applied to them after the initial creation and transmission of the bundle from the bundle source.

BPv7 allows extension blocks to be added to a bundle at any time during its existence in the DTN. A downstream node may add a security block to a bundle, in which case that node is the security source for the added security block.

2.3.4 MIXED SECURITY POLICY

BPSec allows different security operations to be governed by different security policies. This allows security policy enforced by BPAs in the DTN to differ. The determination of security source, security verifier and security acceptor roles are on a security operation basis as a function of local policy and information resident in the received bundle.

NOTE – Policy in this context refers to the determination of the BPA role as it relates to a given security operation and the actions to be taken as a function of attempting to process the security operation in the context of that role.

This is a unique ability as it allows elements of policy determination to be driven by security parameters and annotations carried in security blocks within a bundle.

NOTE – This approach is consistent with other uses of blocks with BPv7 where end-to-end information that cannot be synchronized at endpoints is, instead, carried in extension blocks in a bundle.

The topology of a DTN might evolve over the lifetime of a bundle such that different nodes along a bundle path have different security capabilities and roles than existed at the time of bundle creation. Therefore, the determination of which BPAs have which security roles is made as part of the configuration of the BPA itself, and not pre-determined or asserted at the time of bundle creation.

2.3.5 USER-SELECTABLE SECURITY CONTEXTS

BPSec defines security contexts as a way of differentiating uses of cipher suites based on operational scenarios in a DTN. Importantly, different security contexts can be used for different bundles, and for different security operations within a single bundle.

The application of a given security context to a security operation is defined by local node policy on the BPA serving as the security source of the security operation.

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.

BPSec security blocks identify the security context used in the handling of cryptographic materials associated with the security operations represented by these blocks.

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.

Different security contexts may use different cipher suites or the same cipher suite operating in different modes or with different configurations.

2.3.6 DETERMINISTIC PROCESSING

BPSec ensures that the creation, verification, and acceptance of security services within a bundle always produces deterministic results. To accomplish this, BPSec imposes a strict ordering for certain operations and restricts certain features or feature combinations could endanger this ordering.

3 CCSDS PROFILE OF BPSEC RFC 9172

3.1 GENERAL

This document adopts the Bundle Protocol Security Protocol as specified in Internet RFC TBD (Reference [1]), with the constraints and exceptions specific in section 3 of this document.

3.2 USE OF APPROVED SECURITY CONTEXTS

Security Context Identifiers (SCIs) used shall be assigned by IANA from the IANA Security Context Identifier Registry and listed as approved security contexts by the CCSDS.

NOTES

1. IANA assigns BPSec Security Context Identifiers using the registry located at: (https://www.iana.org/assignments/bundle/bundle.xhtml#bpsec-security-context). This registry should include a range of context identifiers to be assigned and managed by SANA.

2. Wherever possible, missions should use security contexts that are either approved by the CCSDS or otherwise allocated from the SANA-assigned portion of the BPSec Security Context Identifiers registry.

3. Security contexts are defined in normative specifications, to include the default context provided in Annex E of this document. The appropriateness of a security context for any particular mission use is at the sole discretion of the mission as a function of its own security and threat analysis.

4. Missions may characterize their security requirements as low, medium, or high in accordance with Reference [9]. This characterization refers to the scope of security and associated policy. There is not a direct correlation between a specific security context and a specific mission security characterization.

3.3 USE OF CCSDS-APPROVED KEY MANAGEMENT

Security contexts approved by the CCSDS shall support CCSDS-approved key management mechanisms.

NOTES

1. Key management mechanisms for SANA-Registered security contexts shall conform to Reference [2].

2. Symmetric key management mechanisms for SANA-Registered security contexts shall conform to Reference [3].

3. Key management mechanisms may be specified separate from the definition of a security context in cases where multiple security contexts might use the same key management mechanisms (such as exchanging integrity keys and encryption keys) or in cases where missions may choose to employ different mechanisms based on considerations outside of the scope of the security context itself.

3.4 APPLICATION OF CCSDS POLICY CONSIDERATIONS

The specification and management of security policy on BPAs shall be in conformance with recommended practices outlined in Annex F of this document.

3.5 SANA REGISTRY CONSIDERATIONS

5.2.5 SECURITY CONTEXT IDENTIFIERS

5.2.5.3 SANA has established the registry

http://sanaregistry.org/TBD

to manage BPSec security context identifiers. The registry shall be used to catalog agency-managed BPSec security context definitions.

NOTES

1. The purpose of the registry is to ensure the uniqueness of BPSec security contexts as used in agency missions, separate from the specification of contexts by other organizations whose presumed use is over the terrestrial Internet.

2. It is envisioned that the SANA registry of identifiers be an allocated segment of a larger IANA registry of identifiers. This prevents overlap in cases where missions wish to use terrestrial security contexts for their ground segments or otherwise apply these contexts for on-board processing in appropriate circumstances.

3.5.1.2 Value Range for SANA Security Context Identifiers

The value range for BPSec Security Context Identifiers shall be as assigned by IANA.

3.5.1.3 SANA BPSec Security Context Identifier Registration Policy

The registration policy for the registry shall be: no engineering review required; request must come from an identified CCSDS representative of a member, observer, or affiliate organization.

NOTE - For missions utilizing BPSec, efforts should be made to reduce the number of registered security contexts. Where similar cipher suites are used, a single security context should be defined with additional behavior expressed through the addition of optional security context parameters or by BPSec policy at mission nodes.

4 BPSEC SERVICE DEFINITION

4.1 OVERVIEW

This section provides the service definition for BPSec.

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

(a) ApplyBIB
(b) ApplyBCB
(c) VerifyBIB
(d) VerifyBCB
(e) AcceptBIB
(f) AcceptBCB

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 BPSec 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).

4.2 FUNCTIONS AT SECURITY SOURCES

4.2.1 OVERVIEW

A security block is added to a bundle by the BPA serving as the security source for that block. Each security block may contain one or more related security operations, with each security operation applied to a different target block in the bundle. The BPSec defines two security blocks: BIB and BCB, and these blocks are added to a bundle using two functions: ApplyBIB and ApplyBCB.

4.2.2 ApplyBIB.request

4.2.2.1 Function

The ApplyBIB.request primitive shall be used to apply a BIB integrity operation to one or more target blocks in a bundle.

All blocks that are targets of the BIB operation must be present in the target bundle before invoking ApplyBIB.request.

4.2.2.2 Semantics

ApplyBIB.request shall provide parameters as follows:

ApplyBIB.request(target bundle, target block list, security context, security context parameters)

4.2.2.3 When Generated

This function is run on a BPA when local security policy determines that the BPA is the security source of a bib-integrity service for one or more target blocks in the bundle.

4.2.2.4 Effect on Receipt

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 target bundle in the form of a new security block.

The modified target bundle (with the new BIB block) is returned to the caller in the ApplyBIB.indication.

4.2.2.5 Discussion - Additional Comments

None

4.2.2.6 APPLYBCB.REQUEST

4.2.2.7 Function

The ApplyBCB.request primitive shall be used to apply a BCB integrity operation to one or more target blocks in a bundle.

All blocks that are targets of the BCB operation must be present in the target bundle before invoking ApplyBIB.request.

4.2.2.8 Semantics

ApplyBCB.request shall provide parameters as follows:

ApplyBCB.request(target bundle, target block list, security context, security context parameters)

4.2.2.9 When Generated

This function is run on a BPA when local security policy determines that the BPA is the security source of a bcb-confidentiality service for one or more target blocks in the bundle..

4.2.2.10 Effect on Receipt

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

The modified bundle (with the new BCB block and any other changes to the target bundles) is returned to the caller in the ApplyBCB.indication.

4.2.2.11 Discussion - Additional Comments

None

4.2.3 APPLYBIB.RETURN

4.2.3.1 Function

The ApplyBIB.return primitive shall be used to provide the BPA the contents of the bundle after the requested BIB security operation has been applied.

4.2.3.2 Semantics

ApplyBIB.return shall provide parameters as follows:

ApplyBIB.return(modified bundle, new BIB block id)

4.2.3.3 When Generated

ApplyBIB.return is generated by the BPSec service once the requested BIB security operation has been applied to the target bundle and the contents of the modified bundle have been computed.

4.2.3.4 Effect on Receipt

The effect on receipt of ApplyBIB.return by the BPA is undefined.

4.2.3.5 Discussion - Additional Comments

None

4.2.4 ApplyBCB Return

4.2.4.1 Function

The ApplyBCB.return primitive shall be used to provide the BPA the contents of the bundle after the requested BCB security operation has been applied.

4.2.4.2 Semantics

ApplyBCB.return shall provide parameters as follows:

ApplyBCB.return(modified bundle, new BCB block id)

4.2.4.3 When Generated

ApplyBCB.return is generated by the BPSec service once the requested BCB security operation has been applied to the target bundle and the contents of the modified bundle have been computed.

4.2.4.4 Effect on Receipt

The effect on receipt of ApplyBCB.return by the BP application is undefined.

4.2.4.5 Discussion - Additional Comments

None

4.3 FUNCTIONS AT SECURITY VERIFIERS

4.3.1 OVERVIEW

Security operations within a security block may be verified by BPAs whose local security policy identifies them as a security verifier for one or more operations within the bundle. These services are applied using two functions: VerifyBIB and VerifyBCB.

While these functions are defined for each security block, it is possible that local security policy requires verifying only a subset of the security operations within a given security block.

NOTE – Security verifiers for integrity 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.

NOTE – Security verifiers for confidentiality do not decrypt the target block(s). In order to perform verification of a confidentiality service, the AEAD cipher suite used to

generate cryptographic material must expose its verification function using only the authentication data generated by the security context.

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. For example, if the BPA is the security acceptor for an integrity service, the AcceptBIB function is used instead of the VerifyBIB function.

4.3.2 VERIFYBIB.REQUEST

The VerifyBIB Function is defined for a BPA serving as the security verifier of one or more bib-integrity security services. The function determines whether the integrity result(s) for one or more target blocks present in a BIB created from a call to the ApplyBIB function are equivalent to a local computation of the same integrity result over those target blocks as they exist in the target bundle at the verifying BPA.

NOTE – Failure to match integrity results is not, itself, proof that the target block is corrupt. For example, integrity could fail to verify if the BIB block was corrupted, or the security verifier BPA is misconfigured.

The input parameters of this function include the target blocks whose integrity is being verified, the BIB holding integrity information, and any parameters defined locally to the security verifier associated with the security context used to check integrity. This function only verifies the integrity of those target blocks for which the BPA is configured as a security verifier. Other security operations resident in the same BIB will not have their integrity verified.

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

4.3.2.1 Function

The VerifyBIB.request primitive shall be used to verify a particular BIB.

4.3.2.2 Semantics

VerifyBIB.request shall provide parameters as follows:

VerifyBIB.request(target blocks, BIB to verify, security context, local security context parameters)

4.3.2.3 When Generated
VerifyBIB.request is generated by a BPA that is configured to be a security verifier for the particular BIB instance provided.

4.3.2.4 Effect on Receipt
When invoked, VerifyBIB provides all input parameters to the selected security context, captures the results generated by the security context, determines if the integrity verification succeeded, and generates a VerifyBIB.indication.

4.3.2.5 Discussion - Additional Comments
None

4.3.3 VerifyBCB.request

4.3.3.1 Function

The VerifyBCB.request primitive shall be used to verify a particular BCB.

Security verifiers for confidentiality do not decrypt the target block(s). In order to perform verification of a confidentiality service, the AEAD cipher suite used to generate cryptographic material must expose its verification function using only the authentication data generated by the security context.

4.3.3.2 Semantics
VerifyBCB.request shall provide parameters as follows:

VerifyBCB.request(target blocks, BCB to verify, security context, local security context parameters)

4.3.3.3 When Generated
VerifyBCB.request is generated by a BPA that is configured to be a security verifier for the particular BCB instance provided.

4.3.3.4 Effect on Receipt
When invoked, VerifyBCB provides all input parameters to the selected security context, captures the results generated by the security context, determines if the confidentiality verification succeeded, and generates a VerifyBCB.indication.

4.3.3.5 Discussion - Additional Comments
None

The VerifyBCB Function is defined for a BPA serving as the security verifier of one or more bcb-confidentiality security services in a BCB. The function verifies integrity results associated

with the cipher text associated with the target blocks of the bcb-confidentiality services being verified.
This function must not perform any decryption and must only be used with security contexts that provide an integrity mechanism that can be verified without requiring decryption of cipher text.

The input parameters of this function include the target blocks whose cipher text integrity is being verified, the BCB holding bcb-confidentiality information, and any parameters defined locally to the security verifier associated with the security context used to check authenticity. This function only verifies the integrity of the cipher text associated with target blocks for which the BPA is configured as a security verifier. Other security operations resident in the same BCB will not have their cipher text integrity verified.

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 confidentiality verification succeeded.

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

4.3.4 Input Parameters

4.3.4.1 Overview

The VerifyBIB and VerifyBCB 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 VerifyBIB or VerifyBCB.

4.3.4.2 Target Blocks

The Target Blocks parameter contains the contents of each target block for which integrity or confidentiality is being verified by the function.

4.3.4.3 Security Block

When used with the VerifyBIB function, this parameter consists of the BIB that contains the integrity result(s) generated by the ApplyBIB function for some or all of the target blocks whose integrity is being verified.
When used with the VerifyBCB function, this parameter consists of the BCB that contains the integrity result(s) associated with produced cipher text as generated by the ApplyBCB function for some or all of the target blocks whose integrity is being verified.

4.3.4.4 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 Security Block parameter. These local security context parameters must be provided as inputs to the security context used to verify the integrity or confidentiality 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. When such conflicts exist, local node policy must specify how conflicts are resolved.

4.3.5 VerifyBIB.indication

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

NOTE – If verification of integrity for any of the target blocks provided fails, the VerifyBIB return shall consist of a result code signifying that the integrity of the target blocks was not verified.

4.3.6 VerifyBCB.indication

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

NOTE – If verification of the integrity of the cipher text for any of the target blocks provided fails, the VerifyBCB return shall consist of a result code signifying that the confidentiality of the target blocks was not verified.

4.4 Functions at Security Acceptors

4.4.1 Overview

Security acceptors are BPAs that serve as either the bundle destination or a bundle waypoint configured as the endpoint of one or more security operations within security blocks in a bundle. Upon reaching a security acceptor, a particular security operation is no longer needed and must be removed from its security block using the functions AcceptBIB and AcceptBCB, depending on whether its security service was bib-integrity or bcb-confidentiality, respectively.

4.4.2 AcceptBIB.request

The AcceptBIB Function is defined for a BPA node configured as the security acceptor for one or more security operations within a BIB constructed by the ApplyBIB function.

Similar to the VerifyBIB function, AcceptBIB verifies the integrity of security operations within the BIB for which the BPA is configured as a security acceptor. Additionally, the AcceptBIB function must remove the security operation from its BIB as the security operation must not persist in the bundle after the call to AcceptBIB.

The input parameters of this function include the bundle containing the BIB and associated target blocks, the identifier of the security context used to apply integrity, and any parameters defined local to the acceptor associated with the security context used to check integrity.

The contents of each target block, associated integrity results from 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 operation for each target block is removed from the BIB. If every security operation is removed from the BIB, then the BIB is removed from the bundle.

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

4.4.3 AcceptBCB.request

4.4.4 AcceptBCB.request

4.4.4.1 Function

The AcceptBCB.request primitive shall be used by a security acceptor to verify the integrity of security operations within the BCB for which the BPA is configured as a security acceptor

4.4.4.2 Semantics

AcceptBCB.request shall provide parameters as follows:

AcceptBIB.request(target bundle, BCB to verify, security context, local security context parameters)

4.4.4.3 When Generated

AcceptBCB.request is generated by a CCSDS BPSec implementation that is a security acceptor for a bundle containing the BCB to verify.

4.4.4.4 Effect on Receipt

When invoked, AcceptBCB.request provides all input parameters to the selected security context, captures the results generated by the security context, determines if the confidentiality verification succeeded, and generates a AcceptBCB.indication. If the verification succeeded the AcceptBCB.indication will include decrypted versions of any encrypted blocks.

4.4.4.5 Discussion - Additional Comments

None

The AcceptBCB Function is defined for a BPA node configured as the security acceptor for one or more security operations within a BCB constructed by the ApplyBCB function.

This function removes confidentiality from the target blocks by replacing cipher text within each target block with recovered plain text. The AcceptBCB function must remove the security operation from its BCB as the security operation must not persist in the bundle after the call to AcceptBCB.

The input parameters of this function include the bundle containing the BCB and associated target blocks, the identifier(s) of the target block(s), the identifier of the security context used to apply confidentiality, and any parameters defined local to the acceptor associated with the security context used to process confidentiality.

The contents of each 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 operation is removed from the BCB. If every security operation is removed from the BCB, then the BCB is removed from the bundle.

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

4.4.5 Input Parameters

4.4.5.1 Overview

The AcceptBIB and AcceptBCB 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.

4.4.5.2 Target Bundle

The Target Bundle parameter shall consist of the bundle containing both the security block contaiing security operations being accepted and the target blocks of those security operations.

4.4.5.3 Target Block Identifiers

The Target Block Identifiers parameter uniquely identifies the target blocks within the target bundle of the security operations being accepted.

4.4.5.4 Security Context Identifier

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

4.4.5.5 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 AcceptBIB or AcceptBCB function, respectively.

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

4.4.6 RETURN PARAMETERS

4.4.6.1 AcceptBIB Return

The AcceptBIB Return shall consist of result codes and a modified bundle.

Result codes signify if the integrity of the target blocks were verified. Implementations may choose to return additional result codes to document exactly why integrity verification failed.

The target bundle will be modified such that accepted security operations are removed from the BIB and, if all security operations are removed from a BIB, that the BIB is removed from the bundle.

4.4.6.2 AcceptBCB Return

The AcceptBCB Return shall consist of result codes and a modified bundle.

Result codes signify if the confidentiality of the target blocks were 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, accepted security operations will be removed from the BCB and, if all security operations are removed from a BCB, the BCB will be removed from the bundle. Second, the target blocks 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 blocks will be replaced by the plain text outputs from the security context. If confidentiality removal was not successful, the target block will be removed from the bundle.

PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT (PICS) PROFORMA

(NORMATIVE)

A1 INTRODUCTION

A1.1 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.

A1.2 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.

A1.3 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*, Ihere *i* is a unique identifier, to an accompanying rationale for the noncompliance.

A2 PICS PROFORMA FOR CCSDS BUNDLE PROTOCOL SECURITY

A2.1 GENERAL INFORMATION

A2.1.1 Identification of ICS

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

A2.1.2 Identification of Implementation Under Test

Implementation Name	
Implementation Version	
Special Configuration	
Other Information	

A2.1.3 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)	

A2.1.4 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 [ ]

A2.2 REQUIREMENTS LIST

Classes				
Item	Description	Reference	Status	Support
1	BPSec 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	ASB	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 Abstract 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)	

SECURITY, SANA, AND PATENT CONSIDERATIONS

(INFORMTIVE)

B1 SECURITY CONSIDERATIONS

B1.1 SCOPE

BPSec 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.

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

B1.2 SECURITY CONCERNS

B1.2.1 General

Security concerns informing the design of BPSec are addressed in more detail in references [C1] and [C6]. References [C1] and [C6] additionally 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 BPSec will face and describes how BPSec 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 BPSec threat model is scoped to illustrate threats specific to BPSec operating within DTN environments and therefore focuses on on-path-attackers (OPAs). Reference [C4] describes security threats against space missions.

B1.2.2 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 BPSec 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 BPSec is used, traffic analysis will be possible.

B1.2.3 Data Integrity

Malicious nodes may modify blocks within a bundle, to include replacing existing blocks, adding new blocks, and removing blocks. BPSec 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.

BPSec does not support an in-bundle mechanism to detect (or correct) cases where a malicious node 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 BPSec must be combined with a policy configuration at 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. Further discussion of BPSec policy is found in Annex F.

B1.2.4 Authentication of Communicating Entities

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

B1.2.5 Control of Access to Resources

Resource access controls are not directly addressed by this specification.

B1.2.6 Availability of Resources

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

B1.2.7 Auditing of Resource Usage

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

B1.3 Consequences of not Using BPSec

If BPSec 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 BPSec 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.

B1.4 SANA Considerations

B1.4.1 General

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

B1.4.2 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.

a) **Name**: Security Context Identifiers Registry
b) **Purpose**: Enumerate security contexts defined for use with DTN security protocols within the security context identifier range assigned to SANA from IANA.

c) **Structure**: 3 Columns Table: Value; Description; Reference
d) **Data Types**:
    a. Value: 16-bit unsigned integer representing the enumeration of a security context, which must be in the range of values allocated to SANA from IANA for this purpose.
    b. Description: A string of text describing the security context.
    c. Reference: The formal specification of the security context.
e) **Category**: WG/Local
f) **Review Authority**: SIS Area or designee
g) **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 not be populated with any values.

B1.5 PATENT CONSIDERATIONS

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

INFORMATIVE REFERENCES

(INFORMATIVE)

[C1] *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.

[C2] *Security Guide for Mission Planners*. Issue 2. Report Concerning Space Data System Standards (Green Book) CCSDS 350.7-G-2. Washington, D.C.: CCSDS April 2019.

[C3] E. Rescorla and B. Korver. *Guidelines for Writing RFC Text on Security Considerations*. RFC 3552. Reston, Virginia: ISOC, July 2003.

[C4] *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.

[C5] *Information Processing Systems—Open Systems Interconnection—Basic Reference Model—Part 2: Security Architecture*. International Standard, ISO 7498-2:1989. Geneva: ISO, 1989. 2015.

[C6] *The Application of Security to CCSDS Protocols*. Issue 3. Report Concerning Space Data System Standards (Green Book) CCSDS 350.0-G-3. Washington, D.C.: CCSDS, March 2019.

[C7] E. Birrane, A. White, and S. Heiner. *BPSec Default Security Contexts*. draft-ietf-dtn-bpsec-default-sc-11. July 2021.

[C8] E. Birrane and S. Heiner. *Security Context Template*. Work in Progress. draft-birrane-ietf-dtn-scot-00. July 2020.

[C9] E. Birrane and S. Heiner. *Towards an Interoperable Security Policy for Space-Based Internetworks*. IEEE Space Computing Conference. August 2021.

[C10] E. Birrane and S. Heiner. *A Novel Approach to Transport-Layer Security for Spacecraft*. Smallsat Conference. 2020.

ABBREVIATIONS

(INFORMATIVE)

Term	Meaning
ASB	Abstract Security Block
BCB	Block confidentiality block
BER	Basic Encoding Rules
BIB	Block integrity block
BP	Bundle Protocol
BPA	Bundle Protocol agent
BPSec	Bundle Protocol Security
CBF	Canonical Bundle Block Format
CMS	Cryptographic Message Syntax
COS	class of service
DER	Distinguished Encoding Rules
DTN	Delay-Tolerant Networking
EID	endpoint identifier
HTTP	Hypertext Transfer Protocol
ICV	integrity check value
IETF	Internet Engineering Task Force
IRI	Internationalized Resource Identifier
IV	initialization vector
OPA	on path attacker
RFC	Request for Comments
RL	requirements list
SDNV	self-delimiting numeric value

SRR	status report request
SSP	scheme-specific part
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

ANNEX D

CCSDS PROFILE OF DEFAULT IANA SECURITY CONTEXTS

(NORMATIVE)

D1 OVERVIEW

A default set of BPSec security contexts that can be used for integrity and confidentiality security operations has been defined in Reference [C7]. These security contexts are profiled here without any normative alterations.

This remainder of this section provides a summary of the operation of these security contexts.

D2 ALGORITHM REVIEW

D2.1 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:

a) the key is symmetric;
b) the initialization vector is between 8-16 bytes, with 12 bytes being the recommended length;
c) the output authentication tag is 16 bytes in total length.

D2.2 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:

a) the key must not be less than the length of the keyed hash;
b) the key is symmetric;
c) Hash values are not truncated;
d) The output has value may be one of 256, 384, or 512 bits.

D3 KEY INFORMATION SUMMARY

D3.1 Confidentiality

The baseline implementation uses a symmetric key of 256 bits in length, ~~with key lengths of 128 and 192 also allowed.~~

NOTES

1 It is assumed that the decrypting node either knows the symmetric key used for encryption or can derive that key from encapsulated key material present in the BCB.

2 The configuration, storage, and exchange of keys are outside the scope of this recommendation.

D3.2 Integrity

The baseline implementation recommends a symmetric HMAC key of 384 bits in length, with key lengths of 256 and 512 also supported.

NOTES

1 It is assumed that the node that verifies integrity either knows the HMAC key used when generating the original keyed hash or can derive that key from encapsulated key material present in the BIB.

2 The configuration, storage, and exchange of HMAC keys are outside the scope of this recommendation.

D4 CANONICALIZATION SUMMARY

D4.1 Confidentiality

The baseline implementation uses the canonicalization algorithms defined in Reference [C7] with no exceptions.

D4.2 Integrity

The baseline implementation uses the canonicalization algorithms defined in Reference [C7] with no exceptions.

D5 PROCESSING SUMMARY

D5.1 Confidentiality

D5.1.1 Encryption at a Security Source

The baseline implementation performs encryption using each target block's plain text, an initialization vector (IV), a key, and optional additional authenticated data as inputs to the AES-GCM algorithm.
The cipher text calculated as a result of encrypting the target block must replace each target block's plain text block-type-specific data.

NOTES

1. The IV may be provided as a security context parameter.
2. The authentication tag generated by the security context for each target block is either added as a security result of the BCB or included in the produced cipher text.

D5.1.2 Decrypting at a Security Acceptor

The baseline implementation performs decryption using each target block's cipher text, an IV, key, any additional authenticated data, and the authentication tag as inputs to the AES-GCM algorithm.

The plain text calculated as a result of decrypting the target block must replace each target block's cipher text block-type-specific data.

D5.2 Integrity

D5.2.1 Keyed Hash Generation at a Security Source

The baseline implementation generates a keyed hash using each target block's block-type-specific data, any additional bundle or block information, and an HMAC key as input to the HMAC-SHA algorithm.

Generated keyed hash values are stored as a security result for the security target in the BIB.

D5.2.2 Keyed Hash Verification at a Security Verifier

The baseline implementation generates a keyed hash using each target block's block-type-specific data, any additional bundle or block information, and an HMAC key as inputs to the HMAC-SHA-algorithm. The generated hash is compared to the Integrity Signature

security result of the BIB. If the hashes are identical, integrity verification is successful. Otherwise, integrity of the target block cannot be verified.

D5.2.3 Keyed Hash Verification at a Security Acceptor

The baseline implementation functions similarly to the verification at a security verifier. After verification (success or failure) completes as a security acceptor, the security service is removed from the bundle. If the accepted security service is the only one represented in the block, then the BIB is also removed from the bundle.

ANNEX E

BUNDLE PROTOCOL SECURITY MANAGED INFORMATION

(NORMATIVE)

E1 OVERVIEW

Managed parameters are those parameters provided by management rather than being included in BPSec 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 BPAs in the network.

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 a BPSec block.

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

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

E2 MANAGED PARAMETERS

BPSec managed parameters are those parameters that are independent of the security contexts used to generate cryptographic materials. The managed parameters used for BPSec shall be those listed in table E.1.

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

Table E-1 BPSec Managed Parameters

Managed Parameter	Allowed Values	Defined in Reference
Local BPA parameters used to process BPSec blocks.		
Bundle Node Security Roles	Security Source Security Verifier Security Acceptor	This document
Supported Suites	Integer, Agency-Specific	

Expected security blocks	Block Type(s)	
Known Suite Keys	Algorithm-specific, Agency-Specific	

E3 BPSec Application Data Model

E3.1 Namespace and Nicknames

The namespace and ADM identification for these objects is defined as follows.

Identifier	Value
Namespace	DTN/bpsec
ADM Enumeration	4

Nickname	Collection
80	DTN/bpsec/Const
81	DTN/bpsec/Ctrl
82	DTN/bpsec/Edd
83	DTN/bpsec/Mac
84	DTN/bpsec/Oper
85	DTN/bpsec/Rptt
87	DTN/bpsec/Tblt
89	DTN/bpsec/Var
90	DTN/bpsec/Mdat
91-99	DTN/bpsec/Reserved

E3.2 BPSec Agent ADM JSON Encoding

```
{
  "uses":["Amp:Agent"],
  "Mdat": [
    {
```

```
    "name": "name",
    "type": "STR",
    "value": "bpsec",
    "description": "The human-readable name of the ADM."
  },
  {
    "name": "namespace",
    "type": "STR",
    "value": "DTN:BundleProtocolSecurity",
    "description": "The namespace of the ADM."
  },
  {
    "name": "version",
    "type": "STR",
    "value": "v1.0",
    "description": "The version of the ADM."
  },
  {
    "name": "organization",
    "type": "STR",
    "value": "JHUAPL",
    "description": "The name of the issuing organization of the ADM."
  }
],

"Edd": [
  {
    "name": "num_good_tx_sec_blk",
    "type": "UINT",
    "parmspec": [{"type":"UINT","name":"blk_type"}],
    "description": "Total successfully transmitted security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_bad_tx_sec_blk",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
    "description": "Total unsuccessfully transmitted security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_good_rx_sec_blk",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
    "description": "Total successfully received security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_bad_rx_sec_blk",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
    "description": "Total unsuccessfully received security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_missing_rx_sec_blks",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
    "description": "Total missing-on-RX security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_fwd_sec_blks",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
    "description": "Total forwarded security blocks (1=BIB, 2=BCB, 3=Both)"
  },
  {
    "name": "num_good_tx_sec_bytes",
    "type": "UINT",
    "parmspec": [{"type":"INT","name":"blk_type"}],
```

```
      "description": "Total successfully transmitted security block bytes (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_bad_tx_sec_bytes",
      "type": "UINT",
      "parmspec": [{"type":"INT","name":"blk_type"}],
      "description": "Total unsuccessfully transmitted security block bytes (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_good_rx_sec_bytes",
      "type": "UINT",
      "parmspec": [{"type":"INT","name":"blk_type"}],
      "description": "Total successfully received security block bytes (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "num_bad_rx_sec_bytes",
      "type": "UINT",
      "parmspec": [{"type":"INT","name":"blk_type"}],
      "description": "Total unsuccessfully received security block bytes (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_missing_rx_sec_bytes",
      "type": "UINT",
      "parmspec": [{"type":"INT","name":"blk_type"}],
      "description": "Total missing-on-Rx security block bytes (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "num_fwd_sec_bytes",
      "type": "UINT",
      "parmspec": [{"type":"INT","name":"blk_type"}],
      "description": "Total forwarded security block bytes (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "last_update",
      "type": "TV",
      "description": "Last BPSEC update"
    },
    {
      "name": "num_good_tx_bcb_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}],
      "description": "Number of successfully Tx BCB blocks from SRC"
    },
    {
      "name": "num_bad_tx_sec_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of failed TX security blocks from SRC (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "num_good_rx_sec_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of successfully Rx security blocks from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_bad_rx_sec_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of failed RX security blocks from SRC (1=BIB, 2=BCB, 3=Both)"
    },
    {
```

```
      "name": "num_missing_rx_sec_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of missing-on-RX security blocks from SRC (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "num_fwd_sec_blks_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of forwarded security blocks from SRC (1=BIB, 2=BCB, 3=Both)"
    },
    {
      "name": "num_good_tx_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of successfully Tx security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_bad_tx_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of failed Tx security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_good_rx_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of successfully Rx security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_bad_rx_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of failed Rx security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_missing_rx_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of missing-on-Rx security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "num_fwd_sec_bytes_src",
      "type": "UINT",
      "parmspec": [{"type":"STR","name":"Src"}, {"type":"INT","name":"blk_type"}],
      "description": "Number of forwarded security block bytes from SRC (1=BIB, 2=BCB,
3=Both)"
    },
    {
      "name": "last_update_src",
      "type": "TV",
      "parmspec": [{"type":"STR","name":"Src"}],
      "description": "Last BPSEC update from SRC"
    },
    {
      "name": "last_reset",
      "type": "TV",
      "parmspec": [{"type":"STR","name":"Src"}],
      "description": "Last reset"
    }
  ],
```

```
 "Tblt" : [
   {
     "name": "contexts",
     "columns": [{"type":"UINT","name":"sec_ctx_id"}],
     "description": "This table lists supported security context identifiers."
   }
 ],

"Var": [
 ],

 "Rptt": [
   {
     "name": "full_report",
     "definition" : [
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_good_tx_bcb_blk",
               "ap": [{"type": "ParmName", "value": "Source"}]

       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_bad_tx_bcb_blk",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_good_rx_bcb_blk",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_bad_rx_bcb_blk",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_missing_rx_bcb_blks",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_fwd_bcb_blks",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_good_tx_bcb_bytes",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_bad_tx_bcb_bytes",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_good_rx_bcb_bytes",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_bad_rx_bcb_bytes",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_missing_rx_bcb_bytes",
       },
       {
               "ns":"DTN/bpsec"
               "nm": Edd.num_fwd_bcb_bytes",
       },
```

```
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_good_tx_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_bad_tx_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_good_rx_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_bad_rx_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_miss_rx_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_fwd_bib_blks",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_good_tx_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_bad_tx_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_good_rx_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_bad_rx_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_miss_rx_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_fwd_bib_bytes",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.last_update",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.num_known_keys",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.key_names",
},
{
        "ns":"DTN/bpsec"
        "nm": Edd.ciphersuite_names",
},
{
        "ns":"DTN/bpsec"
```

```
            "nm": Edd.rule_source",
      }
    ],
    "description": "all known meta-data, externally defined data, and
      variables"
  },
  {
    "name": "source_report",
    "parmspec": [{"type":"STR","name":"Source"}],
    "definition": [{
            "ns":"DTN/bpsec",
            "nm": "Edd.num_good_tx_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_bad_tx_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_good_rx_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_bad_rx_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_missing_rx_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_fwd_bcb_blks_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
            "ns":"DTN/bpsec",
            "nm": "Edd.num_good_tx_bcb_bytes_src",
            "ap": [{
            "type": "ParmName",
            "value": "Source"
        }]
      },
      {
```

```
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_tx_bcb_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_good_rx_bcb_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_rx_bcb_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_missing_rx_bcb_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_fwd_bcb_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_good_tx_bib_blks_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_tx_bib_blks_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_good_rx_bib_blks_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_rx_bib_blks_src",
```

```
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_miss_rx_bib_blks_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_fwd_bib_blks_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_good_tx_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_tx_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_good_rx_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_bad_rx_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_missing_rx_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
        "value": "Source"
  }]
},
{
        "ns":"DTN/bpsec",
        "nm": "Edd.num_fwd_bib_bytes_src",
        "ap": [{
        "type": "ParmName",
```

```
                "value": "Source"
            }]
          },
          {
                "ns":"DTN/bpsec",
                "nm": "Edd.last_update_src",
                "ap": [{
                "type": "ParmName",
                "value": "Source"
            }]
          },
          {
                "ns":"DTN/bpsec",
                "nm": "Edd.last_reset",
                "ap": [{
                "type": "ParmName",
                "value": "Source"
            }]
          }
        ],
        "description": "security info by source"
      }
    ],

    "Ctrl": [
    {
        "name": "rst_all_cnts",
        "description": "This control causes the Agent to reset all counts
          associated with block or byte statistics and to set the Last
          Reset Time of the BPsec EDD data to the time when the control
          was run."
      },
      {
        "name": "rst_src_cnts",
        "parmspec": [{"type":"STR","name":"src"}],
        "description": "This control causes the Agent to reset all counts
          (blocks and bytes) associated with a given bundle source and set
          the Last Reset Time of the source statistics to the time when
          the control was run."
      }
    ]
}
```

E4 SECURITY CONTEXT MANAGED PARAMETERS

Security context managed parameters are those parameters associated with the security context of a BPSec block, but managed and provided by the BPAs comprising the security source, security verifier (as 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.

ANNEX F
BUNDLE PROTOCOL SECURITY POLICY CONSIDERATIONS

(INFORMATIVE)

F1 OVERVIEW

BPSec policy is defined as the set of reactions to security operation events occurring throughout the security operation lifecycle. This annex describes the events in the security operation lifecycle, processing actions that may be executed in response to those events, and the way in which these events and actions can be related using security policy.

Figure F-1 illustrates the relationship between security operation events, mandatory processing actions, and the role of policy in determining optional actions. When an occurrence of a security operation event is identified by a BPA, the mandatory processing actions associated with that event are executed. In addition, security policy is examined and any optional processing actions which have been configured for that particular event are executed.

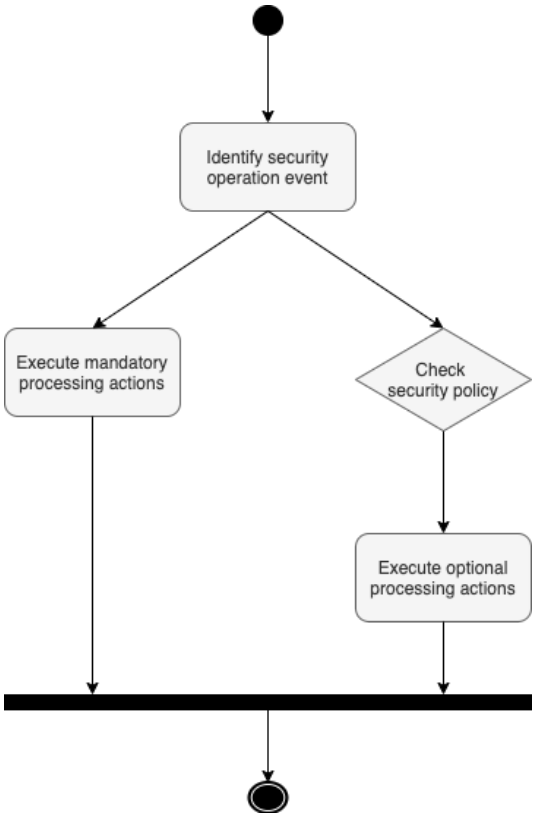

Figure F- 1: Application of Security Policy

F2 SECURITY OPERATION LIFECYCLE

A BPSec security operation, represented in BPSec as a security block, is the application of a security service to a security target block. The lifecycle of a security operation is the set of events that may occur during the creation, processing, and termination of the security service as performed at BPAs in the roles of Security Source, Security Verifier, and Security Acceptor for the security operation.

The events defined in this lifecycle are associated with mandatory and optional processing actions. Optional processing actions are performed or not performed as a function of the local security policy configuration.

F2.1 OVERVIEW

The security operation lifecycle outlined in Figure F-2 is independent of the security service or security context associated with the operation.

The lifecycle is initiated with the source_for_sop event, which designates the current BPA as a Security Source. A BPA assuming the role of a Security Source applies a security policy rule requiring the addition of a security operation to the bundle. Successful addition of the security operation results in the transition to the sop_added_at_source event. If the security operation cannot be added to the bundle, for instance, if a new security block cannot be created or the generated security result(s) cannot be added to the security block, it is designated as misconfigured (sop_misconfigured_at_source) and the security operation is removed, marking the end of the security operation's lifecycle.

During transmission, the bundle may encounter BPA(s) configured to serve as a Security Verifier for the security operation. In this case, the verifier_for_sop event is triggered. If the required security operation cannot be located in the bundle, the sop_missing_at_verifier event occurs. The security operation may be identified as misconfigured (sop_misconfigured_at_verifier) by the BPA for a number of reasons, including the use of an incorrect security context or parameter to generate its cryptographic material. If the sop_corrupted_at_verifier event occurs, verification of the security operation has failed. The sop_verified event is used to indicate successful verification.

When the bundle reaches the Security Acceptor for the security operation, the acceptor_for_sop event is triggered. Failures similar to those at a Security Verifier node may also occur during the final processing of the security operation. The security operation may be identified as missing, misconfigured, or corrupted. If the security operation is successfully processed at the Security Acceptor, the sop_processed event is used to indicate the successful completion of the security operation's lifecycle.

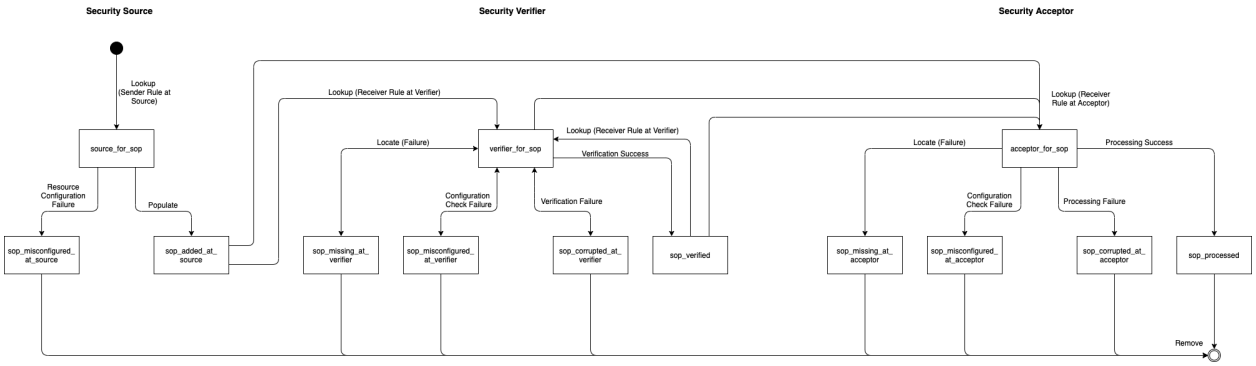

Figure F- 2: The Security Operation Lifecycle

F2.2 SECURITY POLICY FOR SECURITY OPERATION EVENTS

Security policy can be configured for the following security operation events:

a) source_for_sop;
b) sop_added_at_source;
c) sop_misconfigured_at_source;
d) verifier_for_sop;
e) sop_missing_at_verifier;
f) sop_misconfigured_at_verifier;
g) sop_corrupted_at_verifier;
h) sop_verified;
i) acceptor_for_sop;
j) sop_missing_at_acceptor;
k) sop_misconfigured_at_acceptor;
l) sop_corrupted_at_acceptor; and
m) sop_processed.

BPSec policy provides security operation event-level granularity for the definition of security policy. This means that security policy should be configured with consideration for the condition indicated by the security operation event.

F3 SECURITY POLICY RULES

BPSec policy is defined in the context of security policy rules. A security policy rule defines:

a) The bundle(s) and block(s) to which the rule applies;
b) A required security operation for those bundles;
c) The role the BPA must play when applying the rule to those bundles; and
d) The security operation events for which policy is configured in the form of optional processing actions.

F4 PROCESSING ACTIONS

Security operation events may be associated with both mandatory and optional processing actions. Optional processing actions are configurable as a function of security policy and are presented in further detail in reference [C9] and Section F4.3 Optional Processing Actions.

F4.1 PROCESSING ACTION CLASSIFICATION

The categories below are used to classify the capabilities of processing actions:

a) Block manipulation;
b) Bundle manipulation; and
c) Data generation.

F4.1.1 BLOCK MANIPULATION PROCESSING ACTIONS

Block manipulation processing actions may include:

a) Override the security target block's block processing control flags; and
b) Override the security operation's block processing control flags.

F4.1.2 BUNDLE MANIPULATION PROCESSING ACTIONS

Bundle manipulation processing actions may include:

a) Remove the security operation from the bundle;
b) Remove the security target block from the bundle;
c) Remove all security operations for the security target block from the bundle;
d) Retain the security operation in the bundle;
e) Add the security operation to the bundle; and
f) End bundle transmission at the current node.

F4.1.3 DATA GENERATION PROCESSING ACTIONS

Data generation processing actions may include:
a) Report occurrence of the security operation event with reason code; and
b) Request storage of the bundle at the current node.

F4.2 MANDATORY PROCESSING ACTIONS

Mandatory processing actions for a security operation event must be executed each time an occurrence of that event is identified by a BPA. Mandatory processing actions are associated with security operation events regardless of security policy configuration.

For example, the source_for_sop security operation event is associated with a mandatory processing action. The event indicates that the current BPA has been identified as the Security Source for a security operation. That BPA executes the mandatory processing action by adding a security block to the bundle, representing the security operation.

F4.3 OPTIONAL PROCESSING ACTIONS

Optional processing actions are configurable for security operation events. Any enabled, optional processing actions are executed when an occurrence of that security operation event is identified.

Multiple optional processing actions can be enabled for a single security operation event.

F5 POLICY CONSIDERATIONS

The following capabilities must be provided in order to implement BPSec policy.

F5.1 ASSOCIATION OF SECURITY OPERATION EVENTS WITH PROCESSING ACTIONS

The relationship between a security operation event and its associated processing actions must be established. An event is coupled to both the required and optional processing actions to be performed when an occurrence of that event is identified.

As a function of security policy, the optional processing actions configured for a security operation event must be identified and executed when that event occurs. This relationship is represented below, where P represents the security policy function which makes the association, E is the security operation event for which the processing action(s) are identified, and A is the set of processing actions which must be executed.

$$P(E) \rightarrow A$$

Note that the set of processing actions for an event can be described as the union of mandatory processing actions (M) and optional processing actions (O) associated with that event. There may be zero or more actions in either of these sets.

$$A = M \cup O$$

F5.2 EXECUTION OF PROCESSING ACTIONS

As noted in the above section, the set of processing actions which must be executed for an occurrence of a security operation event may consist of both mandatory and optional processing actions. It is recommended that any processing actions which are categorized as data generation actions are executed first, in order to preserve the original bundle state for later analysis.

Block manipulation processing actions should be executed after data generation actions. When all block manipulation processing actions have been applied, bundle manipulation actions may be performed.

ANNEX G

SERVICE SPECIFICATION FOR BUNDLE PROTOCOL SECURITY POLICY

(INFORMATIVE)

G1 OVERVIEW

BPSec policy is defined in the context of security policy rules, which define a security operation, the bundles for which that security operation is required to be represented, the BPA role, and the optional processing actions that BPA must execute in response to security operation events.

The management of BPSec policy requires the ability to define and configure security policy rules and security operation event sets. As seen in Figure G-1 below, one or more security policy rules may map to an event set. An event set is composed of one or more events.

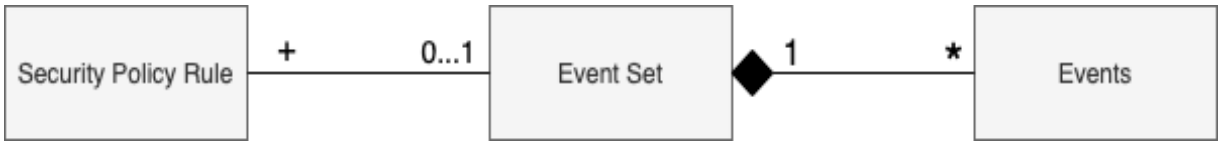

Figure G-1: Mapping Security Policy Rules to Event Sets Composed of Events

G2 EVENT SETS

G2.1 OVERVIEW

An event set defines the security operation event(s) for which one or more optional processing actions are enabled by the BPSec policy configuration.

An event set can be modified by adding and deleting events until that event set is associated with a security policy rule. Further discussion of events can be found in Section G3 Events.

G2.2 SERVICES

The following are the core capabilities which must be provided for event sets.

a) Create a named event set for which the name of the event set serves as a unique identifier;
b) Delete a named event set; and
c) List all event sets currently defined.

G3 EVENTS

G3.1 OVERVIEW

An event set can be modified by adding and deleting events belonging to the event set until that event set is associated with a security policy rule. Once a security policy rule is created which references the event set, it should no longer be modified.

An event defines the optional processing actions to be executed in response to the occurrence of a security operation event by policy. Multiple processing actions may be specified for a single security operation event.

G3.2 SERVICE PARAMETERS

Event set services may accept the following information:

a) Event Set Name: The unique name used to identify the security operation event set.

b) Security Operation Event: The security operation event for which optional processing actions are configured.

c) Optional Processing Actions: The optional processing action(s) to enable for the specified security operation event.

d) Processing Action Parameters: Optional parameters to accompany the processing action selected to be enabled. For example, if the optional processing action enabled requires the reporting of a reason code, that reason code is provided as an additional parameter.

G3.3 SERVICES

The following are the core capabilities which must be provided for events.

a) Add an event to an event set; and
b) Delete an event from an event set.

G4 SECURITY POLICY RULES

G4.1 SERVICE PARAMETERS

A security policy rule service may accept the following types of information:

1) Filter criteria

2) Specification criteria
3) Event criteria

G4.1.1 Filter Criteria

The filter criteria for a security policy rule service are used to identify which bundles and which blocks in the bundle are impacted by application of the rule.

When the security policy role identified in the filter criteria field is the Security Source, the bundle for which the rule applies to is identified using the bundle source, bundle destination, and target block type information provided in the filter criteria.

When the security policy role identified in the filter criteria field is either the Security Verifier or Security Acceptor, the bundle for which the rule applies to is identified using the bundle source, bundle destination, and target block type information provided. Then, to identify the security block in the bundle for which the rule applies, the target block type and security context identifier fields must be used.

The following fields may be present in the filter criteria for a security policy rule:

a) Security Policy Role: Security Source, Security Verifier, or Security Acceptor.

b) Bundle Source EID: The endpoint identifier of the bundle source.

c) Bundle Destination EID: The endpoint identifier of the bundle destination.

d) Target Block Type: The block type identifier of the security operation target.

e) Security Context Identifier: The security context used when processing the security operation.

G4.1.2 Specification Criteria

The specification criteria field for a security policy rule provides the information required to identify and apply the security operation described by the rule.

The following fields may be present in the specification criteria for a security policy rule:

a) Security Service: The security service, bib-integrity or bcb-confidentiality, associated with the security operation.

b) Security Context Name: The security context to use when applying the security operation to the bundle.

c) Security Context Parameters: Optional parameters to be used by the security context when applying the security operation to the bundle.

G4.1.3 EVENT CRITERIA

The event criteria field for a security policy rule is used to associate the rule with either a named event set or an anonymous event set. The event set determines how security operation events are handled as a matter of policy.

The following fields may be present in the event criteria for a security policy rule:

a) Event Set Name: The unique name used to identify the security operation event set.

b) Security Operation Event: The security operation event for which optional processing actions are to be enabled.

c) Optional Processing Actions: The optional processing action(s) to enable for the specified security operation event.

d) Processing Action Parameters: Optional parameters to accompany the processing action selected to be enabled.

G4.2 SERVICES

The following are the core capabilities which must be provided to support the use of security policy rules.

a) Create a security policy rule, associated with either a named event set that has been defined in the system or an anonymous event set defined during the creation of the security policy rule itself;
b) Delete a security policy rule; and
c) Display the information of the security policy rule(s) matching the provided filter criteria.