

**Information Processing Systems - Telecommunications and
Information Exchange between Systems - Protocol for Exchange of
Inter-domain Routeing Information among Intermediate Systems
to Support Forwarding of ISO 8473 PDUs**

March 16, 1992

Status

This document is the revised IDRP text that the Project Editor has prepared in accordance with the instructions given in SC6 N7089 (Summary of Voting on CD 10747...). The associated Disposition of Comments report is contained in SC6 N7194.

Contents

1. Scope and Field of Application	1	8.5 Reliable Delivery of BISPDUs	23
2. Normative References	1	8.5.1 Checksum over BISPDUs	23
3. Informative References	2	8.5.2 Sequence Numbers	23
4. Definitions	2	8.5.3 Flow Control	24
4.1 Reference Model Definitions	2	8.6 BIS-BIS Connection Management	25
4.2 Network Layer Architecture Definitions	3	8.6.1 BIS Finite State Machines	25
4.3 Network Layer Addressing Definitions	3	8.6.2 Closing a Connection	29
4.4 Routeing Framework Definitions	3	8.7 Version Negotiation	29
4.5 Intra-domain Routeing Definitions	3	8.8 Checksum Algorithm	29
4.6 Additional Definitions	3	8.9 Authentication Mechanism	29
5. Symbols and Abbreviations	4	8.10 Routeing Information Bases	31
5.1 Data Unit Abbreviations	4	8.10.1 Identifying an Information Base	31
5.2 Addressing Abbreviations	4	8.10.2 Validation of RIBs	31
5.3 Other Abbreviations	5	8.10.3 Use of the RIB REFRESH PDU	33
6. General Protocol Information	5	8.11 Path Attributes	33
6.1 Inter-RD Topology	5	8.11.1 Categories of Path Attributes	33
6.2 Routeing Policy	7	8.11.2 Handling of Distinguishing Attributes	34
6.3 Types of Systems	7	8.11.3 Equivalent Distinguishing Attributes	35
6.4 Types of Routeing Domains	7	8.12 Path Attribute Usage	35
6.5 Routeing Domain Confederations	7	8.12.1 ROUTE-ID	35
6.6 Routes: Advertisement and Storage	7	8.12.2 EXT_INFO	35
6.7 Distinguishing Path Attributes and RIB-Atts	8	8.12.3 RD_PATH	36
6.8 Selecting the Information Bases	8	8.12.4 NEXT_HOP	36
6.9 Routeing Information Exchange	8	8.12.5 DIST_LIST_INCL	37
6.9.1 Internal Neighbor BIS	8	8.12.6 DIST_LIST_EXCL	38
6.9.2 External Neighbor BIS	8	8.12.7 MULTI-EXIT_DISC	39
6.10 Routeing Domain Identifiers	9	8.12.8 LOC_PREF	39
6.11 Formats of RDIs, NETs, and NSAP Addresses	9	8.12.9 TRANSIT DELAY	39
6.12 Design Objectives	9	8.12.10 RESIDUAL ERROR	39
6.12.1 Within the Scope of the Protocol	9	8.12.11 EXPENSE	40
6.12.2 Outside the Scope of the Protocol	10	8.12.12 SOURCE SPECIFIC QOS	40
7. Structure of BISPDUs	10	8.12.13 DESTINATION SPECIFIC QOS	40
7.1 Header of BISPDUs	11	8.12.14 HIERARCHICAL RECORDING	41
7.2 OPEN PDU	11	8.12.15 RD_HOP_COUNT	41
7.3 UPDATE PDU	13	8.12.16 SOURCE SPECIFIC SECURITY	41
7.4 IDRP ERROR PDU	19	8.12.17 DESTINATION SPECIFIC SECURITY	42
7.5 KEEPALIVE PDU	20	8.12.18 CAPACITY	42
7.6 CEASE PDU	20	8.12.19 PRIORITY	42
7.7 RIB REFRESH PDU	20	8.13 Routeing Domain Confederations	42
8. Elements of Procedure	21	8.13.1 RDC Policies	43
8.1 Naming and Addressing Conventions	21	8.13.2 RDC Configuration Information	43
8.1.1 Interpretation of Address Information	21	8.13.3 Detecting Confederation Boundaries	43
8.1.2 NSAP Address Prefixes	21	8.13.4 Entry Marker for an RDC	43
8.2 Deployment Guidelines	21	8.13.5 Generating RD_PATH Information	43
8.2.1 Minimum Configuration of an RD	21	8.13.6 Updating Received RD_PATH Information	44
8.2.2 Deployment of ISs and ESs	22	8.14 Update-Receive Process	44
8.3 Domain Configuration Information	22	8.14.1 Information Consistency	45
8.4 Advertising NLRI Information	22	8.14.2 Detecting Inconsistencies	45
		8.15 Decision Process	45
		8.15.1 Phase 1: Calculation of Degree of Preference	46
		8.15.2 Phase 2: Route Selection	46
		8.15.3 Phase 3: Route Dissemination	47
		8.15.4 Interaction with Update Process	48
		8.16 Update-Send Process	48
		8.16.1 Internal Updates	48
		8.16.2 External Updates	49
		8.16.3 Controlling Routeing Traffic Overhead	49
		8.17 Efficient Organization of Routeing Information	50
		8.17.1 Information Reduction	50

2	8.17.2 Aggregating Routeing Information	50	A.3.1 General Structure of the PICS Proforma	78
2	8.18 Maintenance of the Forwarding Information Bases	52	A.3.2 Additional Information	78
	8.19 Receive Process	52	A.3.3 Exception Information	79
	8.20 Error Handling for BISPDUs	53	A.3.4 Conditional Status	79
	8.20.1 BISPDU Header Error Handling	53	A.4 Identification	80
	8.20.2 OPEN PDU Error Handling	53	A.4.1 Implementation Identification: IDRP	80
	8.20.3 UPDATE PDU Error Handling	53	A.4.2 Protocol Summary: IDRP	81
	8.20.4 IDRP ERROR PDU Error Handling	55	A.4.3 PICS Proforma: IDRP General	81
	8.20.5 Hold Timer Expired Error Handling	55	A.4.4 PICS Proforma: IDRP Update Send Process	81
1	8.20.6 KEEPALIVE PDU Error Handling	55	A.4.5 PICS Proforma: IDRP Update Receive Process	82
1	8.20.7 CEASE PDU Error Handling	55	A.4.6 PICS Proforma: IDRP Decision Process . . .	82
1	8.20.8 RIB REFRESH PDU Error Handling	55	A.4.7 PICS Proforma: IDRP Receive Process . . .	82
	9. Forwarding Process for CLNS	55	A.4.8 PICS Proforma: IDRP CLNS Forwarding . . .	82
	9.1 Forwarding to Internal Destinations	55	A.4.9 PICS Proforma: IDRP Optional Transitive Attributes	83
	9.2 Determining the NPDU-derived Distinguishing Attributes	55	A.4.10 PICS Proforma: IDRP Well Known Discretionary Attributes	83
	9.3 Matching RIB-Att to NPDU-derived Distinguishing Attributes	56	Appendix B. IDRP Checksum Generation Algorithm	85
	9.4 Forwarding to External Destinations	57	B.1 Mathematical Notation	85
	10. Interface to ISO 8473	57	B.2 Algorithm Description	85
	11. Constants	58	Appendix C. Jitter Algorithm	87
	12. System Management and GDMO Definitions	59	Appendix D. Computing a Checksum for an Adj-RIB	87
	12.1 Name Binding	59	Appendix E. RIB Overload	87
	12.2 Local BIS Managed Objects for IDRP	59	Appendix F. Processor Overload	89
	12.3 Adjacent BIS Peer Managed Objects	62	Appendix G. Syntax and Semantics for Policy	89
	12.4 Action Definitions	69	Appendix H. Importing Internal Reachability Information	93
	12.5 Notification Definitions	69	Appendix I. Formation of RDCs	93
	12.6 Parameter Definitions	70	I.1 Forming a New Lower Level Confederation . . .	94
	12.7 Attribute Groups	71	I.2 Forming a Higher Level Confederation	94
	12.8 ASN.1 MODULES	72	I.3 Deleting a Lowest Level Confederation	95
	13. Conformance	76	I.4 Deleting a Higher Level Confederation	95
	13.1 Static Conformance for All BISs	76	Appendix J. Example Usage of MULTI-EXIT_DISC Attribute	95
	13.2 Conformance to Optional Functions	76	Appendix K. Example Routeing Algorithm	97
	13.2.1 Generation of Information in Reduced Form	76	Index	100
	13.2.2 Supporting RDCs	76		
	13.2.3 Generation of Well-known Discretionary Attributes	77		
	13.2.4 Propagation of Well-known Discretionary Attributes	77		
	Appendix A. PICS Proforma	78		
	A.1 Introduction	78		
	A.2 Abbreviations and Special Symbols	78		
	A.2.1 Status Symbols	78		
	A.3 Instructions for Completing the PICS Proforma	78		

Figures

1. Field of Application	2
2. Intermediate Routeing Domains and End Routeing Domains	4
3. Position of IDRP within Network Layer	6
4. Inter-domain Routeing Components	6
5. Structure of the UPDATE PDU	14
6. Validation Pattern in Support of the Authentication Function	30
7. Routeing Information Base	32
8. A Transitive Fully Connected Subnetwork	37
9. Illustrative PICS for Conditional Items	80
10. Jitter Algorithm for Timers	88
11. Example Interconnection of Routeing Domains	92
12. Example 1 Configuration	96
13. Example 2 Configuration	96

Tables

1. The IDRP Information Bases	9
2. BIS Finite State Machine	27
3. Path Attribute Characteristics	34
4. NPDU-Derived Attribute Set	56
5. IDRP-CL Primitives	57
6. Architectural Constants of IDRP	58
7. PICS Proforma for IDRP: Implementation Identification	80
8. PICS Proforma for IDRP: Protocol Summary	81
9. PICS Proforma for IDRP: General	81
10. PICS Proforma for IDRP: Update-Send	81
11. PICS Proforma for IDRP: Update-Receive	82
12. PICS Proforma for IDRP: Decision	82
13. PICS Proforma for IDRP: Receive	82
14. PICS Proforma for IDRP: CLNS Forwarding	82
15. PICS Proforma for IDRP: Optional Transitive Attributes	83
16. PICS Proforma for IDRP: Generating Well Known Discretionary Attributes	83
17. PICS Proforma for IDRP: Propagating Well Known Discretionary Attributes	84
18. PICS Proforma for IDRP: Receiving Well Known Discretionary Attributes	85

Introduction

This Protocol is one of a set of International Standards which facilitate the interconnection of open systems. They cover the services and protocols required to achieve such interconnection.

This Protocol is positioned with respect to other related standards by the layered structure defined in ISO 7498, and by the Network layer organization defined in ISO 8648. It is located at the top of the Network layer and relies on the services of ISO 8473. This protocol permits a routing domain to exchange information with other routing domains to facilitate the operation of the routing and relaying functions of the Network Layer. It applies to the following categories of routing, which are described in ISO TR 9575, making no distinction between them:

- Intra-Administrative Domain routing between routing domains
- Inter-Administrative Domain routing between routing domains.

Within the hierarchical relations between routing protocols, as described in ISO TR 9575, this protocol is situated above the intra-domain routing protocols. That is, this Inter-domain IS-IS protocol:

- maintains information about the interconnections between routing domains, but does not require detailed information about their internal structures
- calculates path segments on a hop-by-hop basis

This protocol calculates path segments which consist of *Boundary Intermediate systems* and the links that interconnect them. An NPDU destined for an End system in another routing domain will be routed via

Intra-domain routing to a Boundary Intermediate system (BIS¹) in the source routing domain. Then, the BIS, using the methods of this inter-domain routing protocol, will calculate a path to a Boundary Intermediate system in an adjacent routing domain lying on a path to the destination. After arriving at the next routing domain, the NPDU may also travel within that domain on its way towards a BIS located in the next domain along its path. This process will continue on a hop-by-hop basis until the NPDU arrives at a BIS in the routing domain which contains the destination End system. The Boundary IS in this routing domain will hand the incoming NPDU over to the domain's intra-domain routing protocol, which will construct a path to the destination End system.

This inter-domain IS-IS routing protocol places requirements on the type of information that a routing domain must provide and on the methods by which this information will be distributed to other routing domains. These requirements are intended to be minimal, addressing only the interactions between Boundary ISs; all other internal operations of each routing domain are outside the scope of this protocol.²

The methods of this protocol differ from those generally adopted for an intra-domain routing protocol because they emphasize the interdependencies between efficient route calculation and the preservation of legal, contractual, and administrative concerns. This protocol calculates routes which will be efficient, loop-free, and in compliance with the domain's local routing policies. IDRP may be used when routing domains do not fully trust each other; it imposes no upper limit on the number of routing domains that can participate in this protocol; and it provides isolation between its operations and the internal operations of each routing domain.

¹ See clause 5 for a listing of symbols and abbreviations.

² For example, this Inter-domain routing protocol does not mandate that a routing domain run a particular intra-domain routing protocol: it is local choice as to whether a domain implements a standard intra-domain protocol (such as ISO/IEC 10589) or a private protocol.

Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol for Exchange of Inter-domain Routeing Information among Intermediate Systems to Support Forwarding of ISO 8473 PDUs

1. Scope and Field of Application

This international standard specifies a protocol to be used by Boundary Intermediate systems³ to acquire and maintain information for the purpose of routeing NPDU's between different routeing domains. Figure 1 illustrates the field of application of this international standard.

This international standard specifies:

- the procedures for the exchange of inter-domain reachability and path information between BISs
- the procedures for maintaining inter-domain routeing information bases within a BIS
- the encoding of protocol data units used to distribute inter-domain routeing information between BISs
- the functional requirements for implementations that claim conformance to this standard

The procedures are defined in terms of:

- interactions between Boundary Intermediate systems through the exchange of protocol data units
- interactions between this protocol and the underlying Network Service through the exchange of service primitives
- constraints on policy feasibility and enforcement which must be observed by each Boundary Intermediate system in a routeing domain

The boundaries of Administrative Domains are realized as artifacts of the placement of policy constraints and the aggregation of network layer reachability information; they are not manifested explicitly in the protocol. The protocol described in this international standard operates at the level of individual routeing domains. The establishment of administrative domains is outside the scope of this standard.

2. Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below.

Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 7498: 1984, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model*

ISO 7498/Add. 1: 1984, *Information Processing Systems - Open Systems Interconnection - Addendum to ISO 7498 Covering Connectionless-mode Transmission*

ISO 7498/Add. 3: 1984, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 3: Naming and Addressing*

ISO/DIS 7498/Add. 4⁴, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 4: OSI Management Framework*

ISO 8348: 1987, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Network Service Definition*

ISO 8348/Add. 1: 1987, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Addendum to the Network Service Definition Covering Connectionless-mode Transmission*

ISO 8348/Add.2: 1988, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Addendum to the Network Service Definition Covering Network Layer Addressing*

³ An Intermediate system that implements this protocol is known as a Boundary Intermediate system (BIS).

⁴ To be published

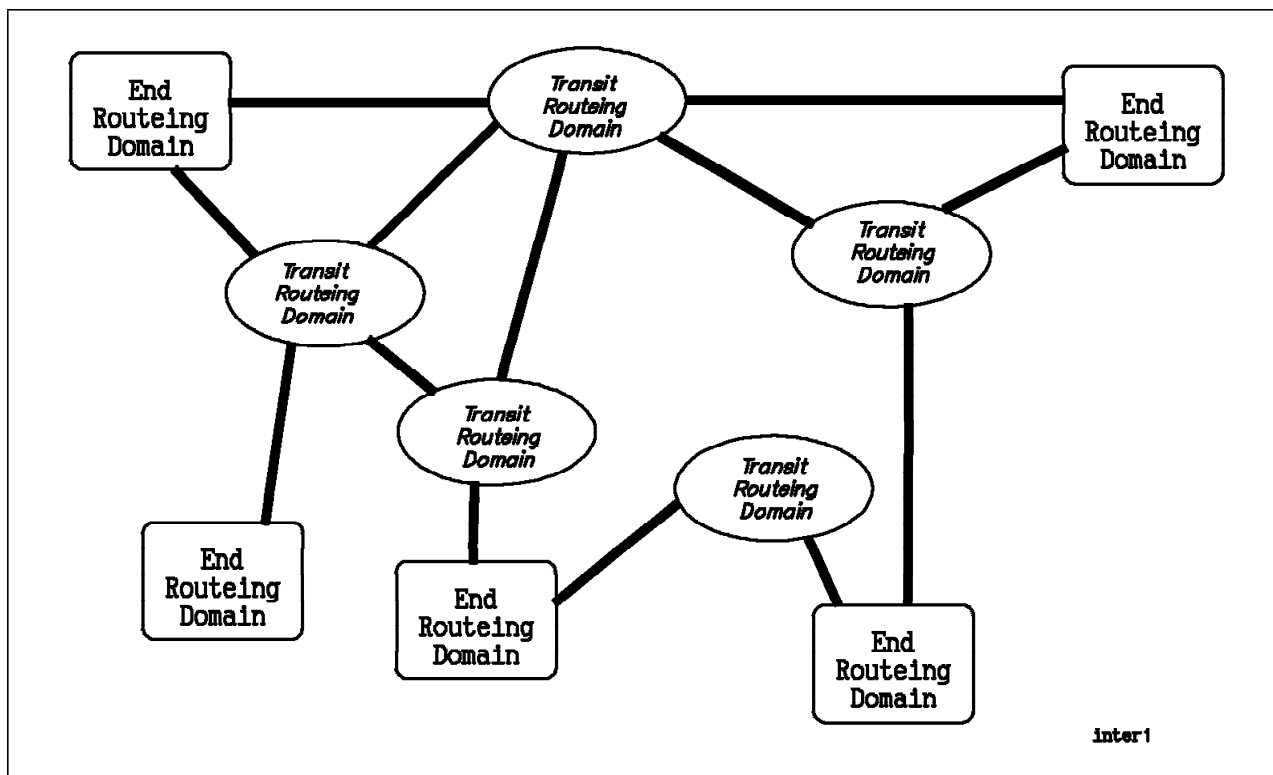


Figure 1. Field of Application. The Inter-domain Routing Protocol operates between routing domains; intra-domain routing is not within its scope.

ISO 8473: 1988, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol for Providing the Connectionless-mode Network Service*

ISO 8648: 1988, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Internal Organization of the Network Layer*

ISO TR 9577: 1991, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol identification in the Network Layer*

ISO TR 9575: 1989, *Information Processing Systems - Telecommunications and Information Exchange between Systems - OSI Routing Framework*

ISO/IEC 10589⁴, *Information Processing Systems - Telecommunications and Information Exchange between Systems - Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol for use in Conjunction with the protocol for providing the Connectionless-mode Network Service (ISO 8473)*

- 1 **RFC 1186**, *The MD4 Message Digest Algorithm*, R. Rivest, October 1990

3. Informative References

The following references contain information which is helpful in understanding the protocol described in this international standard, and for setting the context in which it might be deployed.

ISO 9542:1988, *Information Processing Systems - Telecommunications and Information Exchange between Systems - End system to Intermediate system routing exchange protocol for use in conjunction with the Protocol for providing the connectionless-mode network service (ISO 8473)*

4. Definitions

4.1 Reference Model Definitions

This International Standard uses the following terms defined in ISO 7498:

- a) (N)-entity title
- b) Network entity
- c) Network Layer
- d) Network Protocol
- e) Network Protocol Data Unit
- f) Network relay
- g) Network Service Access Point

- h) Network Service Access Point Address
- i) Real system
- j) Routeing

4.2 Network Layer Architecture Definitions

This International Standard uses the following terms defined in ISO 8648:

- a) End system
- b) Intermediate System
- c) Subnetwork

4.3 Network Layer Addressing Definitions

This International Standard uses the following terms defined in ISO 8348/AD2:

- a) Subnetwork point of attachment

4.4 Routeing Framework Definitions

This International Standard uses the following terms defined in ISO 9575:

- a) Administrative Domain
- b) Common Domain
- c) Fire wall
- d) Routeing Domain

4.5 Intra-domain Routeing Definitions

This International Standard uses the following terms defined in IS 10589:

- a) Adjacency
- b) Link

4.6 Additional Definitions

For purposes of this International Standard, the following definitions are used:

Intra-domain IS-IS routeing protocol: A routeing protocol that is run between Intermediate systems in a single routeing domain to determine routes that pass through only systems and links wholly contained within the domain.

NOTE: Unless reference is made to a specific protocol, this term is used a general designator, encompassing both private and internationally standardized protocols.

- 4 **Inter-domain link:** A real (physical) or virtual (logical)
- 4 link between two or more Boundary Intermediate
- 4 systems (see Figure 2). A link between two BISs in
- 4 the same routeing domain carry both intra-domain
- 4 traffic and inter-domain traffic; a link between two
- 4 BISs located in adjacent routeing domains can
- 4 carry inter-domain traffic, but not intra-domain
- 4 traffic.

Boundary Intermediate system: An intermediate system that runs the protocol specified in this standard, has at least one inter-domain link attached to it, and may optionally have intra-domain links attached to it.

End Routeing Domain: A routeing domain whose local policies permit its BISs to calculate inter-domain path segments only for PDUs whose source is located within that routeing domain. There are two varieties of End routeing domains: stub and multi-homed. A stub ERD has inter-domain links to only one adjacent routeing domain, while a multi-homed ERD has inter-domain links to several adjacent routeing domains.

For example, the domains labelled as multi-homed ERDs in Figure 2 have policies which prohibit them from providing relaying functions; it is these policies, not the topology of their interconnections, that make them ERDs.

Transit Routeing Domain: A routeing domain whose policies permit its BISs to calculate inter-domain path segments for PDUs whose source is located either in the local routeing domain or in a different routeing domain. That is, it can provide a relaying service for such PDUs. See Figure 2 for an illustration of TRDs.

Adjacent RDs: Two RDs ("A" and "B") are adjacent to one another if there is at least one pair of BISs, one located in "A" and the other in "B", that are attached to each other by means of a real subnetwork.

RD Path: A list of the RDs of the routeing domains and routeing domain confederations through which a given UPDATE PDU has travelled.

Routeing Domain Confederation: A set of routeing domains which have agreed to join together and to conform to the rules of clause 8.13 of this international standard. To the outside world, a confederation is indistinguishable from a routeing domain.

Nested RDCs: A routeing domain confederation "A" (RDC-A) is nested within RDC-B when all of the following conditions are satisfied simultaneously:

- a) all members of RDC-A are also members of RDC-B
- b) there are some members of RDC-B that are not members of RDC-A

Overlapping RDCs: A routeing domain confederation (RDC-A) overlaps RDC-B when all the following conditions are satisfied simultaneously:

- a) there are some members of RDC-A that are also members of RDC-B, and
- b) there are some members of RDC-A that are not members of RDC-B, and
- c) there are some members of RDC-B that are not members of RDC-A.

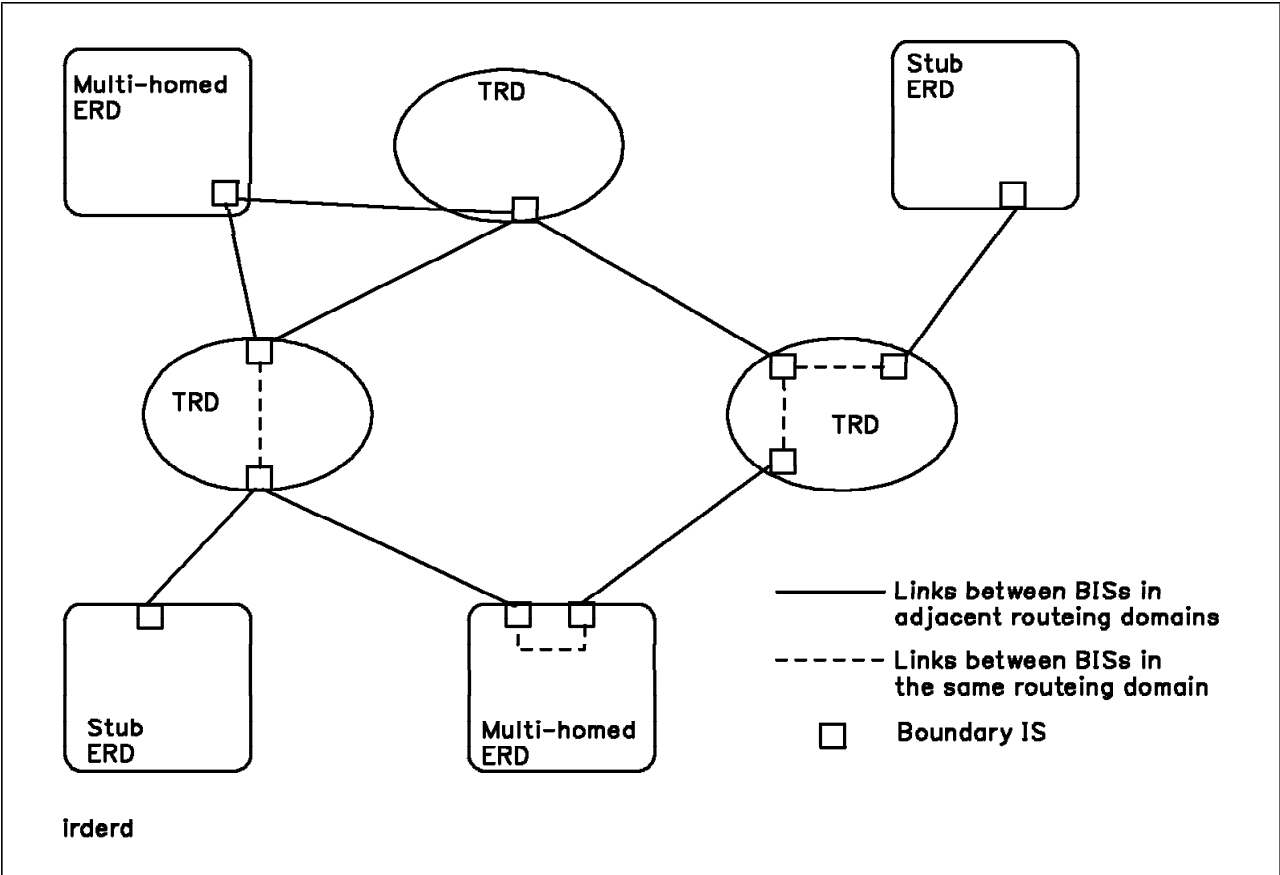


Figure 2. Intermediate Routing Domains and End Routing Domains. The classification of a routing domain as an TRD or an ERD depends upon both its interconnections to other routing domains and its relaying policies with respect to them.

Disjoint RDCs: Two routing domain confederations, RDC-A and RDC-B, are disjoint from one another when there are no routing domains which are simultaneously members of both RDC-A and RDC-B.

Policy Information Base: The collection of routing policies that a BIS will apply to the routing information that it learns using this International standard. It is not required that all routing domains use the same syntax and semantics to express policy; that is, the format of the Policy Information Base is left as a local option.

- 2 **Route Origin:** Each route or component of an aggregated route has a single unique origin. This is the
- 2 RD or RDC in which the route's destinations are
- 2 located.

5. Symbols and Abbreviations

The symbols, acronyms, and abbreviations listed in the following clauses are used in this International Standard.

5.1 Data Unit Abbreviations

- BISPDU** Boundary Intermediate System PDU
- DT PDU** ISO 8473 Data Protocol Data Unit
- ER PDU** ISO 8473 Error Protocol Data Unit
- NPDU** Network Protocol Data Unit
- NSDU** Network Service Data Unit
- PDU** Protocol Data Unit

5.2 Addressing Abbreviations

- AFI** Authority and Format Identifier
- DSP** Domain Specific Part
- IDI** Initial Domain Identifier
- IDP** Initial Domain Part
- NET** Network Entity Title
- NPAI** Network Protocol Address Information
- NSAP** Network Service Access Point
- SNPA** Subnetwork Point of Attachment

5.3 Other Abbreviations

BIS	Boundary Intermediate System
CL	Connectionless Mode
CLNS	Connectionless Mode Network Service
CM	Confederation Member
ERD	End Routeing Domain
ES	End System
FIB	Forwarding Information Base
IDRP	Inter-domain Routeing Protocol (an acronym for the protocol described in this international standard)
MIB	Management Information Base
NLRI	Network layer reachability information
OSIE	OSI Environment
PCI	Protocol Control Information
PIB	Policy Information Base
QOS	Quality of Service
RDC	Routeing Domain Confederation
RDI	Routeing Domain Identifier
RIB	Routeing Information Base
SPI	Subsequent Protocol Identifier
SNICP	Subnetwork independent convergence protocol
TRD	Transit Routeing Domain

6. General Protocol Information

IDRP is a routeing information exchange protocol which is located within the Network layer and interfaces to ISO 8473, which serves as a SNICP (see Figure 3). In particular, BISPDU's are encapsulated as the data portion of ISO 8473 NPDUs. IDRP is a connection-oriented protocol which is implemented only in Intermediate systems. Routeing and control information is carried in BISPDU's (described in clause 7), which flow on connections between pairs of BISs. Each BISPDU is packaged within one or more NPDUs for transmission by the underlying Network service. IDRP relies on the underlying Network service to provide for fragmentation and reassembly of BISPDU's. IDRP queues Outbound BISPDU's as input to the underlying Network Layer service, retaining a copy of each BISPDU until an acknowledgement is

received. Similarly, inbound BISPDU's are queued as input to the BISPDU-Receive process.

IDRP exchanges BISPDU's in a reliable fashion. It provides mechanisms for the ordered delivery of BISPDU's and for the detection and retransmission of lost or corrupted BISPDU's. The mechanisms for achieving reliable delivery of BISPDU's are described in clause 8.5; methods for establishing BIS-BIS connections are described in clause 8.6.

IDRP is consistent with the routeing model presented in ISO TR 9575. To emphasize its policy-based nature, the IDRP routeing model includes a Policy Information Base, as shown in Figure 4. IDRP can be described in terms of four major components:

- a) **BISPDU-Receive Process:** responsible for accepting and processing control and routeing information from the local environment and from BISPDU's of other BISs. This information is used for a variety of purposes, such as receiving error reports and guaranteeing reliable reception of BISPDU's from neighboring BISs. (For example, the Update-Receive process (see 8.14) is the part of the BISPDU-Receive process that deals with the reception of routeing information after a BIS-BIS connection has been established.)
- b) **BISPDU-Send Process:** responsible for constructing BISPDU's which contain control and routeing information which is used by the local BIS for a variety of purposes, such as advertising routeing information to other BISs, initiating BIS-BIS communication, and validating BIS routeing information bases.
- c) **Decision Process:** responsible for calculating routes which will be consistent with local routeing policies. It operates on information in both the PIB and the Adj-RIBs, using it to create the Local RIBs (Loc-RIBs) and the local Forwarding Information Bases (see clause 8.10).
- d) **Forwarding Process:** responsible for supplying resources to accomplish relaying of NPDUs to their destinations. It uses the FIB(s) created by the Decision Process.

6.1 Inter-RD Topology

This protocol views the overall global OSIE as an arbitrary interconnection of Transit Routeing Domains and End Routeing Domains which are connected by real inter-domain links placed between BISs located in the respective routeing domains. This standard provides for the direct exchange of routeing information between BISs, which may be located either in the same routeing domain or in adjacent routeing domains.

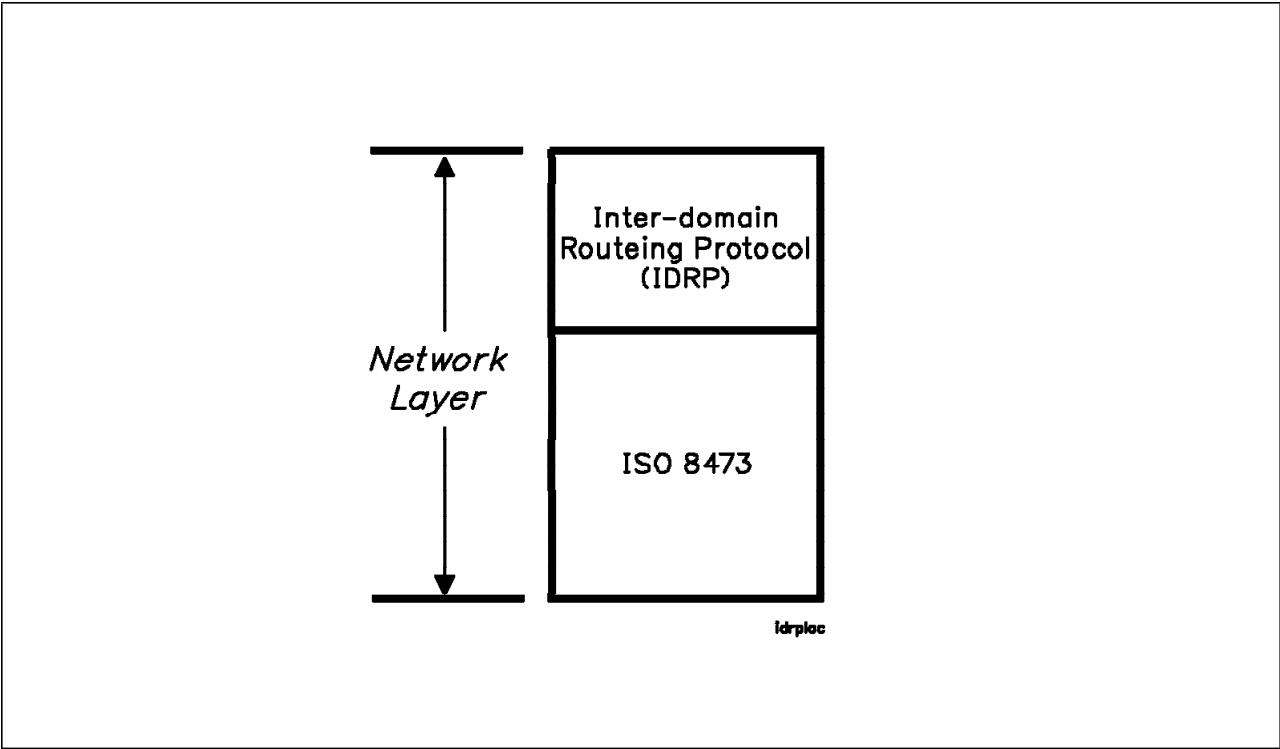


Figure 3. Position of IDR within Network Layer

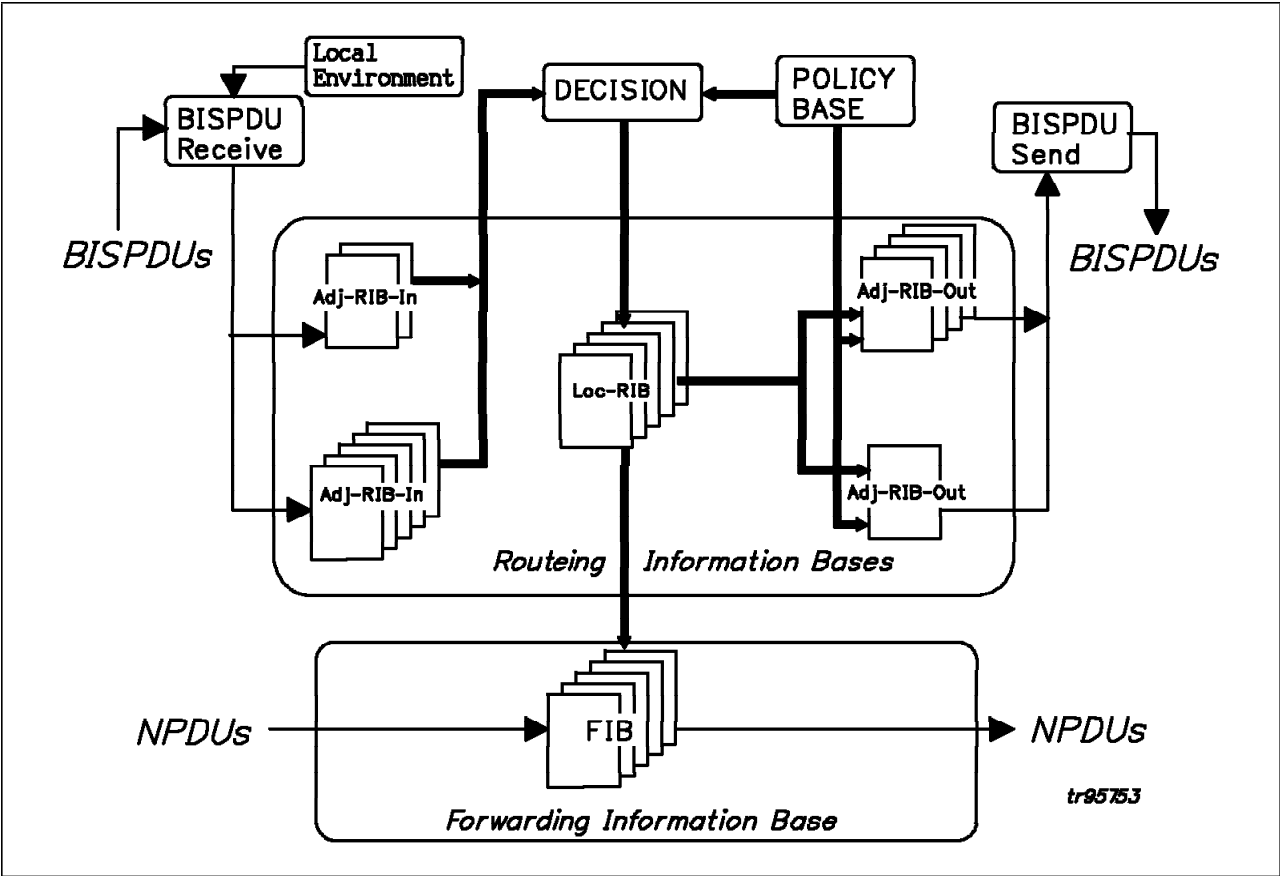


Figure 4. Inter-domain Routeing Components

6.2 Routeing Policy

The direct exchange of policy information is outside the scope of IDRP. Instead, IDRP communicates policy information indirectly in its UPDATE PDUs which reflect the effects of the local policies of RDs on the path to the destination. Since all BISs within a routeing domain must enforce consistent active routeing policies, IDRP provides methods for detecting the existence of active inconsistent policies within a routeing domain. However, the semantics of routeing policies and the methods for establishing them are outside the scope of this standard.⁵

Each routeing domain chooses its routeing policies independently, and insures that all its BISs calculate inter-domain paths which satisfy those policies. Local routeing policies are applied to information in the Routeing Information Base (RIB) to determine a degree of preference⁶ for potential paths. From those paths which are not rejected by the routeing policy, a BIS selects the paths which it will use locally; from the locally selected paths, the BIS will then select the paths that it will advertise externally.

To enforce routeing policies and to insure that policies are both feasible and consistent, this protocol:

- carries path information, expressed in terms of Routeing Domain Identifiers (RDIs) and various path attributes, in its UPDATE PDUs
- permits a routeing domain to selectively propagate its reachability information to a limited set of other routeing domains
- provides a method to detect policy inconsistencies within the set of BISs located in a single routeing domain.
- permits each routeing domain to set its policies individually: that is, global coordination of policy is not required.

6.3 Types of Systems

An Intermediate system that implements the protocol described in this international standard is called a Boundary Intermediate system (BIS). Each BIS resides in a single routeing domain, and may optionally act simultaneously as a BIS and as an intra-domain IS within its own routeing domain.⁷

6.4 Types of Routeing Domains

The protocol described in this International Standard recognizes two types of routeing domains, end routeing domains and transit routeing domains; each of them may contain both ISs and ESs.

6.5 Routeing Domain Confederations

IDRP provides support for Routeing Domain Confederations (RDCs); this optional function permits groups of routeing domains to be organized in a hierarchical fashion.

- An RDC is formed by means outside the scope of this protocol, and composed of a set of *confederation members*. Confederation members (CMs) are either individual routeing domains or routeing domain confederations. Thus, the definition of an RDC is recursive: a confederation member may be a single routeing domain or another confederation.

6.6 Routes: Advertisement and Storage

For purposes of this protocol, a *route* is defined as a unit of information that pairs a destination with the attributes of a path to that destination:

- *Routes* are advertised between a pair of BISs in UPDATE PDUs: the *destination* is the systems whose NSAPs are reported in the NLRI field, and the *path* is the information reported in the path attributes fields of the same UPDATE PDU.
- *Routes* are stored in the Routeing Information Bases: namely, the Adj-RIBs-In, the Loc-RIBs, and the Adj-RIBs-Out. Routes that will be advertised to other BISs must be present in the Adj-RIBs-Out; routes that will be used by the local BIS must be present in the Loc-RIBs, and the next hop for each of these routes must present in the local BIS's Forwarding Information Bases; and routes that are received from other BISs are present in the Adj-RIBs-In.
- A *Route-ID* is assigned to each route that is advertised by a BIS. This identifier is unambiguous in the context of the BIS-BIS connection between the advertising BIS and the receiving BIS.

A BIS can support multiple routes to the same destination by maintaining multiple RIBs and the corresponding multiple FIBs. Each Loc-RIB will be

⁵ Appendix G provides an illustration of a policy description method and its associated semantics as one example of how policies might be expressed.

⁶ *Degree of preference* is discussed in clause 8.15.

⁷ For example, a single system may simultaneously play the roles of a BIS for Inter-domain routeing and a level-2 IS for Intra-domain routeing as described in ISO/IEC 10589.

identified by a different RIB-Att (see clauses 6.7 and 6.8), and can contain only one route to a particular destination.

If the BIS chooses to advertise the route, it may add to or modify the path attributes of the route before advertising it to adjacent BISs. For example, it is possible under certain circumstances to aggregate path attributes, NLRI, or entire routes, as described more fully in 8.17.2.

IDRP also provides mechanisms by which a BIS can inform its neighbor that a previously advertised route is no longer available for use. There are three methods by which a given BIS can indicate that a route has been withdrawn from service:

- a) the Route-ID for a previously advertised route can be advertised in the WITHDRAWN ROUTES field of an UPDATE PDU, thus marking the associated route as being no longer available for use
- b) a replacement route (with the same distinguishing attributes and NLRI) can be advertised, or
- c) the BIS-BIS connection can be closed, which implicitly removes from service all routes which the pair of BISs had advertised to each other.

6.7 Distinguishing Path Attributes and RIB-Atts

Certain path attributes are classified as *Distinguishing Attributes*. Each distinct combination of such attributes identifies a particular information base which will be used to store information about the route. Each combination of distinguishing attributes is called a *RIB-Att* (RIB attribute); the RIB-Att is a common identifier for the Adj-RIB-In, Loc-RIB, Adj-RIB-Out, and FIB with which the route information is associated.

The number of RIB-Atts is limited by the number of distinct sets of permissible distinguishing attributes (see 8.11.2); this in turn limits the number of RIBs and FIBs that a BIS can support. The number of RIBs and FIBs can be further constrained by local decisions—a BIS may choose to support only a limited number of distinct routing information bases (that is, a limited number of RIB-Atts, as described in clause 8.10.1).

6.8 Selecting the Information Bases

Each RIB is identified by a RIB-Att (RIB attribute), and the same RIB-Att also uniquely identifies the associated FIB.

When a BIS receives an UPDATE PDU, it must unambiguously determine the Adj-RIB-In which will be used to store the routing information. The local system examines the path attributes in an UPDATE PDU: the set of distinguishing path attributes that are present

in the UPDATE PDU map into a RIB-Att, and in turn, the RIB-Att identifies the routing information base.

When a BIS receives an NPDU, it must unambiguously determine the FIB that should be used for forwarding this NPDU. This is accomplished by mapping certain fields in the header of the NPDU into a RIB-Att, which then unambiguously identifies a particular FIB (see clauses 9.2 and 9.3).

- A general summary of IDRP's information bases is depicted in Table 1.

6.9 Routing Information Exchange

This International Standard provides several rules governing the distribution and exchange of routing information:

- rules for distributing routing information internally (to BISs within a routing domain)
- rules for distributing routing information externally (to BISs in adjacent routing domains)

Routing information is carried in the protocol's BISPDU's, which are generated on an event-driven basis whenever a BIS receives information which causes it advertise new paths.

6.9.1 Internal Neighbor BIS

Each BIS establishes and maintains communications with all other BISs located in its routing domain. The identity of all BISs within a routing domain is contained in managed object **INTERNAL_BIS** described in clause 8.3.

6.9.2 External Neighbor BIS

Each BIS may establish and maintain communications with other BISs in adjacent routing domains. A BIS has no direct communications link with any BIS in another routing domain unless that RD is adjacent to it, as defined in clause 4.6: that is, a BIS does not communicate directly with a BIS located in a different routing domain unless the pair of BISs are attached to at least one common subnetwork. The identity of neighbor BISs in adjacent routing domains is contained in managed object **EXTERNAL-BIS-NEIGHBORS** described in clause 8.3.

- NOTE:** In the absence of an implementation specific method for ascertaining that a neighbor BIS listed in managed object **EXTERNAL-BIS-NEIGHBORS** is located on a common subnetwork with itself, a local BIS can include the ISO 8473 Complete Route Record parameter so that the recipient of the BISPDU can determine whether the sending BIS is located on the same subnetwork as itself.

1	Table 1. The IDRP Information Bases. The indexing variables and contents of the RIBs and FIBs are shown.		
1	Information Base	Indexed by...	Contains...
1	<i>Adj-RIB-In</i>	— NET of adjacent BIS — RIB-Atts — Route-ID	— Path attributes — NLRI
1	<i>Loc-RIB</i>	— RIB-Atts	— Path attributes — NLRI
1	<i>Adj-RIB-Out</i>	— NET of adjacent BIS — RIB-Atts — Route-ID	— Path attributes — NLRI
1	<i>FIB</i>	— RIB-Atts — NLRI	— NET of next hop BIS — Output SNPA of local BIS — Input SNPA of next hop BIS
1	NOTES:		
1	a) As a local option, a BIS may elect to apply information reduction techniques to path attributes and NLRI information.		
1	b) For each adjacent BIS, a given BIS maintains an Adj-RIB-In for each RIB-Att (including the Empty RIB-Att) that it supports.		
1	c) A BIS maintains a separate Loc-RIB for each RIB-Att (including the Empty RIB-Att) that it supports.		
1	d) For each adjacent BIS, a given BIS maintains an Adj-RIB-Out for each set of RIB-Atts (including the Empty RIB-Att) that it advertises to that neighbor.		
1	e) A given BIS maintains a separate FIB for each set of RIB-Atts (including the Empty RIB-Att) that it has advertised to its neighbor BISs—that is, each FIB corresponds to an Adj-RIB-Out.		
1	To facilitate the forwarding process, a BIS can organize each of its FIBs into two conceptual parts: one containing information for NLRI located within its own RD, and another for NLRI located in other RDs (see 9). For external NLRI, a BIS can further organize the FIB information based on whether the next-hop-BIS is located within its own RD or in another RD (see 9.4, items "a" and "b"). And finally, for those next-hop BISs located in its own RD, the local BIS can organize the information according to a specific forwarding mechanism (see 9.4, items "b1", "b2", and "b3").		

6.10 Routeing Domain Identifiers

Each routing domain (RD) and routing domain confederations (RDC) whose BIS(s) implement the protocol described in this International Standard shall have an unambiguous routeing domain identifier (RDI), which is a generic Network entity title, as described in ISO 7498.

An RDI is assigned statically in accordance with ISO 8348/Add.2, and does not change based on the operational status of the routeing domain. An RDI identifies a routeing domain or a confederation uniquely, but does not necessarily convey any information about its policies or the identities of its members.

6.11 Formats of RDIs, NETs, and NSAP Addresses

Within routeing domains whose BISs implement the protocol defined in this International Standard, RDIs, NETs and NSAP addresses shall be structured as described in clause 8.1.

All Boundary Intermediate systems shall be able to generate and forward PDUs containing NSAP addresses or NETs whose abstract syntax is as described in ISO 8348/AD2.

6.12 Design Objectives

The protocol described in this international standard was developed with a view towards satisfying certain design goals, while others specifically were not addressed by the mechanisms of this protocol.

6.12.1 Within the Scope of the Protocol

This International Standard supports the following design requirements:

Control Transit through an RD: It provides mechanisms to permit a given routeing domain to control the ability of NPDU's from other routeing domains to transit through itself.

Network Layer Protocol Compatibility: It does not require that any changes be made to the following existing Network layer protocols: ISO 8473, ISO 9542, ISO 10030, ISO 10589, and IS 8208.

Autonomous Operation: It provides stable operation in the global OSIE where significant sections of the interworking environment will be controlled by disjoint entities.

Distributed Information Bases: It does not require a centralized global repository for either routeing information or policy information.

QOS-based Routeing: It provides the ability to select routes based on the QOS characteristics of the NPDUs.

Deliverability: It accepts and delivers NPDUs addressed to reachable routeing domains and rejects NPDUs addressed to routeing domains known to be unreachable.

Adaptability: It adapts to topological changes between routeing domains, but not to traffic changes.

Promptness: It provides a period of adaptation to topological changes between domains that is a reasonable function of the maximum logical distance between any pair of routeing domains participating in an instance of this protocol.

Efficiency: It is efficient in the use of both processing resources and memory resources; it does not create excessive routeing traffic overhead.

Robustness: It recovers from transient errors such as lost or temporarily incorrect routeing PDUs, and it tolerates imprecise parameter settings.

Stability: It stabilizes in finite time to "good routes", as long as there are no continuous topological changes or corruptions of the routeing and policy information bases.

Heterogeneity: It is designed to operate correctly over a set of routeing domains that may employ diverse intra-domain routeing protocols. It is capable of running over a wide variety of subnetworks, including but not limited to: ISO 8208 and X.25 subnetworks, PSTN networks, and the OSI Data Link Service.

Availability: It will not result in inability to calculate acceptable inter-domain paths when a single point of failure happens for a pairing of topology and policy that have a *cut set* greater than one.

Fire walls: It will provide fire walls so that:

- Problems within one routeing domain will not affect intra-domain routeing in any other routeing domain
- Problems within one routeing domain will not affect inter-domain routeing, unless they occur on internal inter-domain Links
- Inter-domain routeing will not adversely affect intra-domain routeing.

Scaling: It imposes no upper limit on the number of routeing domains that can participate in a single instance of this protocol.

Multiple Routeing Administrations: It will accommodate inter-domain route calculations without regard to whether or not the participating routeing domains are under control of one or several administrative authorities.

6.12.2 Outside the Scope of the Protocol

The following requirements are not within the design scope of this protocol:

User Data Security: The security of user data carried within an NPDU is outside the scope of this protocol. This protocol is not designed to serve as a substitute for conventional data security practices. However, it can provide a security-related facility to the extent that route computation can be based upon the contents of the ISO 8473 security parameter.

Traffic Adaptation: It does not automatically modify routes based on the traffic load.

Guaranteed delivery: It does not guarantee delivery of all offered NPDUs.

Repair of Partitioned Routeing Domains: It does not use paths external to a partitioned routeing domain to repair the partition.

Repair of Partitioned Routeing Domains Confederations: It does not use paths external to a partitioned routeing domain confederation to repair the partition.

Shared Use of Inter-domain Links: It will not permit an external inter-domain link to carry intradomain traffic.

Suppression of Transient Loops: Although it provides mechanisms to detect and suppress looping of routeing information, it provides no mechanisms to detect or suppress transient looping of ISO 8473 NPDUs.

7. Structure of BISPDUs

In this document, the term *BISPDU* (Boundary IS PDU) is used as a general term to refer to any of the PDUs defined in this clause.

Octets in a PDU are numbered starting from 1, in increasing order. Bits in an octet are numbered from 1 to 8, where bit 1 is the low-order bit and is pictured on the right. When consecutive octets are used to represent a number, the lower octet number has the most significant value. The more significant semi-octet of each pair of semi-octets in a given octet is encoded in the high order bit positions (8 through 5).

- 2 Values are given in decimal, and all numeric fields
2 are unsigned, unless otherwise noted.

The types of PDUs used by this protocol are:

- OPEN PDU
- UPDATE PDU
- IDRP ERROR PDU
- KEEPALIVE PDU
- CEASE PDU
- RIB REFRESH PDU

7.1 Header of BISPDU

Each BISPDU has a fixed size header. There may or may not be a data portion following the header, depending on the PDU type.

The layout of the header fields and their meanings are shown below:

	Inter-domain Routeing Protocol Identifier (1 octet)
	BISPDU Length (2 octets)
	BISPDU Type (1 octet)
	Sequence (4 octets)
	Acknowledgement (4 octets)
2	Credit Offered (1 octet)
2	Credit Available (1 octet)
	Validation Pattern (16 octets)

The meaning and use of these fields are as follows:

Inter-domain Routeing Protocol Identifier: An architectural constant for use in identifying the protocol by the methods of ISO DTR 9577.

BISPDU Length: The BISPDU Length field is a 2 octet unsigned integer. It contains the total length in octets of this BISPDU, including both header and data portions. The value of the BISPDU Length field shall be at least **MinBISPDULength** octets, and no greater than the value carried in the **Maximum_PDU_Size** field of the OPEN PDU received from the remote BIS.

Further, depending on the PDU type, there may be other constraints on the value of the Length field; for example, a KEEPALIVE PDU must have a length of exactly 30 octets. No padding after the end of BISPDU is allowed, so the value of the Length field must be the smallest value required given the rest of the BISPDU.

BISPDU Type: The BISPDU Type field contains a one octet type code which identifies the specific type of the PDU. The supported type codes are:

BISPDU Type	Code
OPEN PDU	1

BISPDU Type	Code
UPDATE PDU	2
IDRP ERROR PDU	3
KEEPALIVE PDU	4
CEASE PDU	6
RIB REFRESH PDU	7

All other BISPDU type codes are reserved for future extensions.

Sequence: The Sequence field contains a 4 octet unsigned integer that is the sequence number of this PDU. The procedures for generating sequence numbers for the various types of BISPDU are described in clause 8.5.2.

Acknowledgement: The Acknowledgment field is a 4 octet unsigned integer that contains the sequence number of the PDU that the sender last received correctly and in sequence number order.

2 **Credit Offered:** The contents of this field indicate the
2 number of additional BISPDU's that the local BIS is
2 willing to accept from the remote BIS; it is used by
2 the flow control process described in 8.5.3.

2 **Credit Available:** The contents of this field indicate the
2 number of additional BISPDU's that the local BIS is
2 able to send to the remote BIS; it is used by the
2 flow control process described in 8.5.3.

Validation Pattern: This 16-octet field is used to provide a validation function for the BISPDU. Depending upon the contents of the field "Authentication Code" of the OPEN PDU, this field can be used to a checksum on the content of the BISPDU (see 8.8 and Appendix B), or a checksum with authentication of the source of the BISPDU (see 8.9). The authentication function is achieved by encryption of the basic checksum.

7.2 OPEN PDU

The OPEN PDU is used by a BIS for starting a BIS-BIS connection. The first PDU sent by either side is an OPEN PDU.

The OPEN PDU contains a fixed header and the additional fields shown below:

Fixed Header
Version (1 octet)
Hold Time (2 octets)
Maximum PDU Size (2 octets)
Source RDI Length Indicator (1 octet)
Source RDI (variable)
RIB-AttsSet (variable)
Confed-IDs (variable)

Authentication Code (1 octet)
Authentication Data (variable)

The meaning and use of these fields are as follows:

- 2 **Version:** This one octet field contains the version
2 number of the protocol. Its value is currently 1.

Hold Time: This field contains a 2 octet unsigned integer that is the maximum number of seconds that may elapse between the receipt of two successive BISPDU's of any of the following types: KEEPALIVE, UPDATE, or RIB REFRESH PDU's.

- 1 **Maximum PDU Size:** This field contains a 2 octet
1 unsigned integer that is the maximum number of
1 octets that this BIS will accept in an incoming
1 UPDATE PDU, IDRP ERROR PDU, or RIB REFRESH
1 PDU.

- 1 Independent of this value, every BIS shall accept
1 KEEPALIVE PDUs or CEASE PDUs of length 30
1 octets; every BIS shall also accept any OPEN PDU
1 with length less than or equal to 3000 octets.

- 1 As a minimum, every BIS is required to support
1 reception of all BISPDUs whose size is greater than
1 or equal to **MinBISPDULength** octets and less than
1 or equal to 1024 octets: that is, the minimum
1 acceptable value for this field is 1024.

Source RDI Length Indicator: This one octet field contains the length in octets of the Source RDI field.

Source RDI: This variable length field contains the RDI of the routing domain in which the BIS that is sending this BISPDU is located.

RIB-AttsSet: This variable length field contains a list of all *RIB-Atts* that the local BIS is willing to support when communicating with the remote BIS (that is, the one that receives this OPEN PDU). That is, it contains an encoding of all or part of the information contained in managed object **RIB-AttsSet** . (See clauses 8.3 and 8.10.1.)

A BIS is not required to list all of its supported RIB-Atts in an OPEN PDU that is sent to a neighbor BIS located in an adjacent routing domain. It must include only those RIB-Atts that correspond to Adj-RIBs-Out that the local BIS will use to communicate with the neighbor BIS, and those that correspond to the RIB-Atts of the Adj-RIBs-In that the local BIS supports for storing UPDATE PDUs received from that neighbor BIS.

However, a BIS shall include all of the RIB-Atts listed in managed object **RIB-AttsSet** in an OPEN PDU that is sent to another BIS located in its own routing domain. Failure to do so will result in an OPEN PDU error, as described in clause 8.20.2.

The encoding of this field is as follows:

Number of Distinguishing Attributes in First Set
First attribute, first set
....
Last Attribute, first set
Number of Distinguishing Attributes in Second Set
First attribute, second set
....
Last Attribute, second set
...
Number of Distinguishing Attributes in Nth Set
First attribute, Nth set
....
Last Attribute, Nth set
...
Number of Distinguishing Attributes in Last Set
First attribute, last set
....
Last Attribute, last set

The field *Number of RIB-Atts* is one octet long. It contains the total number of non-empty RIB-Atts supported by this BIS. Since every BIS supports an empty RIB-Att (see clause 8.10.1.), the empty RIB-Att shall not be listed in the OPEN PDU.

If a BIS supports no RIB-Atts other than the empty RIB-Att, then the field *Number of Non-empty RIB-Atts* shall contain 0.

If the *Number of Non-Empty RIB-Atts* is non-zero, then the BIS supports all of the listed RIB-Atts plus the empty RIB-Att.

Each field *Number of Distinguishing Attributes in the Nth Set* is one octet long, and it contains the number of distinguishing attributes that are contained in the *Nth* RIB-Att.

Each of the individual attributes in a set is encoded as follows:

- a type specific distinguishing attribute (see clause 8.11.2) is encoded as a single octet, using the type code specified in clause 7.3 for the corresponding UPDATE PDU path attribute.
- a type-value-specific distinguishing attribute (see clause 8.11.2) is encoded as a triple <type, length, value>, using the type code, length, and the first four fields of the value field, as specified in clause 7.3 for the corresponding UPDATE PDU path attribute.

NOTE: The first four fields of the source- and destination-specific attributes list the NSAP address to which the attribute applies and the

Number of Non-empty RIB-Atts

value of the attribute. Although additional fields may be present (for example, to list metrics), they are not carried in the OPEN PDU.

Confed-IDs: This is a variable length field which reports the RDIs of all RDCs that this BIS is a member of. The encoding of this field is as follows:

Number of RDCs (1 octet)
Length of First RDI (1 octet)
RDI of first RDC
....
Length of Last RDI (1 octet)
RDI of last confederation

The 1 octet field *Number of RDCs* gives the number of RDCs of which this BIS is a member. A value of zero indicates that this BIS participates in no RDCs.

For each such confederation, the following fields give the length and RDI for each confederation, where each RDI is encoded as a single prefix, according to clause 8.1.2.1.

Authentication Code: This 1-octet unsigned integer indicates the authentication mechanism being used:

- a) Code 1 indicates that the Validation Pattern field in the header of each BISPDU contains an unencrypted checksum covering the contents of that BISPDU. Its use is described in clause 8.5.1.
- b) Code 2 indicates that the Validation Pattern field in the header of each BISPDU contains an encrypted checksum covering the contents of that BISPDU. Encryption of the checksum provides the authentication function for the source of the BISPDU. Its use is described in clause 8.9.

Authentication Data: The form and meaning of this field is a variable-length field that depends on the Authentication Code, as described in clause 8.9. The length of the authentication data field can be determined from the Length field of the BISPDU header.

7.3 UPDATE PDU

2 An UPDATE PDU is used to advertise a single feasible
2 route to a neighbor BIS, or to withdraw multiple
2 unfeasible routes from service (see 6.6). An UPDATE
2 PDU may simultaneously advertise a feasible route
2 and withdraw multiple unfeasible routes from service.
2 The UPDATE PDU always includes the fixed header,
2 and can optionally contain the other fields shown in
2 Figure 5:

2 — If routes are being explicitly withdrawn from
2 service, then the UNFEASIBLE ROUTE COUNT

2 and (for a non-zero count) the WITHDRAWN
2 ROUTES fields will be present

2 — if feasible route is being advertised, then the
2 PATH ATTRIBUTES and NLRI fields will be
2 present.

2 An UPDATE PDU can advertise at most one route,
2 which may be described by several path attributes,
2 each of which is encoded as a 4-tuple. All path attri-
2 butes contained in a given UPDATE PDU apply to the
2 destinations carried in the Network Layer Reachability
2 Information field of the UPDATE PDU.

2 An UPDATE PDU can list multiple routes to be with-
2 drawn from service. Each such route is identified by
2 its Route-ID, which unambiguously identifies the route
2 in the context of the BIS-BIS connection in which it
2 had been previously been advertised.

2 An UPDATE PDU may advertise only routes to be
2 withdrawn from service, in which case it will not
2 include Path Attributes or NLRI. Conversely, it may
2 advertise only a feasible route, in which case the
2 UNFEASIBLE ROUTE COUNT and WITHDRAWN
2 ROUTES need not be present.

2 The components of the UPDATE PDU are described
2 below:

Fixed Header
Unfeasible Route Count (1 octet)
Withdrawn Routes (variable)
Total Path Attributes Length (2 octets)
Path Attributes (variable)
Network Layer Reachability Information (variable)

The use of these fields is as follows:

2 **Unfeasible Route Count** This is a 1-octet long field
2 that contains an integer whose value is equal to the
2 number of ROUTE-IDs that are included in the sub-
2 sequent WITHDRAWN ROUTES field. A value of 0
2 indicates that no routes are being withdrawn from
2 service, and that the WITHDRAWN ROUTES field is
2 not present in this UPDATE PDU.

2 **Withdrawn Routes:** This is a variable length field that
2 contains a list of Route-IDs for the routes that are
2 being withdrawn from service. Each Route-ID is 4
2 octets in length.

Total Path Attribute Length: The length of this field is
2 octets. It is the total length of all Path Attributes
2 in the UPDATE PDU in octets.

2 **Path Attributes:** A variable length sequence of path
2 attributes is present in every UPDATE PDU that is
2 used to advertise a feasible route. Each path attri-
2 bute is a 4-tuple of variable length—<flag, type,
2 length, value>. The elements are used as follows:

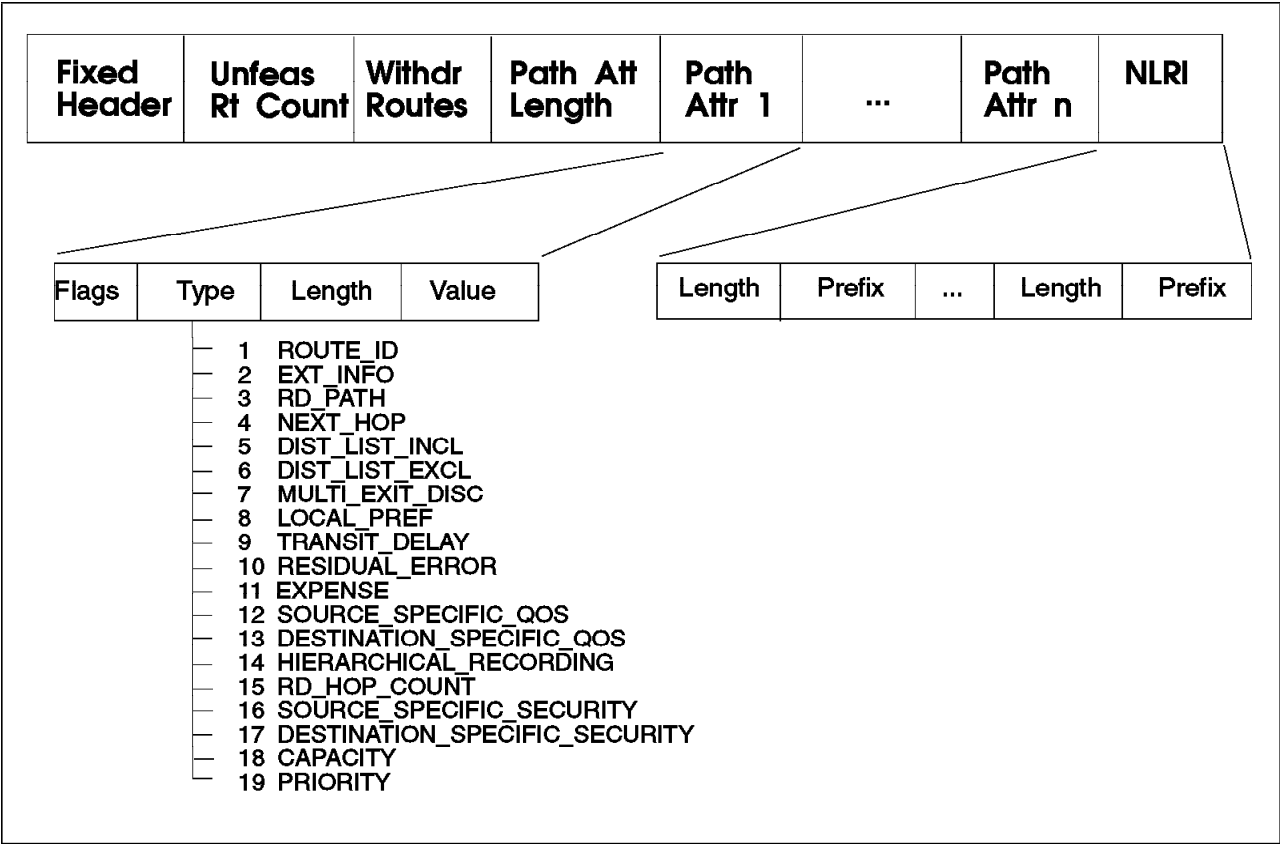


Figure 5. Structure of the UPDATE PDU

— Flag:

The first element of each attribute is a one octet field:

- The high-order bit (bit 8) of the attribute flags octet is the Optional bit. If it is set to 1, the attribute is optional; if set to 0, the attribute is well-known.
- The second high-order bit (bit 7) of the attribute flags octet is the Transitive bit. It defines whether an optional attribute is transitive (if set to 1) or non-transitive (if set to 0). For well-known attributes, the Transitive bit shall be set to 1.
- The third high-order bit (bit 6) of the attribute flags octet is the Partial bit. It defines whether the information contained in the optional transitive attribute is partial (if set to 1) or complete (if set to 0). For well-known attributes and for optional non-transitive attributes the Partial bit shall be set to 0.
- The lower order five bits (1 through 5) of the attribute flags octet are reserved. They shall be transmitted as 0 and shall be ignored when received.

— Type:

The second element of each attribute is a one octet field which contains the type code for the attribute. Currently defined attribute type codes are discussed in clause 8.11.

— Length:

The third field of each path attribute is 2 octets in length; it contains the length in octets of the immediately following Value field.

— Value:

The remaining octets of each path attributes field contain the values of the attributes, which are interpreted according to the attribute flags and the attribute type code. The supported attribute values and their uses are the following:

- a) ROUTE-ID (Type Code 1):
ROUTE-ID is a well-known mandatory attribute whose length is 4 octets. It contains the route identifier for the route being advertised in this UPDATE PDU. The ROUTE-ID is assigned by the BIS that is advertising the route, and is unambiguous in the context of the current BIS-BIS connection. This attribute is non-transitive: that is, it is not redistributed by a BIS that receives it. Details are given in 8.12.1.
- b) EXT_INFO (Type Code 2):

EXT_INFO is a well-known discretionary attribute that has a length of zero octets; its presence indicates that some (or all) of the path attributes or Network Layer Reachability Information contained in this UPDATE PDU have been obtained by methods not specified by IDRP. Conversely, its absence indicates that all path attributes and NLRI have been learned by methods defined within IDRP. Its usage is defined in clause 8.12.2

c) RD_PATH (Type Code 3):

RD_PATH is a well-known mandatory attribute that is composed of a series of RD path segments. Each RD path segment is represented by a triple <path segment type, path segment length, path segment value>.

The path segment type is a 1-octet long field, with the following values defined:

Value	Segment Type
1	RD_SET: unordered set of RDIs that the UPDATE PDU has traversed
2	RD_SEQUENCE: ordered set of RDIs the UPDATE PDU has traversed

The path segment length is a two octet field containing the length in octets of the path segment value field.

1 The path segment value field contains one
1 or more 2-tuples <length, RDI>: *length* is
1 a one octet field that contains a positive
1 integer that gives the length in bits of the
1 following *RDI* field, and the *RDI* field itself
1 is encoded according to clause 8.1.2.1.

Usage of this attribute is defined in clause 8.12.3.

d) NEXT_HOP (Type Code 4):

This is a well-known discretionary attribute that can be used for two principal purposes:

- 1) to permit a BIS to advertise a different BIS's NET in the "NET of Next Hop" field
- 2) to allow a given BIS to report some or all of the SNPAs that exist within the local system

It is encoded as shown below:

IDRP_Server_Allowed (1 octet)
Length of NET (1 octet)

NET of Next Hop (variable)
Number of SNPAs (1 octet)
Length of 1st SNPA(1 octet)
First SNPA (variable)
Length of 2nd SNPA (1 octet)
Second SNPA (variable)
...
Length of Last SNPA (1 octet)
Last SNPA (variable)

The use and meaning of these fields are as follows:

IDRP_Server_Allowed: The value X'FF' indicates the recipient of this UPDATE PDU has the option of advertising (in its own outbound UPDATE PDUs) the NET and SNPA information learned from this UPDATE PDU. If the value is not X'FF', then the recipient of this UPDATE PDU shall not advertise the NET and SNPA information learned from this UPDATE PDU.

Length of NET: A 1 octet field whose value expresses the length of the "NET of Next Hop" field as measured in octets

NET of Next Hop: A variable length field that contains the NET of the next BIS on the path to the destination system

Number of SNPAs: A 1 octet field which contains the number of distinct SNPAs to be listed in the following fields. The value 0 may be used to indicate that no SNPAs are listed in this attribute.

Length of Nth SNPA: A 1 octet field whose value expresses the length of the "Nth SNPA of Next Hop" field as measured in semi-octets

Nth SNPA of Next Hop: A variable length field that contains an SNPA of the BIS whose NET is contained in the "NET of Next Hop" field. The field length is an integral number of octets in length, namely the rounded-up integer value of one half the SNPA length expressed in semi-octets; if the SNPA has an contains an odd number of semi-octets, a value in this field will be padded with a trailing all-zero semi-octet.

Its usage is defined in clause 8.12.4.

e) DIST_LIST_INCL (Type Code 5):

The DIST_LIST_INCL is a well-known discretionary attribute that contains a list of the RDIs of routeing domains and confed-

erations to which the Network Layer
Reachability Information contained in that
UPDATE PDU may be distributed. Its
usage is described in clause 8.12.5.

Each RDI in the list is encoded as a
<length, value> pair, using the following
fields:

Count (1 octet)
Length (1 octet)
Value (variable)
...
Length (1 octet)
Value (variable)

The use and meaning of these fields are as
follows:

1) Count:

The count field is an integer that gives
the number of RDIs in the list, where
each RDI is represented by a <length,
value> pair as described below.

2) Length:

The length field indicates the length in
octets of the following RDI.

3) Value:

The value field contains the RDI,
encoded according to clause 8.1.2.1.

The DIST_LIST_INCL attribute shall not be
present together with the DIST_LIST_EXCL
attribute in the same UPDATE PDU.

f) DIST_LIST_EXCL (Type Code 6):

The DIST_LIST_EXCL is a well-know discre-
tionary attribute that contains a list of the
RDIs of routeing domains and confeder-
ations to which the Network Layer
Reachability Information contained in that
UPDATE PDU shall not be distributed. Its
usage is described in clause 8.12.6.

Each RDI in the list is encoded as a
<length, value> pair, using the following
fields:

Count (1 octet)
Length (1 octet)
Value (variable)
...
Length (1 octet)
Value (variable)

The use and meaning of these fields are as
follows:

1) Count:

The count field is an integer that gives
the number of RDIs in the list, where
each RDI is represented by a <length,
value> pair as described below.

2) Length:

The length field indicates the length in
octets of the following RDI.

3) Value:

The value field contains the RDI,
encoded according to clause 8.1.2.1.

The DIST_LIST_EXCL attribute shall not be
present together with the DIST_LIST_INCL
attribute in the same UPDATE PDU.

g) MULTI-EXIT_DISC (Type Code 7):

MULTI-EXIT_DISC (Multi-exit Discriminator)
is an optional non-transitive attribute that
is a 1 octet non-negative integer. The
value of this attribute may be used by a
BIS's decision process to discriminate
between multiple exit points to an adjacent
routeing domain. Its usage is defined in
clause 8.12.7.

h) LOCAL_PREF (Type Code 8):

LOCAL_PREF is a well-known discretionary
attribute that is 1 octet in length. It is
used by a BIS to inform other BISs in its
own RD of the originating BIS's degree of
preference for an advertised path. Usage
of this attribute is described in clauses
8.12.8 and 8.14.2.

i) TRANSIT_DELAY (Type Code 9)

The TRANSIT_DELAY is a well-known dis-
cretionary attribute that has a length of 2
octets, and is used to notify a BIS that the
path to destination has transit delay deter-
mined by the value of the attribute. Its
usage is defined in 8.12.9.

j) RESIDUAL_ERROR (Type Code 10)

The RESIDUAL_ERROR is a well-know dis-
cretionary attribute that has a length of 4
octets, and is used to notify a BIS that the
path to destination has residual error prob-
ability determined by the value of the attri-
bute. Its usage is defined in clause
8.12.10.

k) EXPENSE (Type Code 11)

The EXPENSE is a well-known discretionary
attribute that has a length of 2 octets, and
is used to notify a BIS that the path to des-
tination has expense determined by the

value of the attribute. Its usage is defined in clause 8.12.11.

l) SOURCE SPECIFIC QOS (Type Code 12)

This is a well-known discretionary attribute whose variable length field contains the parameters associated with ISO 8473's source address specific QOS parameters, and is encoded as follows:

NSAP prefix length (1 octet)
NSAP prefix (variable)
QOS length (1 octet)
QOS value (variable)
Metric length (1 octet)
Metric value (variable)

The use and meaning of the fields is as follows:

1) NSAP prefix length:

The length field indicates the length in bits of the NSAP address prefix (see 8.1.2.1). A length of zero indicates a prefix that matches all NSAPs.

2) NSAP prefix:

If the Length field specifies an integral number of octets, then the Prefix field shall contain only the NSAP address prefix encoded according to 8.1.2.1. If the Length field does not specify an integral number of octets, then the Prefix field shall contain the NSAP address prefix encoded according to 8.1.2.1, followed by enough trailing zeroes to make the end of the field fall on an octet boundary. If an ISO 8473 NPDU's source NSAP address matches the NSAP prefix, then it will be routed according to the indicated QOS value.

3) QOS length:

Gives the length in octets of the QOS value field.

4) QOS value:

Contains the value of the QOS parameter.

5) Metric length:

Gives the length in octets of the metric value field. A length of zero is a permitted value.

6) Metric value:

Contains a positive integer that is the value of the metric associated with the

path being advertised (that is, its "cost")

Its usage is defined in clause 8.12.12.

m) DESTINATION SPECIFIC QOS (Type Code 13)

This is a well-known discretionary attribute whose variable length field contains the parameters associated with ISO 8473's destination address specific QOS parameters, and is encoded as follows:

NSAP prefix length (1 octet)
NSAP prefix (variable)
QOS length (1 octet)
QOS value (variable)
Metric length (1 octet)
Metric value (variable)

The use and meaning of the fields is as follows:

1) NSAP prefix length:

The length field indicates the length in bits of the NSAP address prefix (see 8.1.2.1). A length of zero indicates a prefix that matches all NSAPs.

2) NSAP prefix:

If the Length field specifies an integral number of octets, then the Prefix field shall contain only the NSAP address prefix encoded according to 8.1.2.1. If the Length field does not specify an integral number of octets, then the Prefix field shall contain the NSAP address prefix encoded according to 8.1.2.1, followed by enough trailing zeroes to make the end of the field fall on an octet boundary. If an ISO 8473 NPDU's destination NSAP matches the NSAP prefix, then it will be routed according to the indicated QOS value.

3) QOS length:

Gives the length in octets of the QOS value field.

4) QOS value:

Contains the value of the QOS parameter.

5) Metric length:

Gives the length in octets of the metric value field. A length of zero is a permitted value.

6) Metric value:

Contains a positive integer that is the value of the metric associated with the path being advertised (that is, its "cost")

Its usage is defined in clause 8.12.13.

n) HIERARCHICAL RECORDING (Type Code 14)

This is a well-known discretionary attribute. It contains a 1-octet field used by members of a routing domain confederation to control the transitivity of NPDUs through the confederation. It is encoded as follows:

Flag (1 octet)

Its usage is defined in clause 8.12.14.

o) RD_HOP_COUNT (Type Code 15)

The RD_HOP_COUNT is a well-known mandatory attribute that contains a 1 octet long field. It contains an unsigned integer that is the upper bound on the number of routing domains through which this UPDATE PDU has travelled. Its usage is defined in clause 8.12.15.

p) SOURCE SPECIFIC SECURITY (Type Code 16)

This is a well-known discretionary attribute whose variable length field contains the parameters associated with ISO 8473's source address specific security parameter, and is encoded as follows:

NSAP prefix length (1 octet)
NSAP prefix (variable)
Length (1 octet)
Security label (variable)

The use and meaning of the fields is as follows:

1) NSAP prefix length:

The length field indicates the length in bits of the NSAP address prefix (see 8.1.2.1). A length of zero indicates a prefix that matches all NSAPs.

2) NSAP prefix:

If the Length field specifies an integral number of octets, then the Prefix field shall contain only the NSAP address prefix encoded according to 8.1.2.1. If the Length field does not specify an integral number of octets, then the

Prefix field shall contain the NSAP address prefix encoded according to 8.1.2.1, followed by enough trailing zeroes to make the end of the field fall on an octet boundary. If an ISO 8473 NPDU's source NSAP address matches the NSAP prefix, then it will be routed according to the indicated security label.

3) Length:

Gives the length in octets of the Security label field.

4) Security label:

Contains the value of the security label.

Its usage is defined in clause 8.12.16.

q) DESTINATION SPECIFIC SECURITY (Type Code 17)

This is a well-known discretionary attribute whose variable length field contains the parameters associated with ISO 8473's destination address specific security parameter, and is encoded as follows:

NSAP prefix length (1 octet)
NSAP prefix (variable)
Length (1 octet)
Security label (variable)

The use and meaning of the fields is as follows:

1) NSAP prefix length:

The length field indicates the length in bits of the NSAP address prefix (see 8.1.2.1). A length of zero indicates a prefix that matches all NSAPs.

2) NSAP prefix:

If the Length field specifies an integral number of octets, then the Prefix field shall contain only the NSAP address prefix encoded according to 8.1.2.1. If the Length field does not specify an integral number of octets, then the Prefix field shall contain the NSAP address prefix encoded according to 8.1.2.1, followed by enough trailing zeroes to make the end of the field fall on an octet boundary. If an ISO 8473 NPDU's destination NSAP matches the NSAP prefix, then it will be routed according to the indicated security label.

3) Length:

Gives the length in octets of the security label field.

4) Security label:

Contains the value of the security label.

Its usage is defined in clause 8.12.17.

r) CAPACITY (Type code 18)

CAPACITY is a well-known discretionary attribute that has a length of 1 octet, and is used to denote the relative capacity of the RD_PATH for handling traffic. High values indicate a lower traffic handling capacity than do low values. Its usage is defined in clause 8.12.18.

s) PRIORITY (Type Code 19)

The PRIORITY is a well-known discretionary attribute that is 1 octet in length. The content of the field is an integer in the range from 0 to 14. It enables paths to be distinguished on the basis of which values of the ISO 8473 priority parameter (see ISO 8473, clause 7.5.7). As in ISO 8473, the value 0 indicates the normal (default) priority, and the values 1 through 14 indicate higher priorities. A particular value of this parameter, *m*, means that the BIS will not forward any ISO 8473 PDUs whose priority parameter has a value less than *m*. Detailed usage rules are presented in clause 8.12.19.

Network Layer Reachability Information: This variable length field contains a list of NSAP address prefixes. The length in octets of the *Network Layer Reachability Information* is not encoded explicitly, but can be calculated (see Figure 5) as:

$$BISPDU\ Length - 30 - Total\ Path\ Attributes\ Length,$$

where *BISPDU Length* is the value encoded in the Fixed Header, *Total Path Attributes Length* is the value encoded in the variable part of the UPDATE PDU, and 30 octets is the combined length of the *Fixed Header* field and the *Total Path Attributes Length* field.

Reachability information is encoded as one or more 2-tuples of the form <length, prefix>, whose fields are described below.

Length (1 octet)
Prefix (variable)

The use and meaning of these fields are as follows:

a) Length:

- 5 The length field indicates the length in bits of
5 the NSAP address prefix (see 8.1.2.1). A length

- 5 of zero indicates a prefix that matches all
5 NSAPs.

b) Prefix:

- 5 If the Length field specifies an integral number
5 of octets, then the Prefix field shall contain only
5 the NSAP address prefix encoded according to
5 8.1.2.1. If the Length field does not specify an
5 integral number of octets, then the Prefix field
5 shall contain the NSAP address prefix encoded
5 according to 8.1.2.1, followed by enough trailing
5 zeroes to make the end of the field fall on an
5 octet boundary.

7.4 IDRP ERROR PDU

IDRP ERROR PDUs report error conditions which have been detected by the local BIS. In addition to its fixed header, the IDRP ERROR PDU contains the following fields:

Fixed Header
Error Code (1 octet)
Error Subcode (1 octet)
Data (variable)

The use of these fields is as follows:

Error code: The Error code field is 1 octet long. It describes the type of error. The following error codes are defined:

Error Code	Value
OPEN_PDU_Error	1
UPDATE_PDU_Error	2
Hold_Timer_Expired	3
FSM_Error	4
RIB_REFRESH_PDU_Error	5

All errors are fatal to the BIS connection.

- 1 **Error subcode:** The Error subcode is one octet long, and shall be present in every IDRP ERROR PDU. The error subcode provides more specific information about the nature of the reported error. A given IDRP ERROR PDU may report only one error subcode for the indicated error code. The supported error subcodes are as follows:

a) OPEN PDU Error subcodes:

Subcode	Value
Unsupported_Version_Number	1
Bad_Max_PDU_Size	2
Bad_Outstanding_PDUs	3
Bad_Peer_RD	4
Unsupported_Authentication_code	5
Authentication_Failure	6
Bad_RIB-AttsSet	7

Subcode	Value
RDC_Mismatch	8

b) UPDATE PDU Error subcodes:

Subcode	Value
Malformed_Attribute_List	1
Unrecognized_Well-known_Attribute	2
Missing_Well-known_Attribute	3
Attribute_Flags_Error	4
Attribute_Length_Error	5
RD_Routeing_Loop	6
Invalid_NEXT_HOP_Attribute	7
Optional_Attribute_error	8
Invalid_Reachability_Information	9
Misconfigured_RDCs	10

c) Hold_Timer_Expired Error Subcodes:

Subcode	Value
NULL	0

d) FSM_Error Error Subcodes:

When an FSM Error (see 8.6.1) has occurred, the first semi-octet of the error subcode carries the type number of the BISPDU that should not have been received and the second semi-octet encodes the state that the FSM was in when the reception took place. The BISPDU type numbers are defined in 7.1; the FSM states are encoded as follows:

FSM State	Encoded Value
CLOSED	1
OPEN-RECVD	2
OPEN-SENT	3
CLOSE-WAIT	4
ESTABLISHED	5

e) RIB_REFRESH_PDU_Error Subcodes:

Subcode	Value
Invalid_OpCode	1
Unsupported_RIB-Atts	2

7.5 KEEPALIVE PDU

A KEEPALIVE PDU consists of only a PDU header and has a length of 30 octets.

A BIS can use the periodic exchange of KEEPALIVE PDUs with an adjacent BIS to verify that the peer BIS is reachable and active. KEEPALIVE PDUs are exchanged often enough as not to cause the hold time advertised in the OPEN PDU to expire. The maximum time between KEEPALIVE PDUs shall be one third of the Hold Time interval as advertised in the Hold Time field of the OPEN PDU.

A KEEPALIVE PDU may be sent asynchronously to acknowledge the receipt of other BISPDU. Sending a KEEPALIVE PDU does not cause the sender's sequence number to be incremented.

7.6 CEASE PDU

A CEASE PDU consists of only a PDU header and has length of 30 octets.

A CEASE PDU is used by the originating BIS to instruct the peer BIS to close the BIS-BIS connection.

Receipt of a CEASE PDU will cause the BIS to close down the connection with the BIS that issued it, as described in 8.6.2.

7.7 RIB REFRESH PDU

The RIB REFRESH PDU is used to allow a BIS to send a refresh of the routing information in an Adj-RIB-Out to a neighbor BIS, or to solicit a neighbor BIS to send a refresh of its Adj-RIB-Out to the local BIS. The RIB REFRESH PDU contains a fixed header and also the additional fields shown below:

Fixed Header
OpCode (1 octet)
RIB-Atts (variable)

The use and meaning of these fields is as follows:

There are three OpCode values defined:

Code	Meaning
1	RIB Refresh Request
2	RIB Refresh Start
3	RIB Refresh End

The RIB-Atts field contains the RIB-Atts of the Adj-RIB-In for which a refresh is being requested. This field is encoded in the same way that the RIB-AttsSet field of the OPEN PDU is encoded.

Data: This variable length field contains zero or more octets of data to be used in diagnosing the reason for the IDRP ERROR PDU. The contents of the Data field depends upon the error code and error subcode.

Note that the length of the Data field can be determined from the Length field of the BISPDU header. The minimum length of the IDRP ERROR PDU is 30 octets, including BISPDU header.

Its usage is defined in clause 8.10.3.

8. Elements of Procedure

This clause explains the elements of procedure used by the protocol specified in this International Standard; it also describes the naming conventions and system deployment practices assumed by this protocol.

8.1 Naming and Addressing Conventions

Correct operation of this protocol requires that all NSAP-addresses, NETs, and RDIs shall be encoded according to the preferred binary encoding method of ISO 8348/Add.2.

8.1.1 Interpretation of Address Information

IDRP does not assume or require any particular internal structure for the address. That is, as long as the routing domain administrator assigns values within this field that are consistent with the deployment constraints of clause 8.2.2, the protocol specified in this International Standard will operate correctly.

8.1.2 NSAP Address Prefixes

NSAP address prefixes provide a compact method for identifying groups of systems that reside in a given routing domain or confederation. A prefix may have a length that is either smaller than, or the same size as, the base NSAP address.

NOTE: At one extreme, for example, the largest NSAP address prefix will be identical with the full NSAP address—in this case, it would identify a single system rather than a group of systems. At another extreme, the NSAP prefix may be based on only a portion of NSAP address's IDP—in this case, it will identify a large group of systems.

8.1.2.1 Encoding of Prefixes

The encoded form of an NSAP address prefix is obtained by encoding the prefix (expressed in its abstract syntax) according to the preferred binary encoding of ISO 8348/Add.2, unless the end of the prefix falls within the IDP. In this case, each decimal digit in the prefix shall be encoded as the corresponding semi-octet in the range 0000-1001 and no padding characters shall be inserted.

The length of an encoded prefix is denominated in bits. The prefix shall end on a boundary that is legal in the abstract syntax of the address family from

which it is derived. For example, the encoding of a prefix whose DSP is expressed in decimal syntax must end on a semi-octet boundary, while the encoding of a prefix whose DSP is expressed in binary syntax can end on an arbitrary bit boundary.

8.1.2.2 Prefix Matching

As part of the forwarding process, a BIS must match the destination NSAP address from an ISO 8473 NPDU against the NSAP address prefixes that are used to identify the Forwarding Information Bases. An NSAP address prefix that extends into the DSP shall be compared directly against the encoded NSAP address, including any padding characters that may be present. An NSAP address prefix which does not extend into the DSP shall be compared against NSAP', which is obtained from the encoded NSAP address by removing all padding characters that were inserted by the binary encoding process of ISO 8348/Add.2.

The existence of a match shall be determined as follows:

- a) If the encoded NSAP address (or NSAP') contains fewer bits than the NSAP address prefix, then there is no match.
- b) If the encoded NSAP address (or NSAP') contains at least as many bits as the NSAP address prefix, and all bits of the NSAP address prefix are identical to the corresponding leading bits of the encoded NSAP address (or NSAP'), there is a match. Otherwise, there is no match.

In cases where a given NSAP address matches several address prefixes, the match with the longest prefix shall take precedence. For purposes of prefix matching, the length of a prefix is defined to be the number of bits that it contains.

NOTE: Any implementation of a matching process that satisfies the requirements listed above may be used. The key point is that matching process must be aware of whether or not the NSAP address prefix extends into the DSP, and must then either include or exclude padding characters from the encoded NSAP address, as defined above.

8.2 Deployment Guidelines

8.2.1 Minimum Configuration of an RD

To participate in this inter-domain routing protocol, a routing domain must, as a minimum:

- contain at least one Boundary Intermediate system
- contain at least one BIS capable of delivering NPDUs to the intra-domain routing function, if the RD contains End systems.

8.2.2 Deployment of ISs and ESs

End systems and intermediate systems may use any NSAP address or NET that has been assigned under ISO 8348/AD2 guidelines. However, correct and efficient operation of this protocol can only be guaranteed if the following additional guidelines are met:

- 2 a) An NSAP prefix carried in the NLRI field of route
- 2 originated by a BIS in a given routing domain
- 2 should be associated with only that routing domain: that is, no system identified by the prefix should reside in a different routing domain. Ambiguous routing may result if several routing domains originate routes whose NLRI fields contain identical NSAP address prefixes, since this would imply that the same system(s) are simultaneously located in several routing domains. (As noted in clause 8.1.2.2, prefixes may be discriminated by both content and length.)
- b) Several different NSAP prefixes may be associated with a single routing domain identifier (RDI). Thus, it is possible to construct a single routing domain which contains a mix of systems which use NSAP addresses assigned by several different naming authorities.

The protocol defined in this international standard assumes that the guidelines listed above have been satisfied, but it contains no means to verify that this is so. Therefore, such verification will be the responsibility of the administrators of routing domains.

8.3 Domain Configuration Information

Correct operation of the protocol described in this international standard assumes that a minimum amount of information is available to both the inter-domain and the intra-domain routing protocols. The information is static in nature, and is not expected to change frequently. This protocol specifies the content of the information in terms of managed objects.

The information required by a BIS that implements the protocol described in this international standard is:

a) Location and identity of adjacent intra-domain ISs:

The managed object **INTRA-IS** lists the NETs of systems to which the local BIS may deliver an inbound NPDU whose destination lies within the BIS's routing domain. The managed object contains the NETs of ISs that support the intra-domain routing protocol and are located on the same common subnetwork as the local BIS. In particular, if the BIS participates in the intra-domain routing protocol (that is, the protocol machines for both intra- and inter-domain routing are located in the same real system),

then the NET of the local BIS will be listed in the managed object **INTRA-IS**.

b) Location and identity of BISs in the domain:

This information permits a BIS to identify all other BISs located within its routing domain. This information is contained in the managed object **INTERNAL-BIS**, which contains a set of NETs which identify the BISs in the domain.

c) Location and identity of BISs in adjacent domain:

Each BIS needs information to identify the NETs of each BIS located in an adjacent RD and reachable via a single subnetwork hop. This information is contained in managed object **EXTERNAL-BIS-NEIGHBORS**, which consists of a list of NETs.

d) NETs and NSAP addresses of all systems in the routing domain:

This information is used by the BIS to construct its network layer reachability information. This information is contained within the managed object **INTERNAL-SYSTEMS**, which is a list of the systems contained within the routing domain.

e) Local RDI:

This information is contained in managed object **LOCAL-RDI**; it is the RDI of the routing domain in which the BIS is located.

f) RIB-AttsSet:

This managed object lists all of the RIB-Atts (RIB attributes) which are supported by BISs located in this routing domain. As noted in clause 8.10.1, a RIB-Att is a set of Distinguishing Attributes.

g) RDC-Config:

This information identifies all the routing domain confederations (RDCs) to which the RD of the local BIS belongs, and it describes the nesting relationships that are in force between them. It is contained in managed object **RDC-Config**.

h) Local SNPs:

This managed object contains a list of the SNPs of the BIS.

8.4 Advertising NLRI Information

The NLRI field in an UPDATE PDU contains information about the NSAP addresses of systems that reside within a given routing domain or whose NSAP addresses are under control of the administrator of that routing domain; it should not be used to convey information about the operational status of these systems. The information in the NLRI field is intended to convey static administrative information rather than dynamic transient information: for example, it is not necessary to report that a given system has changed its status from online to offline.

The following guidelines for inclusion of NSAP address prefixes in the NLRI field of UPDATE PDUs originated within a given routing domain will provide efficient operation of this protocol:

- a) NSAP addresses that are within the control of the administrator of a given routing domain may be reported in the NLRI field for that routing domain. The NSAP prefixes can provide information about systems that are online, systems that are offline, or unallocated NSAP addresses. The ability to include unallocated NSAP addresses and NSAP addresses of offline systems in the NLRI allows a routing domain administrator to advertise compact prefixes, thus minimizing the amount of data carried in the BISPDU.
- b) NSAP addresses that are known to correspond to systems that are not under control of the routing domain administrator should not be included in the NLRI field for that routing domain. By not listing NSAP address prefixes that identify systems that are not under his control, a routing domain administrator will comply with the deployment guidelines for ISs and ESs as described in clause 8.2.2, thus assuring correct operation of this protocol.
- 2 c) For efficient operation of this protocol, the WITH-
- 2 DRAWN ROUTES field should not be used to report the NLRI of systems in the local routing domain that are offline. This field should be used only to advertise NSAP prefixes that are no longer under control of the administrator of the local routing domain, regardless of whether such systems are online or offline.

NOTES:

- a) Although the protocol in this international standard will operate correctly if each system is reported individually as a maximum-length NSAP address prefix, this will result in a large amount of routing information, and hence can result in inefficient operation of this protocol.
- b) This protocol provides no means to verify that the preceding guidelines are followed. However, it is within the prerogative of the administrator of any routing domain to implement policies that ignore UPDATE PDUs that contain an excessive amount of NLRI information or that can cause inefficient operation of this protocol.

8.5 Reliable Delivery of BISPDU

The protocol described in this international standard is a connection oriented protocol in which the underlying Network Layer service is used to establish full-duplex communication channels between pairs of BISs, as described in clause 8.6. It uses the mechanisms described below to provide reliable delivery of BISPDU between pairs of BISs communicating over a connection.

8.5.1 Checksum over BISPDU

A checksum that covers the contents of each BISPDU is always present in the Validation Pattern field of that BISPDU. If the Authentication Type code of the OPEN PDU was 2, the Validation Pattern field contains an encrypted checksum generated in accordance with clause 8.9.; if it was 1, then the Validation Pattern field contains an unencrypted checksum generated as defined below. A BISPDU received with an incorrect checksum (either encrypted or unencrypted) shall be discarded without any further action.

For all BISPDU that flow on a connection that was established in response to an OPEN PDU whose authentication code field was equal to 1, the validation field shall contain a 16-octet unencrypted checksum:

- a) Generating a Validation Pattern:

The contents of the Validation Pattern field that is included in an outbound BISPDU shall be generated by applying the algorithm of annex Appendix B to the input data stream that consists of the contents of the entire BISPDU with all bits of the Validation Pattern field initially set to 0. The output of this step is an unencrypted 16-octet long checksum, which shall be placed in the Validation Pattern field of the BISPDU.

- b) Checking the Validation Pattern of an Inbound BISPDU:

The contents of the Validation Pattern field of an inbound BISPDU shall be checked by applying the algorithm of annex Appendix B to the contents of the inbound BISPDU with its Validation Pattern set to all zeros. Call this quantity the "reference pattern".

If the "reference pattern" matches the contents of the Validation Pattern field of the inbound BISPDU, then the BISPDU's checksum is correct; otherwise, it is incorrect.

8.5.2 Sequence Numbers

A sequence number is a 4-octet unsigned value. Sequence numbers shall increase linearly from 1 up to a maximum value of $2^{32} - 1$. The value 0 is not a valid sequence number. Sequence numbers do not wrap.

The rules for manipulating sequence numbers are:

- When the FSM for a given BIS-BIS connection is initialized, it shall start with its sequence number equal to 1.
- The sequence number shall not be incremented for the KEEPALIVE PDU, the CEASE PDU, and the IDRP ERROR PDU.
- If a BIS-BIS connection is closed and subsequently re-established without re-initializing the

associated FSM, the next sequential sequence number shall be used.

- After it sends a BISPDU with the maximum permissible sequence number ($2^{32} - 1$), a BIS shall close the BIS-BIS connection (see 8.6.2) by issuing a Stop Event and then establish a new connection which shall start with sequence number equal to 1.

NOTE: The maximum lifetime of an 8473 NPDU is 128 seconds. Since the architectural constant **CloseWaitDelay** is 150 seconds, it can be guaranteed that all outstanding BISPDU (which are carried as user data within an encapsulating 8473 NPDU) will have expired before the new BIS-BIS connection is established.

8.5.3 Flow Control

The IDRP flow control process shall obey the following rules:

- a) A separate series of sequence numbers shall be maintained for each direction of a BIS-BIS connection, with the initial value chosen by the sender of a BISPDU and declared in the *Sequence* field of its OPEN PDU. The local BIS will maintain a window to manage transmission of BISPDU to the remote BIS. The sender's *lower window edge* shall be set to the initial sequence number plus one; the sender's *upper window edge* shall be set to the *lower window edge* plus the value of *credit offered* contained in the peer BIS's OPEN PDU. Record is also kept of the *next expected sequence number* for an inbound UPDATE PDU or RIB REFRESH PDU to be received from the peer BIS; this is initially set to the value of one plus *Sequence* that is carried in the peer BIS's OPEN PDU.
- b) An UPDATE PDU or RIB REFRESH PDU shall not be sent if the upper window edge is less than or equal to the lower window edge. When a BISPDU is sent, the value of *Sequence* in the fixed header shall be set to the current value of the lower window edge. If an UPDATE PDU or a RIB REFRESH PDU is sent, then the lower window edge shall be incremented by one; if an UPDATE or RIB REFRESH PDU is not sent, then the lower window edge is not incremented. The value of *Acknowledgement* shall be set to the value of the next expected sequence number less one. The value of *credit offered* shall be set to the number of additional BISPDU that the local BIS is currently able to accept from the peer BIS. The value of *credit available* shall be set to the upper window edge less the lower window edge. The local BIS shall retain a copy of the transmitted BISPDU for possible retransmission.
- c) An incoming UPDATE PDU or RIB REFRESH PDU whose *Sequence* value corresponds to the next expected sequence number shall be accepted and passed to the Receive Process; the next expected

sequence number shall be incremented by one. An incoming UPDATE PDU or RIB REFRESH PDU whose *Sequence* does not correspond to the next expected sequence number shall be discarded (unless re-ordering is supported as a local implementation option).

Whenever a BIS receives an UPDATE PDU, RIB REFRESH PDU, or KEEPALIVE PDU, it shall inspect its *Acknowledgement* and *credit offered* fields. Any BISPDU retained for retransmission whose sequence number is less than or equal to the value of the *Acknowledgement* field shall be discarded. If the sum of one plus the value of *Acknowledgement* plus the value of *credit offered* in the received BISPDU is greater than the local BIS's current upper window edge, then the BIS shall set its upper window edge to this sum.

- d) A BIS shall acknowledge receipt of incoming UPDATE PDUs and RIB REFRESH PDUs within a period t_A of their receipt. The acknowledgement may be accomplished by means of an UPDATE PDU or a RIB REFRESH PDU sent as outlined in item b above. However, if no UPDATE PDU or RIB REFRESH PDU is available to be sent, then a KEEPALIVE PDU may be sent instead, with its *Sequence* set to the lower window edge and its *Acknowledgement*, *credit offered*, and *credit available* set as in step b above.

- e) If a retained BISPDU remains unacknowledged after a period t_R , then it shall again be transmitted and again retained for possible retransmission. If, for a retained BISPDU, t_R expires after n retransmissions, the local BIS shall issue a Stop Event to close the BIS-BIS connection.

NOTE: The value t_R should be chosen to be greater than the value $t_A + 2 \times L$, where L is the transmission delay over the subnetwork or virtual link between the pair of communicating BISs.

- f) The local BIS shall provide its peer BIS with sufficient credit to send further BISPDU as long as the local BIS has resources to receive them. Therefore, if the local BIS receives a BISPDU whose *credit available* is equal to zero (that is, the peer BIS believes itself unable to send additional BISPDU), then as soon as resources are available locally, the local BIS shall send an UPDATE PDU or a RIB REFRESH PDU, if appropriate. If not, then a KEEPALIVE PDU shall be sent.

NOTE: An UPDATE PDU of minimal size will contain the Unfeasible Route Count field with a value of zero, but will not contain any path attributes or NLRI. Thus, its size will be only 33 octets.

A KEEPALIVE PDU that advertises a non-zero value of *credit offered* in response to a received BISPDU with a *credit available* of zero shall be retransmitted within a period t_R unless the local BIS receives:

- 2 1) an UPDATE PDU or RIB REFRESH PDU that
2 has a sequence number greater than that of
2 the BISPDU that originally reported the *credit*
2 *available* of zero, or
2 2) a KEEPALIVE PDU that has a sequence
2 number greater than that of the BISPDU that
2 originally reported the *credit available* of
2 zero, and that reports a non-zero *credit avail-*
2 *able* field.
- 2 If t_R expires after n retransmissions, then the
2 local BIS shall issue a Stop Event to close the
2 connection.
- 2 g) A BIS that sends a BISPDU with zero credit avail-
2 able shall respond within a period t_A to a BISPDU
2 that causes the upper window edge to be
2 increased. The response shall consist of an
2 UPDATE PDU or a RIB REFRESH PDU, if avail-
2 able, or a KEEPALIVE PDU, if not.
- 2 h) A BIS that has not sent any BISPDU for a period
2 t_I shall send a KEEPALIVE PDU, with *Sequence*
2 equal to the lower window edge, and *Acknowl-*
2 *edgement*, *credit offered*, and *credit available* set
2 as in step b above.
- 2 **NOTE:** The condition $t_I > t_R$ should be satisfied,
2 where t_I is one third of the Hold Timer value.
- 2 i) A BIS that has sent a BISPDU containing a *credit*
2 *offered* of zero shall, as soon as its local
2 resources become available to process additional
2 BISPDU from its peer, send an UPDATE PDU or
2 RIB REFRESH PDU, if appropriate, containing a
2 non-zero value of *credit offered*. If neither of
2 these BISPDU types is appropriate, then a
2 KEEPALIVE PDU shall be sent.
- 2 j) The BIS shall issue a Stop Event to close the
2 BIS-BIS connection if no BISPDU are received
2 for a period equal to the value of Hold Time that
2 is carried in the OPEN PDU.

8.6 BIS-BIS Connection Management

The protocol described in this international standard relies on the underlying Network layer service to establish a full-duplex communications channel between each pair of BISs.

8.6.1 BIS Finite State Machines

- 1 A BIS shall maintain one finite state machine (FSM)
1 for each BIS-BIS connection that it supports, and each
1 FSM in a given BIS shall run independently of one
1 another. A BIS-BIS connection will progress through
1 a series of states during its lifetime, which are sum-
1 marized in the state table shown in Table 2.

- 1 In describing the FSM transitions in response to
1 receipt of BISPDU, the following shorthand notation
1 is used:

- 1 a) *Receive with no errors* means that the none of
1 the error conditions defined in the appropriate
1 subclause of 8.20 have been detected, and that
1 the BISPDU has satisfied the flow control proce-
1 dures of 8.5.3.
- 1 b) *Receive with errors* means that an error condition
1 defined in the appropriate subclause of 8.20 has
1 been detected, or the BISPDU has not satisfied
1 the flow control procedures of 8.5.3.

- 1 It is possible to receive a BISPDU which is properly
1 formed, but which normally should not be received
1 while the FSM is in the given state. Such an event
1 constitutes an *FSM Error*. If an FSM Error can occur
1 for a given state, it is shown in the description of that
1 state.

8.6.1.1 CLOSED State

- 1 Initially the BIS Finite State Machine is in the CLOSED
1 state. The CLOSED state exists when no BIS-BIS con-
1 nection exists and there is no connection record allo-
1 cated.

- 1 While in the CLOSED state, the BIS shall take the fol-
1 lowing actions:

- 1 a) If the BIS receives a Stop Event, no action shall
1 be taken and the FSM shall remain in the
1 CLOSED state.
- 1 b) When the FSM receives a Start Event, the local
1 BIS shall allocate a connection record, shall gen-
1 erate an initial sequence number (see 8.5.2), and
1 shall send an OPEN PDU to the remote BIS. The
1 sequence field of the OPEN PDU shall contain the
1 Initial Sequence Number (ISN), and the acknowl-
1 edgement field shall contain 0. The FSM shall
1 enter the OPEN-SENT state.
- 1 c) If the BIS receives a BISPDU with a header error,
1 it shall ignore it, and the FSM shall remain in the
1 CLOSED state.

- 1 The BIS shall ignore any OPEN PDUs, IDRP
1 ERROR PDUs, UPDATE PDUs, KEEPALIVE PDUs,
1 CEASE PDUs, or RIB REFRESH PDUs, and the
1 FSM shall remain in the CLOSED state. The BIS
1 does not discriminate between receiving these
1 types of BISPDU with errors or with no errors.

8.6.1.2 OPEN-SENT State

- 1 While in the OPEN-SENT state, the BIS shall take the
1 following actions:

- 1 a) If the FSM receives a Start event, the BIS shall
1 ignore it, and the FSM shall remain in the
1 OPEN-SENT state.
- 1 b) If the FSM receives a Stop event, the BIS shall
1 send a CEASE PDU to its peer, and the FSM shall
1 enter the CLOSE-WAIT state.

- 1 c) If the BIS receives a BISPDU with header errors
1 (see 8.20.1), it shall log the error event, and the
1 FSM shall remain in the OPEN-SENT state.
- 1 d) If the BIS receives an OPEN PDU with errors (see
1 8.20.2), it shall send an IDRP ERROR PDU to the
1 adjacent BIS, using the same sequence number
1 previously carried in the OPEN PDU to that BIS.
1 The IDRP ERROR PDU shall acknowledge the
1 remote BIS's OPEN PDU. The FSM shall then
1 enter the CLOSE-WAIT state.
- 1 e) If the BIS receives an OPEN PDU with no errors
1 that does not acknowledge its own previously
1 sent OPEN PDU, then the local BIS shall resend
1 its own OPEN PDU with the same sequence
1 number and with an acknowledgement of the
1 remote BIS's OPEN PDU. The FSM shall then
1 change its state to OPEN-RCVD. (There have
1 been simultaneous attempts to open a con-
1 nection, with the OPEN PDUs passing each other
1 in transit.)
- 1 f) If the BIS receives an OPEN PDU with no errors
1 that acknowledges its own previously sent OPEN
1 PDU, the local BIS shall send a KEEPALIVE PDU
1 that acknowledges the OPEN PDU received from
1 the remote BIS. The FSM shall then enter the
1 ESTABLISHED state.
- 1 g) If the BIS receives an IDRP ERROR PDU, either
1 with or without error, it shall send a CEASE PDU,
1 and the FSM shall change its state to
1 CLOSE-WAIT.
- 1 h) If the BIS receives a RIB REFRESH PDU, or
1 UPDATE PDU, either with or without errors, it
1 shall issue an IDRP ERROR PDU, indicating "FSM
1 Error". The FSM shall then enter the
1 CLOSE-WAIT state.
- 1 i) If the BIS receives a KEEPALIVE PDU, it shall
1 issue an IDRP ERROR PDU, indicating "FSM
1 Error". The FSM shall then enter the
1 CLOSE-WAIT state.
- 1 j) If the BIS receives a CEASE PDU, the FSM shall
1 enter the CLOSE-WAIT state.

8.6.1.3 OPEN-RCVD State

- 1 While in the OPEN-RCVD state, the BIS shall take the
1 following actions:
- 1 a) If the BIS receives a Start Event, it shall ignore it
1 and the FSM shall remain in the OPEN-RCVD
1 state.

- 1 b) If the BIS receives a Stop Event, it shall send a
1 CEASE PDU to the remote BIS, and the FSM shall
1 enter the CLOSE-WAIT state.
- 1 c) If the Hold Timer expires, the BIS shall issue an
1 IDRP ERROR PDU to the peer BIS, reporting a
1 Hold_Timer_Expired error. The FSM shall enter
1 the CLOSE-WAIT STATE.
- 1 d) If the BIS receives a BISPDU with a header error,
1 it shall log the error event, and the FSM shall
1 remain in the OPEN-RCVD state.
- 1 e) If the BIS receives a KEEPALIVE PDU that
1 acknowledges its previously sent OPEN PDU, then
1 the FSM shall enter the ESTABLISHED state.
- 1 f) If the BIS receives a KEEPALIVE PDU that does
1 not acknowledge its previously sent OPEN PDU,
1 the BIS shall ignore it. The FSM shall remain in
1 the OPEN-RCVD state.
- 1 g) If the BIS receives a CEASE PDU, it shall enter
1 the CLOSE-WAIT state.
- 1 h) If the BIS receives an OPEN PDU with no errors
1 from the remote BIS that acknowledges the local
1 BIS's previously sent OPEN PDU, the BIS shall
1 send a KEEPALIVE PDU that acknowledges OPEN
1 PDU received from the remote BIS. The FSM
1 shall then enter the ESTABLISHED state.
- 1 i) If the BIS receives an OPEN PDU with no errors
1 that does not acknowledge the local BIS's previ-
1 ously sent OPEN PDU, then the local BIS shall
1 resend its own OPEN PDU with the same
1 sequence number, and shall also include an
1 acknowledgement of the remote BIS's OPEN PDU.
1 The FSM shall remain in the OPEN-RCVD state.
- 1 j) If the BIS receives an UPDATE PDU or RIB
1 REFRESH PDU with errors, the shall send an IDRP
1 ERROR PDU to the remote BIS, and the FSM shall
1 enter the CLOSE-WAIT state.
- 1 k) If the BIS receives an UPDATE PDU or RIB
1 REFRESH PDU with no errors that acknowledges
1 the OPEN PDU previously sent by the local BIS,
1 the FSM shall enter the ESTABLISHED state.
- 1 l) If the BIS receives an UPDATE PDU or RIB
1 REFRESH PDU with no errors that does not
1 acknowledge the OPEN PDU previously sent by
1 the local BIS, the BIS shall ignore it, and the FSM
1 shall remain in the OPEN-RCVD state.
- 1 m) If the BIS receives an IDRP ERROR PDU, either
1 with or without errors, it shall send a CEASE PDU
1 to the remote BIS, and the FSM shall enter the
1 CLOSE-WAIT state.

Table 2 (Page 1 of 2). BIS Finite State Machine. This table summarizes the effects that its inputs will have on an IDRP FSM, giving both state transitions and the actions to be taken.

STATE → INPUT ↓	CLOSED	OPEN-RCVD	OPEN-SENT	CLOSE-WAIT	ESTABLISHED
Start Event	S=OPEN-SENT A=send OPEN PDU	S=OPEN-RCVD A=none	S=OPEN-SENT A=none	S=CLOSE-WAIT A=none	S=ESTABLISHED A=none
Stop Event	S=CLOSED A=none	S=CLOSE-WAIT A=send CEASE PDU	S=CLOSE-WAIT A=send CEASE PDU	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=send CEASE PDU
Expiration of CloseWaitDelay Timer	S=CLOSED A=none	S=OPEN-RCVD A=none	S=OPEN-SENT A=none	S=CLOSED A=log event	S=ESTABLISHED A=none
Expiration of Hold Timer	S=CLOSED A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to report error	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to report error
Receive BISPDU with Header Error	S=CLOSED A=none	S=OPEN-RCVD A=log error event	S=OPEN-SENT A=log error event	S=CLOSE-WAIT A=none	S=ESTABLISHED A=log error event
Receive KEEPALIVE PDU with no errors	S=CLOSED A=none	If ACK is correct, S=ESTABLISHED A=none If ACK is incorrect, S=OPEN-RCVD A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to report FSM Error	S=CLOSE-WAIT A=none	S=ESTABLISHED A=Restart Hold Timer
Receive CEASE PDU with no errors	S=CLOSED A=none	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=none
Receive OPEN PDU with errors	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report OPEN PDU error	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report OPEN PDU error	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report OPEN PDU error
Receive OPEN PDU, no error	S=CLOSED A=none	If ACK is correct, S=ESTABLISHED A=send KEEPALIVE PDU If ACK is incorrect, S=OPEN-RCVD, A=send OPEN PDU	If ACK is correct, S=ESTABLISHED A=send KEEPALIVE PDU If ACK is incorrect, S=OPEN-RCVD A=send OPEN PDU	S=CLOSE-WAIT A=none	S=ESTABLISHED A=none
Receive UPDATE PDU with errors	S=CLOSED A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report UPDATE PDU error	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report FSM error	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report UPDATE PDU error
Receive UPDATE PDU with no errors	S=CLOSED A=none	If ACK is correct, S=ESTABLISHED A=none If ACK is incorrect, S=OPEN-RCVD A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report FSM error	S=CLOSE-WAIT A=none	S=ESTABLISHED A=8.14, restart Hold Timer
Receive IDRP ERROR PDU with errors	S=CLOSED A=none	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU
Receive IDRP ERROR PDU with no errors	S=CLOSED A=none	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU	S=CLOSE-WAIT A=Send CEASE PDU

Table 2 (Page 2 of 2). BIS Finite State Machine. This table summarizes the effects that its inputs will have on an IDRP FSM, giving both state transitions and the actions to be taken.

STATE → INPUT ↓	CLOSED	OPEN-RCVD	OPEN-SENT	CLOSE-WAIT	ESTABLISHED
Receive RIB REFRESH PDU with errors	S=CLOSED A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report RIB REFRESH PDU error	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report FSM error	S=CLOSE-WAIT A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to peer BIS to report RIB REFRESH PDU error
Receive RIB REFRESH PDU with no errors	S=CLOSED A=none	If ACK is correct, S=ESTABLISHED A=none If ACK is incorrect, S=OPEN-RCVD A=none	S=CLOSE-WAIT A=Send IDRP ERROR PDU to report FSM Error	S=CLOSE-WAIT A=none	S=ESTABLISHED A=Restart Hold Timer

NOTES:

- "S" indicates the state into which the FSM will make a transition after performing the indicated action.
- "A" indicates the action to be taken.
- "X.Y.Z" is shorthand notation for "do as specified in clause X.Y.Z".
- The phrase "no errors" for a given BISPDU type means that no condition described in the appropriate subclause of 8.20 has been detected, and the BISPDU has complied with the flow control procedures of 8.5.4.
- The phrase "with errors" for a given BISPDU type means that a condition described in the appropriate subclause of 8.20 has been detected, or the BISPDU has not complied with the flow control procedures of 8.5.4.
- Since the KEEPALIVE PDU and the CEASE PDU consist of only a fixed BISPDU header, errors in these BISPDUs are handled as Header Errors. Hence, there are no explicit entries in the table for "KEEPALIVE with errors" or "CEASE with errors".

8.6.1.4 ESTABLISHED State

The ESTABLISHED state is entered from either the OPEN-SENT or the OPEN-RCVD states. It is entered when a connection has been established by the successful exchange of state information between two sides of the connection. Each side has exchanged and received such data as initial sequence number, maximum PDU size, credit offered, protocol version number, hold time, and RDI of the other side. In addition, the remote BIS may also have been authenticated.

In ESTABLISHED state, both BISs that are involved in the connection may exchange UPDATE PDUs, KEEPALIVE PDUs, IDRP ERROR PDUs, RIB REFRESH PDUs, and CEASE PDUs.

While in the ESTABLISHED state, the local BIS shall take the following actions:

- If the FSM receives a Start Event, the FSM shall ignore it, and the FSM shall remain in the ESTABLISHED state.
- If the FSM receives a Stop event, the BIS shall send a CEASE PDU to the peer BIS. The FSM shall enter the CLOSE-WAIT state.
- If the Hold Timer expires, the BIS shall issue an IDRP ERROR PDU to the remote BIS, reporting a

Hold_Timer error. The FSM shall enter the CLOSE-WAIT state.

- If the BIS receives a BISPDU with a header error, it shall log the error event, and the FSM shall remain in the OPEN-RCVD state.
- If the BIS receives a KEEPALIVE PDU, it shall restart its Hold Timer, and the FSM shall remain in the ESTABLISHED state.
- If the BIS receives a CEASE PDU, the FSM shall enter the CLOSE-WAIT state.
- If an OPEN PDU with no errors is received from the peer BIS, it shall be ignored: that is, the BIS shall not respond to duplicate delayed OPEN PDUs. The FSM shall remain in the ESTABLISHED state.
- If the BIS receives an UPDATE PDU with no errors, the BIS shall perform the actions provided in clause 8.14, and shall restart its Hold Timer. The FSM shall remain in the ESTABLISHED state.
- If the BIS receives a RIB REFRESH PDU with no errors, the BIS shall perform the actions provided in clause 8.10.3, and shall restart its Hold Timer. The FSM shall remain in the ESTABLISHED state.
- If the BIS receives an UPDATE PDU with errors, an OPEN PDU with errors, or a RIB REFRESH PDU with errors, it shall send an IDRP ERROR PDU to

- 1 the remote BIS to report the error, and the FSM
1 shall enter the CLOSE-WAIT state.
- 1 k) If the BIS receives an IDRP ERROR PDU, either
1 with or without errors, it shall send a CEASE PDU
1 to the remote BIS. The FSM shall enter the
1 CLOSE-WAIT state.

8.6.1.5 CLOSE-WAIT State

When an FSM enters the CLOSE-WAIT state, the local BIS is preparing to close the connection with the remote BIS. Upon entering this state, the local BIS shall mark all entries in the Adj-RIB-In associated with the adjacent BIS as unreachable, and shall then re-run its Decision Process. The CloseWaitDelay timer shall be started.

While in the CLOSE-WAIT state, the BIS shall take the following actions:

- a) If the CloseWaitDelay timer expires, the BIS shall log the event. The connection record for the peer BIS shall be deallocated, and the connection cease to exist. The FSM shall enter the CLOSED state.
- 1 b) If an IDRP ERROR PDU is received, either with or
1 without errors, the BIS shall send a CEASE PDU
1 to the remote BIS. The FSM shall remain in the
1 CLOSE-WAIT state.
- 1 c) The BIS shall take no action on any of the fol-
1 lowing inputs, and the FSM shall remain in the
1 CLOSE-WAIT state:
- 1 — Start event
 - 1 — Stop event
 - 1 — Receipt of BISPDU with header errors
 - 1 — Receipt of KEEPALIVE PDU, OPEN PDU (with
1 or without error), CEASE PDU, UPDATE PDU
1 (with or without error), or RIB REFRESH PDU
1 (with or without error)
 - 1 — Expiration of Hold Timer

8.6.2 Closing a Connection

The closing of a connection can be initiated by a Stop event generated by the local system, by receipt of an incorrect PDU, by receipt of an IDRP ERROR PDU, or by receipt of a CEASE PDU.

- a) In the case of the Stop event, the local system shall send a CEASE PDU to the other side of the connection, shall deallocate all resources associated with the peer BIS, and the FSM shall then enter the CLOSE-WAIT state.
- b) In the case of receiving an incorrect PDU, the local system may send an IDRP ERROR PDU, and the FSM shall enter the CLOSE-WAIT state. Unless otherwise specified, the Data field of the IDRP ERROR PDU shall be empty.
- 2 c) In the case of receiving an IDRP ERROR PDU
2 from an adjacent BIS, the local system shall send

- 2 a CEASE PDU to the adjacent BIS, and shall deal-
2 locate all local resources associated with it, and
2 the FSM shall enter the CLOSE-WAIT state.
- 2 d) In the case of receiving a CEASE PDU from an
2 adjacent BIS, the FSM shall enter the
2 CLOSE-WAIT state.

8.7 Version Negotiation

BIS peers may negotiate the version number of IDRP by making successive attempts to open a BIS-BIS connection, starting with the highest supported version number (contained in managed object **version**) and decrementing the number each time a connection attempt fails. The lack of support for a particular IDRP version is indicated by an IDRP ERROR PDU with error code "OPEN_PDU_Error" and an error subcode of "Unsupported_Version_Number". One BIS may determine the highest version number supported by the other BIS (as advertised in its OPEN PDU) by examining the "Data" field of the received IDRP ERROR PDU. No further retries should be attempted if the version number reaches zero.

8.8 Checksum Algorithm

The checksums used in this international standard shall be generated in accordance with the procedures described in normative annex Appendix B. For an input data stream of any length, this algorithm will generate a checksum that is 16 octets long. This algorithm shall be used to generate the checksums for both the BISPDU and the RIBs.

8.9 Authentication Mechanism

NOTE: This international standard includes as an optional function a mechanism that can be used for authentication of the source of a BISPDU. Other security-related facilities (for example, protection against replay of BISPDU or the ability to re-key during a BIS-BIS connection) are not intended to be provided by this protocol, and therefore are not specified in this International Standard.

All of the relevant security services identified in ISO 7498-2, including authentication, could be achieved by the use of an appropriate security protocol specified for the provision of secure ISO 8473 communications.

For an OPEN PDU with an authentication code field of 2, and for all BISPDU that flow on a BIS-BIS connection established by this OPEN PDU, the validation field shall contain a 16-octet encrypted checksum:

- a) Generating a Validation Pattern:

The contents of the Validation Pattern field that is included in an outbound BISPDU shall be generated by the following two step process, which is illustrated in Figure 6:

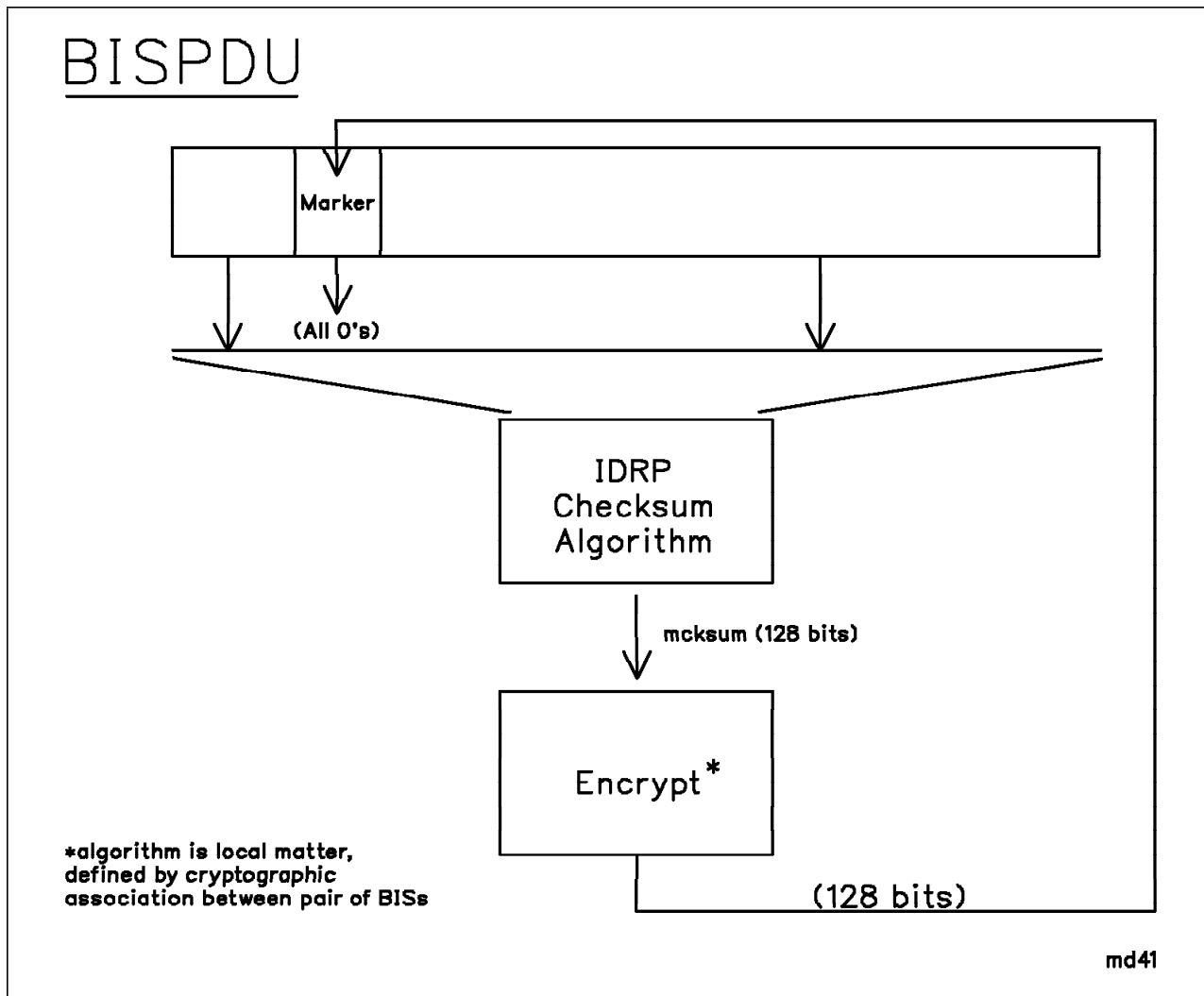


Figure 6. Validation Pattern in Support of the Authentication Function

- 1) An unencrypted checksum that covers the contents of the BISPDU shall be generated by applying the procedures of annex Appendix B to the input data stream that consists of the contents of the entire BISPDU with all bits of the Validation Pattern field initially set to 0. The output of this step is an unencrypted 16-octet long checksum, which is called *chksum*.
- 2) The 16-octet quantity *chksum* shall be encrypted, and the encrypted pattern shall be placed in the Validation Pattern field of the BISPDU.
- 3) The presence or absence of the authentication function is specified in a "per BIS-BIS connection" basis. Thus, a given BIS may support some BIS-BIS connections that use authentication, and others that do not.

NOTES:

- 1) The encryption algorithm must be agreed upon in the cryptographic association set up by the two BISs involved in the authentication process. This international standard does not mandate use of a specific encryption algorithm. Explicit indication of the specific algorithm to be used is outside the scope of IDRPs. However, the "Authentication Data" field of IDRPs OPEN PDU can be used to specify an

- b) Checking the Validation Pattern of an Inbound BISPDU:

The contents of the Validation Pattern field of an inbound BISPDU shall be checked by the following procedures:

- 1) Apply the IDRPs checksum algorithm to the data stream that consists of the contents of the inbound BISPDU with its Validation

Pattern set to all zeros. Call this quantity the "reference pattern".

- 2) Decrypt the Validation Pattern field of the inbound BISPDU, calling the result the "received pattern".

If the "reference pattern" and the "received pattern" are identical, then the peer BIS has been authenticated, and the inbound BISPDU shall be accepted. If the "reference pattern" and the "received pattern" are not identical, the receiving BIS shall inform system management that an authentication failure has occurred. The incoming BISPDU shall be ignored. The receiving BIS shall not send an IDRP ERROR PDU to the peer-BIS because the identity of the peer has not been authenticated.

- 1 **NOTE:** If a BISPDU has a malformed header, it will be discarded. As a result, the Validation Pattern of such BISPDU's will not be checked.

8.10 Routing Information Bases

The Routing Information Base (RIB) within a BIS consists of three distinct parts, as shown in Figure 7:

- a) **Adj-RIBs-In:** The Adj-RIBs-In store routing information that has been learned from inbound UPDATE PDUs. Their contents represent routes that are available as input to the Decision Process. A BIS must support at least one Adj-RIB-In for each of its neighbor BISs; it may optionally support several Adj-RIBs-In for a given neighbor BIS. Within the set of Adj-RIBs-In associated with a given neighbor BIS, no two shall have the same RIB-Att (see clause 8.10.1).
- b) **Loc-RIBs:** The Loc-RIBs contain the local routing information that the BIS has selected by applying its local policies to the routing information contained in its Adj-RIBs-In. A BIS may support multiple Loc-RIBs. No two Loc-RIBs within a given BIS shall have the same RIB-Att (see clause 8.10.1). Information in the Loc-RIB is used to build the Adj-RIBs-Out.
- c) **Adj-RIBs-Out:** The Adj-RIBs-Out store the information that the local BIS has selected for advertisement to its neighbors. A BIS must support at least one Adj-RIB-Out for each of its neighbor BISs; it may optionally support several Adj-RIBs-Out for a given neighbor BIS. Within the set of Adj-RIBs-Out associated with a given neighbor BIS, no two shall have the same RIB-Att (see clause 8.10.1). The routing information stored in the Adj-RIBs-Out will be carried in the local BIS's UPDATE PDUs and advertised to its neighbor BISs.

In summary, the Adj-RIBs-In contain unprocessed routing information that has been advertised to the local BIS by its neighbors; the Loc-RIBs contain the

routes that have been selected by the local BIS's Decision Process; and the Adj-RIBs-Out organize the selected routes for advertisement to specific neighbor BISs by means of the local BIS's UPDATE PDUs.

NOTE: Although the conceptual model distinguishes between Adj-RIBs-In, Adj-RIBs-Out, and Loc-RIBs, this does neither implies nor requires that an implementation must maintain three separate copies of the routing information. The choice of implementation (for example, 3 copies of the information vs. 1 copy with pointers) is not constrained by this standard.

8.10.1 Identifying an Information Base

Each information base (a single Adj-RIB-In, a single Loc-RIB, or a single Adj-RIB-Out) has one and only one RIB-Att associated with it. A RIB-Att is composed of a set of Distinguishing Attributes that the local BIS supports: in particular, a RIB-Att may consist of one or more Distinguishing Attributes that form a permissible combination, as defined in clause 8.11.2.

The managed object **RIB-AttsSet** explicitly enumerates all the RIB-Atts that a BIS supports. Managed object **RIB-AttsSet** shall not contain any pairs of RIB-Atts that are identical, thus assuring that each RIB-Att is unambiguous within the BIS.

All BISs located within a given routing domain shall support the same RIB-Atts: that is, managed object **RIB-AttsSet** shall be identical for every BIS within an RD. When a BIS receives an OPEN PDU from another BIS located in its own routing domain, it shall compare the information in the field *RIB-AttsSet* with the information in its local managed object **RIB-AttsSet**. If they do not match, then the appropriate error handling procedure in clause 8.20.2 shall be followed.

Each BIS shall support default information bases (Adj-RIBs-In, Adj-RIBs-Out, Loc-RIB, and FIB) that correspond to the RIB-Att that is composed of an empty set of Distinguishing Attributes.

NOTE: Because policy is a local matter, IDRP does not specify the criterion used to select the information to be placed in the default Loc-RIB. However, since the following mandatory path attributes are present in every route, it is suggested that RD_PATH and RD_HOP_COUNT should be used for this purpose.

8.10.2 Validation of RIBs

A BIS shall not continue to operate for an extended period with corrupted routing information. Therefore, the BIS shall operate in a "fail-stop" manner: when corruption of a RIB is detected, the BIS shall immediately take action to cease using the routes contained in the corrupted information base.

In the absence of an implementation-specific method for insuring this, the BIS shall perform the following checks at least every **maxRIBIntegrityCheck** seconds:

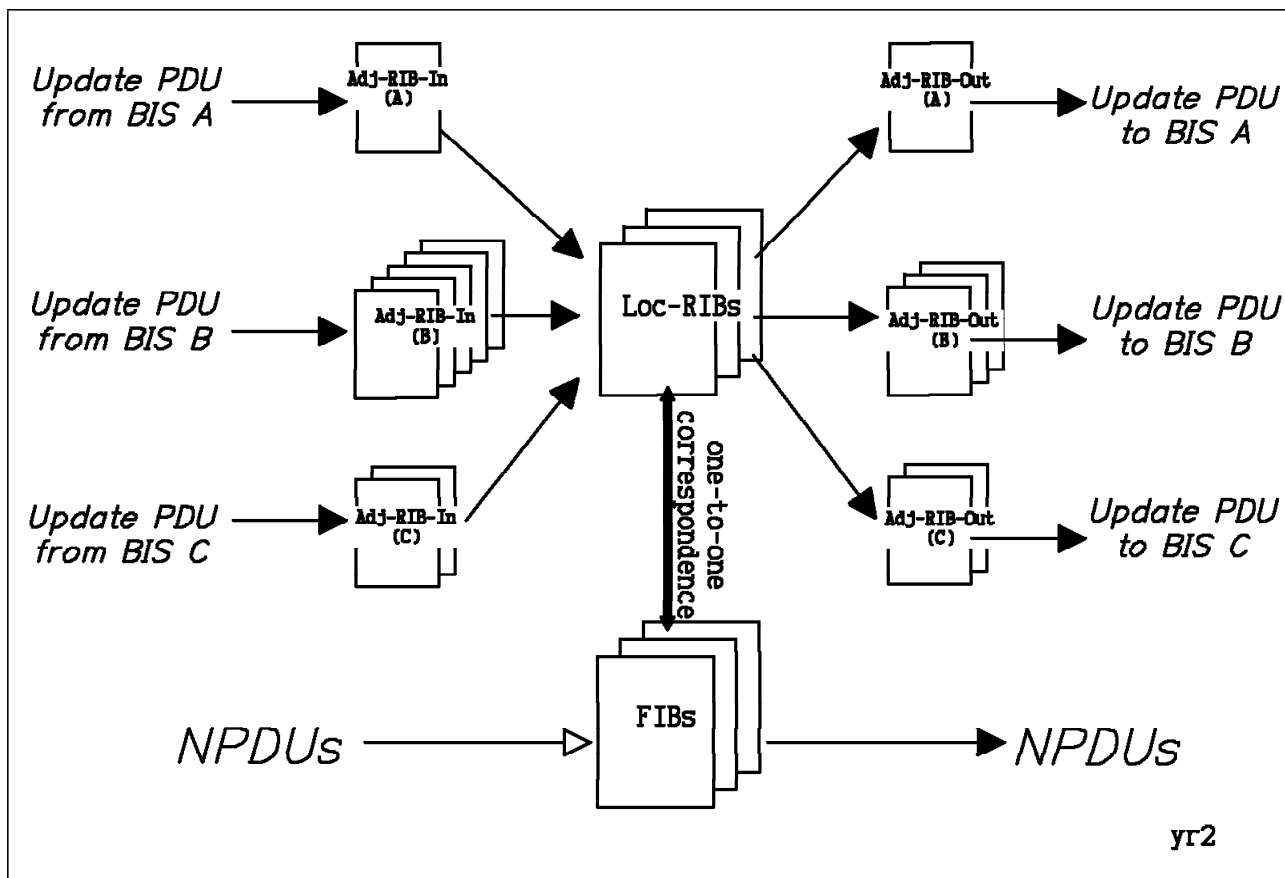


Figure 7. Routing Information Base. An RIB is comprised of Adj-RIBs-In, Adj-RIBs-Out, and Loc-RIBs

- a) Upon expiration of its **maxRIBIntegrityCheck** timer, the BIS shall recheck the checksum of the routing information contained in each of its Adj-RIBs-In in order to detect corruption of routing information while in memory.

If corruption is detected, the BIS shall purge the Adj-RIB-In, and shall notify System Management of a "Corrupted AdjRIBIn" event.

NOTE: This standard does not prescribe a specific checksum algorithm but notes that the procedures described in Appendix D satisfy the requirements given above. Other approaches can also be used: for example, it may be possible to use an incremental algorithm to compute the checksum for a given Adj-RIB-In when new information is received.

After detection of a corrupted Adj-RIB-In, a BIS may choose to issue a RIB REFRESH PDU, asking for a solicited refresh of the routing information from its peer BIS.

- b) On completion of its check of the Adj-RIBs-In, the BIS shall rerun its Decision Process, regardless of whether or not corruption of the Adj-RIBs-In has been detected. As a byproduct of running the Decision Process, the BIS will construct new information for its Loc-RIBs, and will then regenerate

its Adj-RIBs-Out and its FIBs. Thus, any corrupted information that may have been present in the Adj-RIBs-Out or the FIBs will be replaced as a result.

- c) Upon completion of these checks, the BIS shall reset the timer to the value **MaxRIBIntegrityCheck** with jitter applied in accordance with clause 8.16.3.3.

Since a given Adj-RIB-In that had been corrupted will have been purged before the Decision Process is re-executed, the defective information will not be used in the recalculation.

An explicit integrity check on the contents of the Loc-RIBs, Adj-RIBs-Out, and the FIBs is not required, since corrupted information will be replaced periodically when the Decision Process is re-run.

As a local option, a BIS may also choose to perform an explicit integrity check on the routing information in its Loc-RIBs, Adj-RIBs-Out, and FIBs. If such an integrity check detects that the information base has become corrupted, then the BIS shall immediately rerun its Decision Process, and should notify System Management of "Corrupt Loc-RIB", "CorruptAdjRIBOut", or "CorruptFIB", as appropriate.

8.10.3 Use of the RIB REFRESH PDU

The RIB REFRESH PDU can be used by a BIS to solicit a refresh of its Adj-RIBs-In by a neighbor BIS, or to send an unsolicited refresh to a neighbor BIS:

a) *Solicited Refresh*

A BIS may request a neighbor BIS to refresh one or more of the local BIS's Adj-RIBs-In by sending a RIB-REFRESH PDU that contains the OpCode for RIB-Refresh-Request and the RIB-Atts of the Adj-RIBs-In that it wants to be refreshed.

When the neighbor BIS receives a RIB-REFRESH PDU with OpCode RIB-Refresh-Request, it shall send back a RIB-REFRESH PDU with OpCode RIB-Refresh-Start, followed by a sequence of UPDATE PDUs that contain the information in its Adj-RIBs-Out associated with the requesting BIS. The neighbor BIS shall indicate the completion of the refresh by sending a RIB-REFRESH PDU with OpCode RIB-Refresh-End.

b) *Unsolicited Refresh*

A BIS may initiate an unsolicited refresh by sending a RIB-REFRESH PDU with OpCode RIB-Request-Start, followed by a sequence of UPDATE PDUs that contain the information in its Adj-RIBs-Out that been advertised to a given BIS. The completion of the refresh shall be indicated by sending the RIB-REFRESH PDU with OpCode RIB-Refresh-End.

When a BIS receives a RIB REFRESH PDU with OpCode 2 (RIB Refresh Start), it shall not change any of the routing information currently stored in the Adj-RIB-In which is identified by the distinguishing attributes of the RIB REFRESH PDU until the refresh cycle has been completed or has been aborted.

The BIS shall accumulate the routing information contained in all the UPDATE PDUs that are received in a completed refresh cycle. Completion of a refresh cycle is indicated by receipt of a RIB REFRESH PDU with OpCode 3 (RIB Refresh End). Then the BIS shall replace the previous routing information in the associated Adj-RIB-In with the routing information that was learned during the refresh cycle.

Abortion of a refresh cycle is indicated by receipt of another RIB REFRESH PDU with OpCode 2 (RIB Refresh Start) before receipt of a RIB REFRESH PDU with OpCode 3 (RIB Refresh End). In this case, any routing information learned in the time between receipt of the two successive RIB Refresh Starts shall be discarded, and a new refresh cycle (triggered by receipt of the second RIB Refresh Start) shall begin.

If the refreshing BIS receives a new RIB-Refresh-Request while it is in the middle of refresh (after sending RIB-REFRESH PDU with OpCode RIB-Refresh-Start, but before sending

RIB-REFRESH PDU with OpCode RIB-Refresh-End), then the current refresh shall be aborted and the new refresh is initiated.

8.11 Path Attributes

The UPDATE PDU contains fields which carry a set of *path attributes*.

8.11.1 Categories of Path Attributes

Path attributes fall into four categories:

- a) Well-known mandatory: these attributes must be recognized upon receipt by all BISs, and must be present in every UPDATE PDU
- b) Well-known discretionary: these attributes must be recognized upon receipt by all BISs, but are not necessarily present in an UPDATE PDU
- c) Optional transitive: these attributes need not be recognized upon receipt by all BISs, and are not necessarily present in an UPDATE PDU. Even if a given BIS does not recognize an optional transitive attribute, it may pass it on to other BISs
- d) Optional non-transitive: these attributes need not be recognized upon receipt by all BISs, and are not necessarily present in an UPDATE PDU. If it does not recognize an optional non-transitive attribute, a BIS shall ignore it and shall not include it in any of its own UPDATE PDUs.

A BIS shall handle optional attributes in the following manner:

- a) If a route with an unrecognized optional transitive attribute is accepted and passed on to other BISs, then the BIS shall set the PARTIAL bit in the attributes flag of its outbound UPDATE PDU to 1
- b) If a route with a recognized optional transitive attribute is accepted and passed on to other BISs, then the BIS shall not modify the value of the PARTIAL bit.
- c) If a route with an unrecognized optional non-transitive attribute is received, the receiving BIS shall ignore the attribute and shall not propagate that attribute to any other BIS. However, it may propagate the remainder of the route: that is, the route without the unrecognized optional non-transitive attribute.
- d) If a route with a recognized optional non-transitive attribute is received, the receiving BIS may accept this information, and may propagate it to other BISs.

BISs shall observe the following rules for attaching and updating the values of optional attributes:

- New optional transitive attributes may be attached to the path information by any BIS in the path, and that BIS shall then set the PARTIAL bit in the attributes flag of its UPDATE PDU to 1.

Table 3. Path Attribute Characteristics

Attribute	Category	Type Code	Length (octets)	Distinguishing
ROUTE-ID	well-known mandatory	1	variable	No
EXT_INFO	well-known discretionary	2	0	No
RD_PATH	well-known mandatory	3	variable	No
NEXT_HOP	well-known discretionary	4	variable	No
DIST_LIST_INCL	well-known discretionary	5	variable	No
DIST_LIST_EXCL	well-known discretionary	6	variable	No
MULTI-EXIT DISC	optional non-transitive	7	1	No
LOCAL_PREF	well-known discretionary	8	1	No
TRANSIT DELAY	well-known discretionary	9	2	Yes
RESIDUAL ERROR	well-known discretionary	10	4	Yes
EXPENSE	well-known discretionary	11	2	Yes
SOURCE SPECIFIC QOS	well-known discretionary	12	variable	Yes
DESTINATION SPECIFIC QOS	well known discretionary	13	variable	Yes
HIERARCHICAL RECORDING	well-known discretionary	14	1	No
RD_HOP_COUNT	well-known mandatory	15	1	No
SOURCE SPECIFIC SECURITY	well-known discretionary	16	variable	Yes
DESTINATION SPECIFIC SECURITY	well-known discretionary	17	variable	Yes
CAPACITY	well-known discretionary	18	1	Yes
PRIORITY	well-known discretionary	19	1	Yes

- The rules for attaching new non-transitive optional attributes depend on the nature of each specific attribute. The definition of each non-transitive optional attribute specifies such rules.
- Any optional attribute may be updated by any BIS in its path.

The order of appearance of attributes within the path information of a given UPDATE PDU is immaterial; however, the same attribute can appear only once within a given UPDATE PDU. Path attributes are summarized in Table 3; their encoding is described in clause 7.3.

8.11.2 Handling of Distinguishing Attributes

Certain well-known discretionary path attributes are classified as Distinguishing Attributes (see clause 6.7), which can be used to discriminate among multiple routes to a destination, based on differences in quality between the routes.

Distinguishing path attributes shall only be created by the BIS that originates the routing information; they can be updated by any BIS that receives an UPDATE

PDU that contains them. The rules for updating each of IDRP's distinguishing attributes are defined in the appropriate subclause of clause 8.12.

1 A permissible set of distinguishing attributes is
 1 defined to be a set that can be derived from informa-
 1 tion that can be validly encoded in the header of an
 1 ISO 8473 NPDU, using the mappings described in 9.2.
 1 In turn, a valid RIB-Att (see 6.7) is also a permissible
 1 set of distinguishing attributes, which is used to iden-
 1 tify the RIB that holds a route characterized by those
 1 distinguishing attributes. Therefore, a permissible set
 1 of distinguishing attributes and a corresponding valid
 1 RIB-Att:

- 1 a) Can consist of an empty set (that is, the Empty
 1 Distinguishing Attribute)
- 1 b) Can contain at most one of the following distin-
 1 guishing attributes: SOURCE SPECIFIC SECURITY
 1 or DESTINATION SPECIFIC SECURITY.

1 **NOTE:** These distinguishing attributes are derived from
 1 the ISO 8473 Security parameter, as described
 1 in 9.2, which can contain at most one of them.

- 1 c) Can contain at most one of the following attri-
 1 butes: RESIDUAL ERROR, TRANSIT DELAY,

1 EXPENSE, CAPACITY, SOURCE SPECIFIC QOS,
1 DESTINATION SPECIFIC QOS.

1 **NOTE:** These distinguishing attributes are derived from
1 the ISO 8473 Quality of Service Maintenance
1 parameter, as described in 9.2, which can
1 contain at most one of them.

1 d) Can include the Priority Distinguishing Attribute.

1 **NOTE:** This distinguished attribute is derived from the
1 ISO 8473 Priority parameter, as described in
1 9.2.

1 e) Can not include any instance of equivalent distin-
1 guishing attributes, as defined in 8.11.3.

1 Therefore, the number of distinguishing attributes that
1 can comprise either a valid RIB-Att or a permissible
1 set of distinguishing attributes is not unbounded: it is
1 limited to at most three.

8.11.3 Equivalent Distinguishing Attributes

IDRP recognizes two categories of distinguishing attribute: type specific, and type-value specific. Certain Distinguishing Attributes are unambiguous by their type—namely, Capacity, Priority, Transit Delay, Expense, and Residual Error. These are called *type specific*. Others can not be disambiguated based solely on their type, but require knowledge of both type and value—namely, Source Specific QOS, Destination Specific QOS, Source Specific Security, and Destination Specific Security. These are called *type-value specific*.

Within IDRP, two instances of Distinguishing Attributes are equivalent each other if either:

- a) they are both type specific and they both have the same type, or
- b) they are both type-value specific, and they both have the same type and the same value.

In all other cases two instances of Distinguishing Attribute are not equivalent.

8.12 Path Attribute Usage

The usage of each of IDRP's path attributes is described in the following clauses.

8.12.1 ROUTE-ID

2 ROUTE-ID is a well-known mandatory attribute that
2 contains an unambiguous 4-octet identifier for the
2 route being advertised in this UPDATE PDU. An
2 UPDATE PDU that advertises a feasible route shall
2 contain this path attribute. The identifier is assigned
2 by the BIS that advertises the route, and must be
2 unambiguous within the context of the BIS-BIS con-
2 nection.

2 The identifier associated with a route received from
2 an adjacent BIS and then subsequently advertised by
2 the local BIS are not related to one another.

2 NOTES:

- 2 a) Since the sequence number assigned to an UPDATE
2 PDU is unique within an instance of BIS-BIS communi-
2 cation, and is a 4-octet long quantity, a BIS could gen-
2 erate the Route-ID for a given route by choosing a value
2 that is numerically equal to the sequence number of the
2 UPDATE PDU that advertises the route.
- 2 b) Since the ROUTE-ID must be unambiguous within each
2 instance of BIS-BIS communication, it follows that a BIS
2 might advertise the same route to different neighbor
2 BISs, using different ROUTE-IDs in each instance of
2 BIS-BIS communications.
- 2 c) The ROUTE-IDs associated with routes to be advertised
2 by a BIS (that is, the routes in its Adj-RIBs-Out) bears no
2 relationship to the ROUTE-IDs associated with routes
2 received from other BISs (that is, the routes in the local
2 BIS's Adj-RIBs-In).

8.12.2 EXT_INFO

EXT_INFO is a well-known discretionary attribute. It shall be recognized upon receipt by all BISs. It shall be included in each UPDATE PDU that reports either an RD_PATH attribute or Network Layer Reachability Information that has been learned by methods not described in this international standard.⁸

The EXT_INFO attribute shall be generated by the RD that originates the associated routing information. If the EXT_INFO attribute was present in a received UPDATE PDU, then it shall also be included in the UPDATE PDUs of all BISs that choose to propagate this information to other BISs.

NOTE: If a BIS selects a route which has been advertised with the EXT_INFO attribute, it is possible that there may be undetected looping of routing information. Therefore, it is recommended that distribution of information not learned by the methods of IDRP be tightly controlled. The path attributes DIST_LIST_INCL and DIST_LIST_EXCL afford a convenient method for providing this control. Further-

⁸ For example, information obtained from the managed object **INTERNAL-SYSTEMS** or information obtained from UPDATE PDUs which do not contain the EXT_INFO attribute are learned by methods within IDRP's scope; however, manually configured reachability information about an RD which does not run IDRP is an example of information which is learned by means outside IDRP's scope.

more, a given RD may also enforce policies which prohibit any of its BISs from selecting routes which have the EXT_INFO attribute associated with them.

8.12.3 RD_PATH

RD_PATH is a well-known mandatory attribute. It shall be present in every UPDATE PDU, and shall be recognized upon receipt by all BISs. This attribute identifies the RDs and RDCs through which this UPDATE PDU has passed. The components of the list can be RD_SETs or RD_SEQUENCES.

When a BIS propagates a route which it has learned from another BIS's UPDATE PDU, it shall modify the route's RD_PATH attribute based on the location of the BIS to which the route will be sent:

- a) When a given BIS advertises the route to another BIS located in its own RD, then the advertising BIS shall not modify the RD_PATH attribute associated with the route.
- b) When a given BIS advertises the route to a BIS located in an adjacent RD, then the advertising BIS shall update the RD_PATH attribute as follows:

- 2 1) if the last path segment of the RD_PATH is of
2 type RD_SEQUENCE, the local BIS shall
2 append its own RDI as the last element of the
2 sequence
- 2 2) if the last path segment of the RD_PATH is of
2 type RD_SET, the local BIS shall either: a)
2 include its RDI in the set, or b) append a new
2 path segment of type RD_SEQUENCE to the
2 RD_PATH, including its own RDI in that
2 segment.

The BIS that modifies the RD_PATH attribute is free to collapse the previously received RD_PATH information in accordance with its local policies: for example, it may create an RD_SET from a received RD_SEQUENCE.

When a BIS originates a route (that is, the destinations are located in the local BIS's routing domain), then:

- a) the originating BIS shall include its own RDI in the RD_PATH attribute of all UPDATE PDUs sent to BISs located in adjacent RDs. (In this case, the RDI of the originating BIS's RD will be the only entry in the RD_PATH attribute.)
- b) the originating BIS shall include an empty RD_PATH attribute in all UPDATE PDUs sent to BISs located in its own RD. (An empty RD_PATH attribute is one whose length field contains the value zero.)

If the originating BIS also is a member of a Routing Domain Confederation, then it shall additionally

comply with the requirements of clause 8.13: in particular, RDs which are members of a confederation shall not be permitted to collapse RD_SEQUENCE information into an RD_SET when the original sequence contains an entry marker (see clause 8.13.4), nor shall they be permitted to generate information expressed as an RD_SET.

8.12.4 NEXT_HOP

NEXT_HOP is a well-known discretionary attribute. It shall be recognized upon receipt by all BISs.

For purposes of defining the usage rules for this attribute, a subnetwork is transitive with respect to system reachability if all of the following conditions are true:

- a) Systems A, B, and C are all attached to the same subnetwork,
- b) When A can reach B directly, and B can reach C directly, it follows that A can reach C directly.

Verification of the above conditions should be accomplished by means outside of IDRP. For example, systems located on a common subnetwork could use an ES-IS protocol (such as IS 9542) to ascertain if there is direct reachability between them. Examples of such media are IEEE 802.2 and SMDS.

Consider three BISs attached to a fully connected transitive subnetwork, as shown in Figure 8: A and B share a BIS-BIS connection, B and C share a BIS-BIS connection, but A and C have no BIS-BIS connection between themselves. If C propagates an UPDATE PDU to B, then with respect to the UPDATE PDU advertised by B:

- C is defined to be the *source BIS*
- B is defined to be the *first recipient BIS*
- A is defined to be the *subsequent recipient BIS*.

In terms of these definitions, the following rules apply to the usage of the NEXT_HOP attribute:

a) Generating the Attribute

When a given BIS generates an UPDATE PDU:

- 2 1) It may list its own NET and the SNPA's of sub-
2 networks that connect itself to the remote
2 BIS in the NEXT_HOP attribute of that
UPDATE PDU.
- 2 2) It may choose not to include a NEXT_HOP
attribute in its UPDATE PDU. When the
NEXT_HOP field is not present, it implies that
the NET of the BIS that advertises the
UPDATE PDU should be considered to be the
NET of the next-hop BIS.
- 3) It may set the value of the
"IDRP_Server_Allowed" field in accordance
with its local policies:

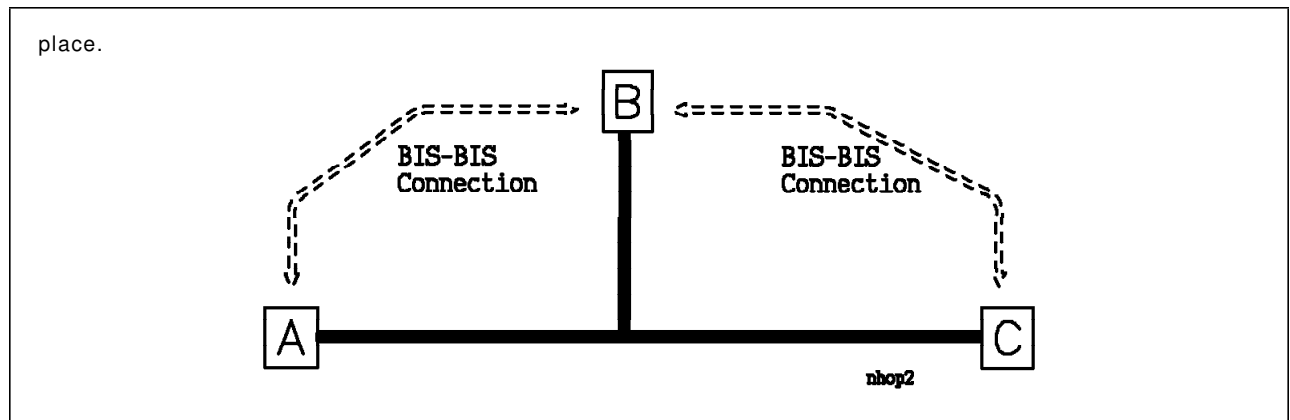


Figure 8. A Transitive Fully Connected Subnetwork

- If the source BIS wants to allow the first recipient BIS to advertise the source BIS's NET and SNPA to a subsequent recipient BIS, then it shall set this field to X'FF'
- If the source BIS does not want the first recipient BIS to advertise the source BIS's NET and SNPA, then it shall set this field to any value other than X'FF'.

b) Advertising Routeing Information

When a BIS chooses to advertise routeing information learned from an UPDATE PDU:

- 2 1) The BIS may choose to list its own NET and the SNPAs of subnetworks that connect itself to the remote BIS in the NEXT_HOP attribute of an UPDATE PDU that propagates the routeing information
- 2 2) The BIS may choose not to include a NEXT_HOP attribute in its UPDATE PDU. When the NEXT_HOP field is not present, it implies that the BIS that advertises the UPDATE PDU is also the next-hop BIS.
- 3) If any condition listed below is not satisfied, then the recipient BIS shall not list the NET and SNPAs of the source BIS in its own UPDATE PDUs. If they are all satisfied, then instead of listing its own NET and SNPAs, the BIS may optionally list the NET and SNPAs of the source BIS (as contained in the UPDATE PDU received from the source BIS) when it propagates the information to a subsequent recipient BIS. The conditions are the following:
 - i) The "IDRP_Server_Allowed" field of the UPDATE PDU of the source BIS was equal to X'FF'.
 - ii) All three BISs (source, first recipient, and subsequent recipient) are located on a common subnetwork which is full-duplex and is transitive with respect to reachability of all three BISs.

- iii) The managed object **Server** is "true".
- iv) The first recipient and subsequent recipient are located in different routeing domains.
- v) Advertisement of this route to the subsequent recipient BIS does not conflict with any of the path attributes that were contained in the UPDATE PDU from the source BIS. (For example, it may not propagate the UPDATE PDU to a recipient that is listed in the DIST_LIST_EXCL attribute.)

NOTES:

- a) The rules stated above do not remove the requirement that there must be a BIS-BIS connections between each pair of BISs located in the same routeing domain.
- b) The contents of the NEXT_HOP attribute have no effect upon the contents of the RD_PATH attribute: that is, the RD_PATH attribute will always be used in accordance with clause 8.12.3.
- c) If the NET and SNPAs are not available in an UPDATE PDU, then a BIS that receives it must learn them by means outside of this international standard. For example, the value of the NET can be learned from the NUNITDATA.INDICATION, and IS 9542 can be used to associate an SNPA with that NET.

8.12.5 DIST_LIST_INCL

DIST_LIST_INCL is a well-known discretionary attribute. It shall be recognized upon receipt by all BISs. When present, this attribute lists the RDIs of the routeing domains and confederations to which the routeing information may be distributed. Since NPDU's usually flow in a direction opposite to the flow of UPDATE PDU's, DIST_LIST_INCL provides a means for a given RD or confederation to control use of its resources by other RDs.

When a BIS receives an UPDATE PDU that contains the DIST_LIST_INCL attribute, then the receiving BIS shall not redistribute the associated routeing informa-

tion to any BIS which is not a member of at least one RD or RDC whose RDI is contained in this attribute.

A BIS shall redistribute only that information which has been locally selected as a route, and shall redistribute it only to RDs or RDCs which are both adjacent to it and are included in the distribution list. A BIS may distribute the information to any adjacent BIS that is a member of any routing domain or confederation whose RDI is contained in DIST_LIST_INCL. A BIS is not required to distribute the routing information to every RD or RDC whose RDI is listed: for example, it is possible for local policy considerations or the contents of the HIERARCHICAL_RECORDING path attribute to further restrict the set of RDIs to whom the routing information will actually be redistributed.

If a BIS receives an UPDATE PDU that contains neither the DIST_LIST_INCL nor DIST_LIST_EXCL attributes, then it may distribute the routing information to all adjacent BISs. Alternatively, the local BIS may also add a DIST_LIST_INCL or DIST_LIST_EXCL attribute, but not both, to the route information.

If the DIST_LIST_INCL attribute is present and has a length of zero octets, then the routing information may be used locally, but shall not be advertised to any other BIS.

When it originates an UPDATE PDU which describes a route to destinations located in its own routing domain, a BIS may append the DIST_LIST_INCL attribute, in accordance with its local policies.

If a BIS chooses to advertise a route which was learned from an UPDATE PDU which already contained the DIST_LIST_INCL attribute, the advertising BIS may modify this attribute by pruning the set of RDIs included in the list. If a BIS chooses to prune the set, it shall not delete the RDI of its own RD, nor shall it delete the RDI of any RDC to which it belongs. However, if the reduced list is empty (that is, has a length of zero), then the BIS shall not advertise the routing information to any BIS located in a different routing domain.

8.12.5.1 Interaction with HIERARCHICAL_RECORDING

When a given UPDATE PDU contains both the HIERARCHICAL_RECORDING attribute and the DIST_LIST_INCL attribute, the constraints imposed by the HIERARCHICAL_RECORDING, as specified in clause 8.12.14, shall take precedence over those imposed by DIST_LIST_INCL.

8.12.6 DIST_LIST_EXCL

DIST_LIST_EXCL is a well-known discretionary attribute. It shall be recognized upon receipt by all BISs. When present, this attribute lists the RDIs of routing domains and confederations to which the routing information may not be distributed. Since NPDU's usually flow in a direction opposite to the flow of UPDATE PDU's, DIST_LIST_EXCL provides a means for a given RD or confederation to control use of its resources by other RDs and RDCs.

When a BIS receives an UPDATE PDU that contains the DIST_LIST_EXCL attribute, then the receiving BIS shall not redistribute the associated routing information to any BIS located in an RD or RDC whose RDI is included in the list. A BIS shall not distribute the information to any adjacent BIS that is a member of any routing domain or confederation whose RDI is contained in DIST_LIST_EXCL. Local policy considerations shall not override redistribution of the routing information as dictated by the DIST_LIST_EXCL attribute.

If a BIS receives an UPDATE PDU that contains neither the DIST_LIST_INCL nor the DIST_LIST_EXCL attributes associated with it, then it may distribute the routing information to all adjacent BISs. Alternatively, the local BIS may also add a DIST_LIST_INCL or DIST_LIST_EXCL attribute, but not both, to the route information.

If the DIST_LIST_EXCL attribute is absent and the DIST_LIST_INCL attribute is present, then the distribution of the routing information is controlled by the DIST_LIST_INCL attribute. If the DIST_LIST_EXCL attribute is present and has a length of zero octets, then the routing information may, in accordance with local policy, be advertised to any other BIS.

When it originates an UPDATE PDU which describes a route to destinations located in its own routing domain, a BIS may append the DIST_LIST_EXCL attribute, in accordance with its local policies.

If a BIS chooses to advertise a route which was learned from an UPDATE PDU which already contained the DIST_LIST_EXCL attribute, the advertising BIS may modify this attribute by augmenting the set of RDIs included in the list. If a BIS chooses to augment the set, it shall not add the RDI of its own RD, nor shall it add the RDI of any RDC to which it belongs.

8.12.6.1 Interaction with HIERARCHICAL_RECORDING

When a given UPDATE PDU contains both the HIERARCHICAL_RECORDING attribute and the DIST_LIST_EXCL attribute, the constraints imposed by DIST_LIST_EXCL, as specified in clause 8.12.6, shall take precedence over those imposed by the HIERARCHICAL_RECORDING attribute.

8.12.7 MULTI-EXIT_DISC

MULTI-EXIT_DISC is an optional non-transitive attribute. If the local BIS's managed object **Multiexit** is "true", the BIS may use the attribute in its path selection algorithm. For example, a routing domain may choose to implement a policy which mandates that if all other path attributes are equal, the exit point with the lowest value of MULTI-EXIT_DISC should be preferred.

Each BIS that is connected to an adjacent RD by one or more common subnetworks may generate a MULTI-EXIT_DISC attribute for each link connecting itself to an adjacent RD. The value of this attribute is a local matter, which will be determined by the policies of the RD in which the originating BIS is located.

A BIS that generates a value for this attribute may distribute it to all neighboring BISs which are located in adjacent RDs.

If a MULTI-EXIT_DISC attribute is received from a BIS located in an adjacent RD, then the receiving BIS may distribute this attribute to all other BISs located in its own RD. However, the receiving BIS shall not redistribute the attribute to any BISs which are not located within its own RD.

8.12.8 LOC_PREF

LOCAL_PREF is a well-known discretionary attribute that shall be included in all UPDATE PDUs that a given BIS sends to the other BISs located in its own routing domain. Its principal use is to permit the detection of inconsistent routing decisions among a set of BISs that are all located in the same routing domain, as described in clause 8.14.2.

A BIS shall not include this attribute in UPDATE PDUs that it sends to BISs located in an adjacent routing domain. If it is contained in an UPDATE PDU that is received from a BIS which is not located in the same routing domain as the receiving BIS, then this attribute shall be ignored by the receiving BIS.

8.12.9 TRANSIT DELAY

TRANSIT DELAY is a well-known discretionary attribute. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports routing based on the Transit Delay attribute, and that it maintains Adj-RIBs and a Loc-RIB distinguished by this attribute.

The average transit delay associated with the local RD is obtained from the managed object **RDTransitDelay** and represents an average transit delay that would be experienced by SNSDU size of 512 octets while traversing the RD. **RDTransitDelay** is specified in units of 500 ms.

If A BIS advertises a route whose destinations are located in its own RD, then the originating BIS may append the Transit Delay attribute to the route, using the value contained in managed object **RDTransitDelay**.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Transit Delay attribute, it shall update the value of this attribute before advertising the route to a BIS located in another routing domain. The updated value shall be computed by adding the value in **RDTransitDelay** to the value of the parameter that was received in the UPDATE PDU's Transit Delay attribute.

If the route is re-distributed to another BIS located in the same RD as the advertising BIS, then the Transit Delay attribute shall not be modified, but shall be distributed with the same value that was present in the received UPDATE PDU.

8.12.10 RESIDUAL ERROR

RESIDUAL ERROR is a well-known discretionary attribute. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports routing based on the Residual Error attribute, and that it maintains Adj-RIBs and a Loc-RIB distinguished by this attribute.

The value contained in the RESIDUAL ERROR attribute, in *RRE*, and in managed object **RDRE** is a positive integer in the range from 0 to $2^{32} - 1$; the actual probability of error can be obtained by dividing the value by $2^{32} - 1$.

If A BIS advertises a route whose destinations are located in its own RD, then the originating BIS may append the RESIDUAL ERROR attribute to the route, using the value contained in managed object **RDRE**. The value of the **RDRE** is an integer value derived from the average ratio of lost, duplicated, or incorrectly delivered SNSDU's to total SNSDUs transmitted by the SNDCF during a measurement period: this ratio is multiplied by $2^{32} - 1$, and then rounded up to the next higher integer.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the RESIDUAL ERROR attribute, it shall update the value of this attribute before advertising the route to a BIS located in another routing domain. The updated value is computed by the following formula:

$$K \times (1 - [(1 - RRE/K) \times (1 - RDLRE/K)])$$

where *RRE* is the value of the RESIDUAL ERROR attribute of the received route, *RDRE* is the value of the average residual error probability associated with the local RD, *K* is the constant $2^{32} - 1$, and the whole expression is rounded up to the nearest integer.

If the route is re-distributed to another BIS located in the same RD as the advertising BIS, then the RESIDUAL ERROR attribute shall not be modified, but shall be distributed with the same value that was present in the received UPDATE PDU.

8.12.11 EXPENSE

EXPENSE is a well-known discretionary attribute. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports routeing based on the Expense attribute, and that it maintains Adj-RIBs and a Loc-RIB distinguished by this attribute. The value of Expense associated with a given routeing domain is contained in managed object **LocExpense**. It is related to monetary cost. Different routeing domains may use different values for this attribute: thus, the attribute must deal in relative monetary costs.

If A BIS advertises a route whose destinations are located in its own RD, then the originating BIS may append the Expense attribute to the route, using the value contained in managed object **LocExpense**.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Expense attribute, it shall update the value of this attribute before advertising the route to a BIS located in another routeing domain. The updated value is computed by adding the value of the expense contained in managed object **LocExpense** to the value of the EXPENSE attribute of the received route.

If the route is re-distributed to another BIS located in the same RD as the advertising BIS, then the Residual Error attribute shall not be modified, but shall be distributed with the same value that was present in the received UPDATE PDU.

8.12.12 SOURCE SPECIFIC QOS

SOURCE SPECIFIC QOS is an well-known discretionary attribute that allows a BIS to control the ability of ISO 8473 NPDUs with a given source NSAP address to transit through its local RD. The finest granularity of this control is a single End system. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports source specific routeing based on the reported QOS value, and that it maintains Adj-RIBs and a Loc-RIB distinguished by the indicated source specific QOS value.

In accordance with clause 7.5.6.1 of ISO 8473, the value of the QOS Value field of the SOURCE SPECIFIC QOS field shall be unique and unambiguous in the context of the QOS maintenance system employed by the authority responsible for assigning the source NSAP address.

NOTE: Methods for the assignment of unambiguous and unique QOS Values throughout the global OSIE are outside the scope of IDRP.

This attribute may be originated only by those BISs that reside in the same RD as the destination End system(s) reported in the NLRI of the UPDATE PDU.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Source Specific QOS attribute, the new UPDATE PDU shall contain the Source Specific QOS attribute which shall have the same QOS value and address fields that are associated with the route, and the metric field (if present) shall be modified in the following way:

- a) when the route is advertised to a BIS located in the same RD as the advertising BIS, the metric value shall not be modified
- b) when the route is advertised to a BIS located in a different RD from the advertising BIS, the metric value shall be modified according to the rules for that metric. Rules for modifying a given metric value field are defined by the authority responsible for assigning the addresses contained in the "address" field of this attribute; such rules are not within the scope of IDRP.

8.12.13 DESTINATION SPECIFIC QOS

DESTINATION SPECIFIC QOS is an well-known discretionary attribute that allows a BIS to control the ability of ISO 8473 NPDUs with a given destination NSAP address to transit through its local RD. The finest granularity of this control is a single End system. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports destination specific routeing based on the reported QOS value, and that it maintains Adj-RIBs and a Loc-RIB distinguished by the indicated destination specific QOS value.

In accordance with clause 7.5.6.2 of ISO 8473, the value of the QOS Value field of the DESTINATION SPECIFIC QOS field shall be unique and unambiguous in the context of the QOS maintenance system employed by the authority responsible for assigning the destination NSAP address.

NOTE: Methods for the assignment of unambiguous and unique QOS Values throughout the global OSIE are outside the scope of IDRP.

This attribute may be originated only by those BISs that reside in the same RD as the destination End system(s) reported in the NLRI of the UPDATE PDU.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Destination Specific QOS attribute, the new UPDATE PDU shall contain the Destination Specific QOS attribute which shall have the same QOS value and address fields that are associated with the route, and the metric field (if present) shall be modified in the following way:

- a) when the route is advertised to a BIS located in the same RD as the advertising BIS, the metric value shall not be modified
- b) when the route is advertised to a BIS located in a different RD from the advertising BIS, the metric value shall be modified according to the rules for that metric. Rules for modifying a given metric value field are defined by the authority responsible for assigning the addresses contained in the "address" field of this attribute; such rules are not within the scope of IDRP.

8.12.14 HIERARCHICAL RECORDING

The HIERARCHICAL_RECORDING attribute provides an optional means for members of a routing domain confederation to control transitivity. The transitivity constraints are based upon two factors:

- a) the value of the HIERARCHICAL ATTRIBUTE that was contained in a received UPDATE PDU
- b) knowledge of whether it is necessary to enter or exit a confederation in order to reach an adjacent RD.

If an RD wishes to support this attribute, then it shall obey the following usage rules:

- a) *Destination BIS in a Disjoint RDC:*

The HIERARCHICAL_RECORDING attribute shall not be included in an UPDATE PDU that is transmitted to a BIS that can be reached only by exiting all of the confederations in which the advertising BIS resides.

- b) *Destination BIS in Same, Nested, or Overlapping RDC:*

- 1) If a given BIS chooses to advertise routing information that it learned from an inbound UPDATE PDU whose HIERARCHICAL_RECORDING attribute was equal to 1, or if it is the originator of the routing information, it may advertise this information to BISs located in any adjacent routing domain, as follows:

- i) If it is necessary to enter a confederation in order to reach the destination BIS, then the advertising BIS shall include the HIERARCHICAL_RECORDING attribute in its outbound UPDATE PDU, and shall set its value to 0.
- ii) If the destination BIS can be reached without entering any confederation, then the advertising BIS shall include the HIERARCHICAL_RECORDING attribute in its outbound UPDATE PDU, and shall set its value to 1.

- 2) If a given BIS chooses to advertise routing information that it learned from an inbound

UPDATE PDU whose HIERARCHICAL_RECORDING attribute was equal to 0, it may advertise that information only to BISs that can be reached without exiting any confederation to which the advertising BIS belongs. The HIERARCHICAL_RECORDING attribute shall be included in the outbound UPDATE PDU, and its value shall be set to 0.

8.12.14.1 Interaction with DIST_LIST_INCL and DIST_LIST_EXCL

When a given UPDATE PDU contains both the HIERARCHICAL_RECORDING attribute and the DIST_LIST_EXCL attribute, the constraints imposed by DIST_LIST_EXCL, as specified in clause 8.12.6, shall take precedence over those imposed by the HIERARCHICAL_RECORDING attribute.

When a given UPDATE PDU contains both the HIERARCHICAL_RECORDING attribute and the DIST_LIST_INCL attribute, the constraints imposed by the HIERARCHICAL_RECORDING, as specified in clause 8.12.14, shall take precedence over those imposed by DIST_LIST_INCL.

8.12.15 RD_HOP_COUNT

This is a well-known mandatory attribute whose usage is as follows:

- a) The initial value of this attribute is 0.
- b) Before sending an UPDATE PDU to a BIS located in an adjacent routing domain, a BIS shall increment the value of this attribute by 1, and shall place the result in the RD_HOP_COUNT field of the outbound UPDATE PDU.
- c) A BIS shall not increment the value of this attribute when it sends an UPDATE PDU to another BIS located in its own routing domain.

NOTE: ISO 8473 limits the maximum lifetime of an NPDU to 256 counts, and requires each Network entity processing a given NPDU to decrement that NPDU's lifetime by at least 1 count. In the limiting case of one BIS per routing domain, this implies that a NPDU's lifetime will expire before it can reach the 257th RD. Hence, there is no need to provide an RD_HOP_COUNT greater than 256.

8.12.16 SOURCE SPECIFIC SECURITY

SOURCE SPECIFIC SECURITY is a well-known discretionary attribute that allows a BIS to specify the security level that is associated with a given path, and to limit usage of that path to systems having the NSAP address prefixes listed in this attribute. The finest granularity of this control is a single End system. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports the source specific security level reported in the security level field, and that it

maintains Adj-RIBs and a Loc-RIB distinguished by the indicated source specific security level.

In accordance with clause 7.5.3.1 of ISO 8473, the value of the security level field shall be unique and unambiguous in the context of the security classification system employed by the authority responsible for assigning the source NSAP address.

NOTE: Methods for the assignment of unambiguous and unique security levels throughout the global OSIE are outside the scope of IDRP.

This attribute may be originated only by those BISs that reside in the same RD as the destination End system(s) reported in the NLRI of the UPDATE PDU.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Source Specific Security attribute, the new UPDATE PDU shall contain the Source Specific Security attribute which shall have the same security level and address fields that are associated with the route.

8.12.17 DESTINATION SPECIFIC SECURITY

DESTINATION SPECIFIC SECURITY is a well-known discretionary attribute that allows a BIS to specify the security level that is associated with a given path, and to limit usage of that path to systems having the NSAP address prefixes listed in this attribute. The finest granularity of this control is a single End system. A BIS shall include this attribute in its UPDATE PDU to indicate that it supports the destination specific security level reported in the security level field, and that it maintains Adj-RIBs and a Loc-RIB distinguished by the indicated destination specific security level.

In accordance with clause 7.5.3.2 of ISO 8473, the value of the security level field shall be unique and unambiguous in the context of the security classification system employed by the authority responsible for assigning the destination NSAP address.

NOTE: Methods for the assignment of unambiguous and unique security levels throughout the global OSIE are outside the scope of IDRP.

This attribute may be originated only by those BISs that reside in the same RD as the destination End system(s) reported in the NLRI of the UPDATE PDU.

When a BIS re-distributes a route which has been learned in an UPDATE PDU that contains the Destination Specific Security attribute, the new UPDATE PDU shall contain the Destination Specific Security attribute which shall have the same security level and address fields that are associated with the route.

8.12.18 CAPACITY

This is a well-known discretionary attribute that is used to denote the traffic handling capacity of the RD_PATH listed in the same UPDATE PDU. It is a distinguishing attribute.

The value of capacity that is associated with a given routing domain is contained in managed object **Capacity**.

If a BIS advertises a route whose destinations are located in its own routing domain, then the originating BIS may include this attribute in its outbound UPDATE PDUs; if present, its value shall be equal to that of managed object **Capacity**.

If a BIS redistributes a route that was advertised with the CAPACITY attribute present, it shall include the CAPACITY attribute in its outbound UPDATE PDU, and shall set its value equal to the higher of the following two quantities: the value of the CAPACITY attribute contained in the UPDATE PDU that advertised the route, or the value of local managed object **Capacity**.

8.12.19 PRIORITY

This well-known discretionary attribute is a distinguishing attribute, used to indicate the minimum priority value that the BIS will support. It is an unsigned integer in the range from 0 to 14, with higher priority indicated by higher values.

The value of this parameter is the same for all BISs in a given routing domain, and is equal to the value contained in managed object **Priority**.

If a BIS originates a route to destinations located in its own routing domain, then the originating BIS may include this attribute in its outbound UPDATE PDUs; if present, its value shall be equal to that of managed object **Priority**.

- 2 If a BIS redistributes a route that was advertised with the PRIORITY attribute present, it shall include the PRIORITY attribute in its outbound UPDATE PDU, and shall set its value equal to the higher of the following two quantities: the value of the PRIORITY attribute contained in the UPDATE PDU that advertised the route, or the value of local managed object **Priority**.

8.13 Routeing Domain Confederations

Formation of an RDC is done via a private arrangement between its members without any need for global coordination; the methods for doing so are not within the scope of this international standard.

- 2 From the outside, an RDC looks similar to a single routing domain: for example, it has an identifier which is an RDI. Other RDs can develop policies with

respect the confederation as a whole, as opposed to the individual RDs that are members of the confederation. Confederations can be disjoint, nested, or overlapping (see clause 4.6).

8.13.1 RDC Policies

Each RD within a confederation may have its own set of policies; that is, different RDs in the same confederation can have different policies. For BISs located within the same RD, the methods of clauses 8.14.2 will detect both internal and external routing inconsistencies.

Since a confederation appears to the external world as if it were an individual RD, IDRP's loop detection methods will detect routing information loops through a given confederation. In particular, a route which leaves the confederation and then later re-enters it will be detected as a loop: thus, a route between two RDs that are members of the same confederation will be constrained to remain within that confederation.

8.13.2 RDC Configuration Information

Each BIS that participates in one or more RDCs must be aware of the RDIs of all confederations of which it is a member, and it must know the partial order which prevails between these confederations: that is, it must know the nesting and overlap relationships between all confederations to which it belongs. This information shall be contained in managed object

RDC-Config, which consists of a list of confederation RDIs and the partial order that prevails among those confederations.

Since RDCs are formed via private arrangement between their members, the partial order of a given confederation is a local matter for that confederation, and bears no relationship to the partial orders that may prevail in different confederations. The RDI of its own routing domain is contained in managed object **LOCAL-RDI**, as defined in clause 8.3.

8.13.3 Detecting Confederation Boundaries

A given BIS can tell which confederations are common to itself and an adjacent BIS by comparing information obtained from the *Confed-IDs* field of the adjacent BIS's OPEN PDU with the local BIS's **RDC-Config** managed object. This knowledge determines when an outbound UPDATE PDU exits a given confederation and when an inbound UPDATE PDU enters a given confederation:

Exiting a Confederation: An UPDATE PDU sent by a given BIS to an adjacent BIS is defined to have exited all those confederations whose RDIs are present in the advertising BIS's **RDC-Config**

managed object but were not reported in the *Confed-IDs* field of the adjacent BIS's OPEN PDU.

Entering a Confederation: An UPDATE PDU received from an adjacent BIS is defined to have entered all those confederations whose RDIs are present in the receiving BIS's **RDC-Config** managed object but were not reported in the *Confed-IDs* field of the sending BIS's OPEN PDU.

8.13.4 Entry Marker for an RDC

All UPDATE PDUs contain the RD_PATH attribute, which reports the RDIs of the routing domains and confederations through which the UPDATE PDU has travelled. To routing domains are outside a given confederation, the RDI of a confederation is indistinguishable from the RDI of a routing domain. However, a marker is needed for the benefit of the members of a given RDC to denote that the UPDATE PDU has either entered or exited the confederation. Knowledge of entry or exit is only needed by confederation members, and therefore the marker for a given confederation shall only be present in the RD_PATH attribute while the UPDATE PDU is travelling within that confederation.

The Entry Marker shall be encoded in the RD_PATH attribute as an RDI of length 0. To denote that an UPDATE PDU has entered one or more confederations, the receiving BIS includes in its outbound UPDATE PDUs an Entry Marker followed by the RDI(s) of the entered confederation(s).

8.13.5 Generating RD_PATH Information

When a BIS in a given RDC generates information about destinations contained within itself, it shall examine the information contained in its managed object **RDC-Config** in order to determine the ordering relationship(s) that exist among all the RDCs to which it belongs. It shall then construct an RD_PATH attribute in the form of an RD_SEQUENCE, as follows:

- a) For each RDC to which it belongs, the BIS shall attach the 2-tuple <Entry Marker><RDI of the RDC>.
- b) If a confederation, RDC-A, is nested within another confederation, RDC-B, then the 2-tuple for RDC-B shall precede the 2-tuple for RDC-A.
- c) The 2-tuples of overlapping confederations may be listed in any order, as long as the order implied by any nesting relationships is maintained.
- d) After the 2-tuples of all its confederations are listed, the BIS shall append the RDI of its own routing domain.

8.13.6 Updating Received RD_PATH Information

When a BIS in a given RDC receives an UPDATE PDU from another BIS, it shall update the RD_PATH attribute of any UPDATE PDUs that it subsequently transmits according to the following rules:

- a) If the UPDATE PDU was received from a BIS in an adjacent RD and is being advertised to a BIS located in the RD of the advertising BIS, then the RD_PATH attribute of the advertised UPDATE PDU shall be updated by adding the following 2-tuple for each RDC that the inbound UPDATE PDU has entered: <Entry Marker><RDI of entered RDC>.

The 2-tuple of each RDC that is entered shall be listed according to local partial order that prevails among them, as contained in managed object **RDC-Config**. That is, if RDC-A is nested within RDC-B, then the 2-tuple for RDC-B shall precede the 2-tuple of RDC-A. The 2-tuples of overlapping RDCs may be listed in any order, as long reporting order implied by any nesting relationships is maintained.

- b) If the UPDATE PDU was received from a BIS in its own RD and is being advertised to a BIS in an adjacent RD, then the BIS shall use the methods of clause 8.13.3 to establish which confederations the new UPDATE PDU will exit. For each such confederation, the advertising BIS shall scan from the back of the previous RD_PATH attribute, looking for the occurrence of the 2-tuple corresponding to the RDC that is being exited. The BIS shall apply the following procedures to each of the exited confederations. The procedures shall be applied first to any exited confederation that is located at the lowest level of the local partial order, as described in managed object **RDC-Config**. The procedures shall then be applied iteratively to the other exited confederations, always working from those at the lowest level to those at the higher levels.

For a given exited confederation (for example, the confederation whose RDI is "X"), the advertising BIS shall then update the RD_PATH in the PDU to be advertised, as follows:

- 1) If the 2-tuple <Entry Marker><X> is not found, the route is in error, and the BIS shall issue a IDR_P ERROR PDU which reports a Misconfigured_RDCs error.
- 2) If the 2-tuple is found and there are no other intervening 2-tuples, then the RD_PATH from the 2-tuple to the end of the attribute shall be replaced by the RDI of the exited confederation.
- 3) If an intervening 2-tuple (for example, <Entry Marker><RDC-Y>) is found, and RDC-X is known to be nested within RDC-Y, then:

- if there are no RDIs listed between the 2-tuple for RDC-X and the 2-tuple for RDC-Y, then RDC-Y and RDC-X were entered simultaneously. In this case, the contents of the RD_PATH attribute from the end back to the 2-tuple for RDC-X shall be replaced by the sequence <Entry Marker><RDC-Y>.

- if there are any RDIs listed between the 2-tuple for RDC-X and the 2-tuple for RDC-Y, this indicates that RDC-X is not in fact nested within RDC-Y, and the route is in error. The local BIS shall send an IDR_P ERROR PDU that reports a Misconfigured_RDCs error.

- 4) If a 2-tuple for RDC-Z is found, but the local BIS's routing domain is not a member of RDC-Z, then the route is in error, and the local BIS shall send a IDR_P ERROR PDU that reports a Misconfigured_RDCs error.

- 5) If a 2-tuple for RDC-Z is found, where RDC-X and RDC-Z are overlapping confederations, the portion of the route between RDC-X and RDC-Z shall be replaced by the sequence <RDC-X><Entry Marker><RDC-Z>.

- c) If the UPDATE PDU was received from a BIS in an adjacent RD and is being advertised to a BIS in an adjacent RD, then the local BIS shall update the RD_PATH attribute by appending the 2-tuples of each confederation of which it is a member, according to the partial order that prevails among them. That is, if RDC-A is nested within RDC-B, then the 2-tuple for RDC-B shall precede the 2-tuple of RDC-A. The 2-tuples of overlapping RDCs may be listed in any order, as long reporting order implied by any nesting relationships is maintained.

8.14 Update-Receive Process

The Update-Receive process is active while the BIS is in the ESTABLISHED state; it accepts incoming BISPDU's as its input and then takes the following actions:

- 2 a) If a RIB REFRESH PDU is received, the BIS shall
- 2 take the actions described in 8.10.3.
- 2 b) If a KEEPALIVE PDU is received, the BIS shall
- 2 take the actions dictated by its Finite State
- 2 Machine, as described in the appropriate sub-
- 2 clause of 8.6.
- 2 c) If a CEASE PDU or an IDR_P ERROR PDU is
- 2 received, the BIS shall close the connection with
- 2 the peer BIS, as described in 8.6.2. As part of
- 2 these procedures, the BIS will deallocate all
- 2 resources that were associated with the remote
- 2 BIS. In particular, deallocation of the associated
- 2 Adj-RIB-In implicitly marks all routes previously

- 2 advertised by the peer BIS as unfeasible, and
2 requires the BIS to rerun its Decision Process.
- 2 d) If an UPDATE PDU is received, the BIS shall
2 update the appropriate Adj-RIB-In. For a feasible
2 route, the Adj-RIB-In is identified by its RIB-Att;
2 for an unfeasible route, the Adj-RIB-In is identi-
2 fied by the ROUTE-ID of the withdrawn route that
2 is carried in the WITHDRAWN ROUTES field of the
2 UPDATE PDU. The actions to be taken are:
- 2 1) If the UPDATE PDU contains a non-empty
2 WITHDRAWN ROUTES field, the previously
2 advertised feasible routes associated with
2 the ROUTE-IDs contained in this field shall be
2 removed from the Adj-RIB-In. The BIS shall
2 run its Decision Process since the previously
2 advertised route is no longer available for
2 use.
- 1 2) If the UPDATE PDU contains a feasible route,
1 it shall be placed in the appropriate
1 Adj-RIB-In, and the following additional
1 actions shall be taken:
- 1 i) If its NLRI and distinguishing attributes
1 are identical to those of a route currently
1 stored in the Adj-RIB-In, then the new
1 route shall replace the older route in the
1 Adj-RIB-In, thus implicitly withdrawing
1 the older route from service. The BIS
1 shall run its Decision Process since the
1 older route is no longer available for use.
- 1 ii) If the new route is an overlapping route
1 that is more specific (see 8.15.3.1) than
1 an earlier route contained in the
1 Adj-RIB-In, the BIS shall run its Decision
1 Process since the more specific route
1 has implicitly made a portion of the less
1 specific route unavailable for use.
- 1 iii) If the new route has identical path attri-
1 butes (both distinguishing and non-
1 distinguishing) to an earlier route
1 contained in the Adj-RIB-In, and is more
1 specific (see 8.15.3.1) than the earlier
1 route, no further actions are necessary.

8.14.1 Information Consistency

Correct operation of this protocol requires that all BISs within a routing domain apply a consistent set of policies for calculating the degree of preference for an RD-path. An internal inconsistency can occur when there is more than a single path to a particular destination. The hop-by-hop routing methodology then requires a BIS to select only one of these paths for each distinguishable QOS category. Internal inconsistency arises if different BISs within the same routing domain make different selections when presented with exactly the same set of routing information.

8.14.2 Detecting Inconsistencies

The Update-Receive and Update-Send processes of each BIS shall support the LOCAL_PREF Attribute, which is a well-known discretionary attribute. Its value for a particular route is determined by the degree of preference computed by the local BIS for that route.

The following procedures shall be applied separately for each set of Distinguishing Attributes supported by the BIS.

- 2 A BIS shall detect inconsistent routing decisions
2 (and, therefore, internal inconsistencies) by calcu-
2 lating a degree of preference for each route carried in
2 an UPDATE PDU that it receives as part of the
2 internal update process (see 8.16.1). If the degree of
2 preference calculated by the local BIS is different
2 from the value of the LOCAL_PREF attribute carried in
2 the UPDATE PDU, the routing policies of the two
2 BISs have internal inconsistencies. The BIS that
2 detects the inconsistency shall report it to system
2 management.

8.15 Decision Process

The Decision Process selects routes for subsequent advertisement by applying the policies in its Policy Information Base to the routes stored in its Adj-RIBs-In. The output of the Decision Process is the set of routes that will be advertised to adjacent BISs; the selected routes will be stored in the local BIS's Adj-RIBs-Out.

- The selection process is formalized by defining a function that takes the attributes of a given route as an argument and returns a non-negative integer denoting the degree of preference for the route. The function that calculates the degree of preference for a given route shall not use as its inputs any of the following:
- 1 the existence of other routes, the non-existence of
 - 1 other routes, or the path attributes of other routes.
 - 1 Route selection then consists of individual application
 - 1 of the degree of preference function to each feasible
 - 1 route, followed by the choice of the one with the
 - 1 highest degree of preference.

IDRP does not a universally agreed-upon metric to exist between multiple RDs. Instead, IDRP allows each RD to apply its own set of criteria for route selection, as determined by its local PIB. An example of the syntax and semantics of a routing policy that could be used to calculate a degree of preference is described in informative Appendix G.

- 2 The Decision process operates on routes contained in
2 each Adj-RIB-In, and is responsible for:
- 2 — selection of routes to be advertised to BISs
 - 2 located in local BIS's routing domain

- 2 — selection of routes to be advertised to BISs
- 2 located in adjacent routeing domains
- 2 — route aggregation and route information
- 2 reduction

2 The Decision process takes place in three distinct

2 phases, each triggered by a different event:

- 2 a) Phase 1 is responsible for calculating the degree
- 2 of preference for each route received from a BIS
- 2 located in an adjacent routeing domain, and for
- 2 advertising to the other BISs in the local Routing
- 2 Domain the routes that have the highest degree
- 2 of preference for each distinct destination.
- 2 b) Phase 2 is invoked on completion of Phase 1. It is
- 2 responsible for choosing the best route out of all
- 2 those available for each distinct destination, and
- 2 for installing each chosen route into the appro-
- 2 priate Loc-RIB.
- 2 c) Phase 3 is invoked after the Loc-RIB has been
- 2 modified. It is responsible for disseminating
- 2 routes in the Loc-RIB to each adjacent BIS
- 2 located in an adjacent routeing domain, according
- 2 to the policies contained in the PIB. Route aggre-
- 2 gation, information reduction and the modification
- 2 of QOS path attributes can optionally be per-
- 2 formed within this phase.

2 8.15.1 Phase 1: Calculation of Degree of Preference

2 The Phase 1 decision function shall be invoked when-

2 ever the local BIS receives an UPDATE PDU from a

2 neighbor BIS located in an adjacent routeing domain

2 that advertises a new route, a replacement route, or a

2 withdrawn route.

2 The Phase 1 decision function is a separate process

2 which completes when it has no further work to do.

2 The Phase 1 decision function shall be blocked from

2 running while the Phase 2 decision function for the

2 same RIB-Att is in process.

2 The Phase 1 decision function shall lock an Adj-RIB-In

2 prior to operating on any route contained within it,

2 and shall unlock it after operating on all new or unfea-

2 sible routes contained within it.

2 For each newly received or replacement feasible

2 route, the local BIS shall compute a degree of prefer-

2 ence. Then, the local BIS shall run the internal

2 update process of 8.16.1 to select and advertise the

2 most preferable routes.

2 A route received from another BIS in the local

2 Routing should have a LOCAL_PREF path attribute. If

2 the value of the LOCAL_PREF path attribute differs

2 from the degree of preference computed above, then

2 an internal inconsistency has occurred, and the local

2 BIS shall report it to system management. If the

2 LOCAL_PREF attribute is not present in the received

- 2 UPDATE PDU, the local BIS shall issue an IDRP
- 2 ERROR PDU, reporting Missing_Well-known_Attribute.

2 8.15.2 Phase 2: Route Selection

2 The Phase 2 decision function shall be invoked on

2 completion of Phase 1. The Phase 2 function is a sep-

2 arate process which completes when it has no further

2 work to do. The Phase 2 process shall consider all

2 routes that are present in the Adj-RIBs-In, including

2 those received from BISs located in its own routeing

2 domain and those received from BISs located in adja-

2 cent routeing domains.

2 The Phase 2 decision function shall be blocked from

2 running while the Phase 3 decision function is in

2 process. The Phase 2 function shall lock all

2 Adj-RIBs-In with the RIB- Att associated with this

2 instance of the process prior to commencing its func-

2 tion, and shall unlock them on completion.

2 For each set of destinations for which a feasible route

2 exists in the Adj-RIBs-In identified by the RIB-Att on

2 which this instance of the function operates, the local

2 BIS shall identify the route that has:

- 2 a) the highest degree of preference of any route to
- 2 the same set of destinations, or
- 2 b) is the only route to that destination, or
- 2 c) is selected as a result of the Phase 2 tie breaking
- 2 rules specified in 8.15.2.1.

2 The local BIS shall then install that route in the

2 Loc-RIB, replacing any route to the same destination

2 that is currently held in the Loc-RIB.

2 If a route copied to a Loc-RIB does not have a

2 NEXT_HOP path attribute, then the local BIS shall add

2 that attribute to the entry in the Loc-RIB. The value of

2 the attribute shall be the NET of the adjacent BIS from

2 which the route was received.

2 Unfeasible routes shall be removed from the Loc-RIB,

2 and corresponding unfeasible routes shall then be

2 removed from the Adj-RIBs-In.

1 **NOTE:** The decision process should not select a route to

1 destinations located within the local routeing domain

1 if that route would exit the local routeing domain and

1 later re-enter it. Such routes would be rejected by

1 other RDs due to the existence of an RD-loop. Fur-

1 thermore, the IDRP CLNS Forwarding Process will

1 not forward NPDUs (destined to internal destinations)

1 outside of the local RD, but will instead hand them

1 over to the intra-domain routeing protocol.

1 8.15.2.1 Breaking Ties (Phase 2)

1 In its Adj-RIBs-In, a BIS may have several routes to

1 the same destination that have the same degree of

1 preference and also have an equivalent set of distin-

1 guishing attributes. The local BIS can select only one

1 of these routes for inclusion in the associated

1 Loc-RIB. The local BIS considers all equally prefer-
 1 able routes, both those received from BISs located in
 1 adjacent RDs and those received from other BISs
 1 located in the local BIS's own RD.

1 Ties shall be broken according to the following rules:

1 a) If the candidate routes have identical path attri-
 1 butes or differ only in the NEXT_HOP attribute,
 1 select the route that was advertised by the BIS in
 1 an adjacent routeing domain whose NET has the
 1 lowest value. If none of the candidate routes
 1 were received from a BIS located in an adjacent
 1 routeing domain, select the route that was adver-
 1 tised by the BIS in the local routeing domain
 1 whose NET has the lowest value.

1 b) If the candidate routes differ only in their
 1 NEXT_HOP and MULTI-EXIT_DISC attributes, and
 1 the local BIS's managed object **ultiexit** is TRUE,
 1 select the route that has the lowest value of the
 1 MULTI-EXIT_DISC attribute.

1 If the managed object **Multiexit** is false, select the
 1 route advertised by the BIS in an adjacent RD
 1 whose NET has the lowest value. If none of the
 1 candidate routes were received from a BIS
 1 located in an adjacent routeing domain, select the
 1 route that was advertised by the BIS in the local
 1 routeing domain whose NET has the lowest value.

1 c) If the candidate routes differ in any path attri-
 1 butes other than NEXT_HOP and
 1 MULTI-EXIT_DISC, select the route that was
 1 advertised by the BIS whose NET has the lowest
 1 value.

1 For purposes of determining the lowest-valued NET,
 1 each binary-encoded NET shall be padded with trailing
 1 0's in order to bring its length up to 20 octets. The
 1 encoded (and possibly padded) NETs shall then be
 1 treated as unsigned binary integers.

2 8.15.3 Phase 3: Route Dissemination

2 The Phase 3 decision function shall be invoked on
 2 completion of Phase 3, or when any of the following
 2 events occur:

- 2 a) when routes in a Loc-RIB to local destinations
2 have changed
- 2 b) when locally generated routes with the EXT_INFO
2 attribute (that is, routes learned by means
2 outside of IDRP) have changed
- 2 c) when a new BIS-BIS connection has been estab-
2 lished
- 2 d) when directed to do so by system management.

2 The Phase 3 function is a separate process which
 2 completes when it has no further work to do.

2 The Phase 3 Routing Decision function shall be
 2 blocked from running while the Phase 2 decision func-
 2 tion is in process.

2 All routes in the Loc-RIB shall be processed into a
 2 corresponding entry in the associated Adj-RIBs-Out
 2 and FIBs (which are identified by the same RIB-Att),
 2 replacing the current entries. Route aggregation and
 2 information reduction techniques (see 8.17—8.17.2.3)
 2 may optionally be applied. Every FIB shall have an
 2 entry for every destination for which a route exists in
 2 an Loc-RIB.

2 A locking scheme shall be implemented to prevent
 2 simultaneous access to an FIB by both the phase 3
 2 function and forwarding engine. The phase 3 function
 2 shall first lock an FIB before entering, replacing or
 2 deleting an entry, and then unlock that FIB once the
 2 operation is complete.

2 When the updating of the Adj-RIBs-Out and the FIBs is
 2 complete, the local BIS shall run the external update
 2 process of 8.16.2.

1 8.15.3.1 Overlapping Routes

1 A BIS may transmit routes with overlapping NLRI to
 1 another BIS. NLRI overlap occurs when a set of des-
 1 tinations are identified in non-matching multiple
 1 routes, all of which have the same set of distin-
 1 guishing attributes. Since IDRP encodes NLRI using
 1 prefixes, overlaps will always exhibit subset relation-
 1 ships. A route describing a smaller set of destina-
 1 tions (a longer prefix) is said to be *more specific* than
 1 a route describing a larger set of destinations (a
 1 shorter prefix); similarly, a route describing a larger
 1 set of destinations (a shorter prefix) is said to be *less*
 1 *specific* than a route describing a smaller set of desti-
 1 nations (a longer prefix).

1 When overlapping routes are present in the same
 1 Adj-RIB-In, the more specific routes shall take preced-
 1 ence, in order from most specific to least specific.

1 This precedence relationship effectively decomposes
 1 less specific routes into two parts:

- 1 — a set of destinations described only by the less
1 specific route, and
- 1 — a set of destinations described by the overlap of
1 the less specific and the more specific routes

1 The set of destinations described by the overlap rep-
 1 resent a portion of the less specific route that is fea-
 1 sible, but is not currently in use. If a more specific
 1 route is later withdrawn, the set of destinations
 1 described by the overlap will still be reachable using
 1 the less specific route.

1 If a BIS receives overlapping routes, the Decision
 1 Process shall not alter the semantics of the overlap-
 1 ping routes. In particular, a BIS shall not accept the
 1 less specific route while rejecting the more specific

1 route, because the destinations represented by the
 1 overlap will not be forwarded along that route. There-
 1 fore, a BIS has the following choices:

- 1 a) Install both the less and more specific routes
- 1 b) Install the more specific route only
- 1 c) Install the non-overlapping part of the less spe-
 1 cific route only
- 1 d) Aggregate the two routes and install the aggre-
 1 gated route
- 1 e) Install neither route

8.15.4 Interaction with Update Process

Since the Adj-RIBs-In are used both to receive inbound UPDATE PDUs and to provide input to the Decision Process, care must be taken that their contents are not modified while the Decision Process is running. That is, the input to the Decision Process shall remain stable while a computation is in progress.

Two examples of approaches that could be taken to accomplish this:

- a) The Decision Process can signal when it is running. During this time, any incoming UPDATE PDUs will be queued and will not be written into the Adj-RIBs-In. If more UPDATE PDUs arrive than can be fit into the allotted queue, they will be dropped and will not be acknowledged.
- b) A BIS can maintain two copies of the Adj-RIBs-In—one used by the Decision Process for its computation (call this the Comp-Adj-RIB) and the other to receive inbound UPDATE PDUs (call this the Holding-Adj-RIB). Each time the Decision begins a new computation, the contents of the Holding-Adj-RIB will be copied to the Comp-Adj-RIB: that is, the a snapshot of the Comp-Adj-RIB is used as the input for the Decision Process. The contents of the Comp-Adj-RIB remain stable until a new computation is begun.

The advantage of the first approach is that it takes less memory; the advantage of the second is that inbound UPDATE PDUs will not be dropped. This international standard does not mandate the use of either of these methods. Any method that guarantees that the input data to the Decision Process will remain stable while a computation is in progress and that is consistent with the conformance requirements of this international standard may be used.

8.16 Update-Send Process

The Update-Send process is responsible for advertising BISPDU to adjacent BISs. For example, it distributes the routes chosen by the Decision Process to other BISs which may be located in either the same RD or an adjacent RD. Rules for information exchange between BIS located in different routing

domains are given in clause 8.16.2; rules for information exchange between BIS located in the same domain are given in clause 8.16.1.

Distribution of reachability information between a set of BISs, all of which are located in the same routing domain, is referred to as internal distribution. All BISs located in a single RD must present consistent reachability information to adjacent RDs, thus requiring that they have consistent routing and policy information among them.

NOTE: This requirement on consistency does not preclude an RD from distributing different reachability information to each of its adjacent routing domains. It does mean that all of a domain's BISs which are attached to a given adjacent domain must provide identical reachability to that domain.

When this protocol is run between BISs located in different routing domains, the communicating BISs must be located in adjacent routing domains—that is, they must be attached to a common subnetwork.

8.16.1 Internal Updates

The internal update process is concerned with the distribution of routing information to BISs located in the local BIS's own routing domain.

The following procedures shall be applied separately for each set of Distinguishing Attributes supported by the BIS:

- a) When a BIS receives an UPDATE PDU from another BIS located in its own routing domain, the receiving BIS shall not re-distribute the routing information contained in that UPDATE PDU to other BISs located in its own routing domain.
- b) When a BIS receives a new route from a BIS in an adjacent routing domain, it shall advertise that route to all other BISs in its routing domain by means of an UPDATE PDU if any of the following conditions occur:
 - 1) the degree of preference assigned to the newly received route by the local BIS is higher than the degrees of preference that the local BIS has assigned to other routes—with the same destinations and the same Distinguishing Attributes—that have been received over inter-domain links, or
 - 2) there are no other routes—with the same destinations and the same Distinguishing Attributes—that have been received over inter-domain links
 - 3) the newly received route is selected as a result of breaking a tie between several routes which have the highest degree of preference, the same destinations, and the same distinguishing attributes.

- 2 c) When a BIS receives an UPDATE PDU with a non-
 2 empty WITHDRAWN ROUTES field, it shall
 2 remove from its Adj-RIBs-In all routes whose
 2 ROUTE-ID was carried in this field. The BIS shall
 2 take the following additional steps:
- 2 1) if the corresponding feasible route had not
 2 been previously advertised, then no further
 2 action is necessary
- 2 2) if the corresponding feasible route had been
 2 previously advertised, then:
- 2 i) if a new route is selected for advertise-
 2 ment that has the same distinguishing
 2 attributes and NLRI as the unfeasible
 2 route, then the local BIS shall advertise
 2 the replacement route
- 2 ii) if a replacement route is not available
 2 for advertisement, then the BIS shall
 2 include the ROUTE-ID of the unfeasible
 2 route in the WITHDRAWN ROUTES field
 2 of an UPDATE PDU, and shall send this
 2 PDU to each neighbor BIS to whom it
 2 had previously advertised the corre-
 2 sponding feasible route.
- 2 All feasible routes which are advertised shall be
 2 placed in the appropriate Adj-RIB-Out, and all unfea-
 2 sible routes which are advertised shall be removed
 2 from the Adj-RIBs-Out.

1 8.16.1.1 Breaking Ties (Internal Updates)

- 1 If a local BIS has connections to several BISs in adja-
 1 cent domains, there will be multiple Adj-RIBs-In asso-
 1 ciated with these neighbors. These Adj-RIBs-In might
 1 contain several equally preferable routes to the same
 1 destination, all of which have the same set of distin-
 1 guishing attributes and all of which were advertised
 1 by BISs located in adjacent routeing domains. The
 1 local BIS shall select one of these routes, according
 1 to the following rules:
- 1 a) If the candidate routes differ only in their
 1 NEXT_HOP and MULTI-EXIT_DISC attributes, and
 1 the local BIS's managed object **Multixit** is TRUE,
 1 select the route that has the lowest value of the
 1 MULTI-EXIT_DISC attribute.
- 1 b) In all other cases, select the route that was
 1 advertised by the BIS whose NET has the lowest
 1 value.
- 1 For purposes of determining the lowest-valued NET,
 1 each binary-encoded NET shall be padded with trailing
 1 0's in order to bring its length up to 20 octets. The
 1 encoded (and possibly padded) NETs shall then be
 1 treated as unsigned binary integers.

8.16.2 External Updates

- 2 The external update process is concerned with the
 2 distribution of routeing information to BISs located in
 2 adjacent routeing domains. As part of the Phase 3
 2 route selection process, the BIS has updated its
 2 Adj-RIBs-Out and its FIBs. All newly installed routes
 2 and all newly unfeasible routes for which there is no
 2 replacement route shall be advertised to BISs located
 2 in adjacent routeing domains by means of UPDATE
 2 PDUs.
- 2 Any routes in the Loc-RIB marked as infeasible shall
 2 be removed. Changes to the reachable destinations
 2 within its own RD shall also be advertised in an
 2 UPDATE PDU.

However, advertisement of a given UPDATE PDU shall not violate any distribution constraint imposed by the path attributes of the route contained therein. (For example, DIST_LIST_INCL, DIST_LIST_EXCL, and HIERARCHICAL_RECORDING are attributes that impose distribution constraints on the UPDATE PDU that contains them.)

A BIS shall not propagate an UPDATE PDU that contains a set of distinguishing path attributes that were not listed in the RIB-AttsSet field of the neighbor BIS's OPEN PDU. If such distinguishing attributes are advertised, it will cause the BIS-BIS connection to be closed, as described in clause 8.20.3.

8.16.3 Controlling Routeing Traffic Overhead

The inter-domain routeing protocol constrains the amount of routeing traffic (that is, BISPDU's) in order to limit both the link bandwidth needed to advertise BISPDU's and the processing power needed by the Decision Process to digest the information contained in the BISPDU's.

1 8.16.3.1 Frequency of Route Advertisement

- 1 The managed object **MinRouteAdvertisementInterval**
 1 determines the minimum amount of time that must
 1 elapse between advertisements of routes to a partic-
 1 ular destination from a single BIS. This rate limiting
 1 procedure applies on a per-destination basis, although
 1 the value of **MinRouteAdvertisementInterval** is set on
 1 a per-BIS basis.
- 1 Two UPDATE PDUs sent from a single BIS that adver-
 1 tise feasible routes to some common set of destina-
 1 tions received from BISs in other routeing domains
 1 must be separated in time by at least
 1 **MinRouteAdvertisementInterval**.
- 1 Since fast convergence is needed within an RD, this
 1 procedure does not apply for routes received from
 1 other BISs in the same routeing domain. To avoid
 1 long-lived black holes, the procedure does not apply
 1 to the explicit withdrawal of unfeasible routes (that is,

1 routes whose ROUTE_ID is listed in the *Withdrawn*
 1 *Routes* field of an UPDATE PDU).

1 This procedure does not limit the rate of route
 1 selection, but only the rate of route advertisement. If
 1 new routes are selected multiple times while awaiting
 1 the expiration of **MinRouteAdvertisementInterval**, the
 1 last route selected shall be advertised at the end of
 1 **MinRouteAdvertisementInterval**.

8.16.3.2 Frequency of Route Origination

The architectural constant **MinRDOriGinationInterval** determines the minimum amount of time that must elapse between successive advertisements of UPDATE PDUs that report changes within the advertising BIS's own routeing domain.

8.16.3.3 Jitter

To minimize the likelihood that the distribution of BISPDU's by a given BIS will contain peaks, jitter should be applied to the timers associated with **MinRouteSelectionInterval** and **MinRDOriGinationInterval**. A given BIS shall apply the same jitter to each of these quantities regardless of the destinations to which the updates are being sent: that is, jitter will not be applied on a "per peer" basis.

The amount of jitter to be introduced shall be determined by multiplying the base value in the appropriate managed object by a random factor which is uniformly distributed in the range from $1 - J$ to 1, where J is the value of the architectural constant **Jitter**. The result shall be rounded up to the nearest 100 millisecond increment.

An example of a suitable algorithm is shown in informative annex Appendix C, using the architectural constant **jitter**.

1 8.17 Efficient Organization of Routeing 1 Information

1 Having selected the routeing information which it will
 1 advertise, a BIS may avail itself of several methods
 1 to organize this information in an efficient manner.

8.17.1 Information Reduction

Information reduction may imply a reduction in granularity of policy control—after information is collapsed, the same policies will apply to all destinations and paths in the equivalence class.

The Decision Process may optionally reduce the amount of information that it will place in the Adj-RIBs-Out by any of the following methods:

- a) **Network Layer Reachability Information:**

Destination NSAP addresses can be represented as NSAP address prefixes. In cases where there is a correspondence between the address structure and the systems under control of a routeing domain administrator, it will be possible to reduce the size of the network layer reachability information that is carried in the UPDATE PDUs.

b) **RD_PATHS:**

RD path information can be represented as ordered RD-SEQUENCES or unordered RD_SETs. RD_SETs are used in the route aggregation algorithm described in 8.17.2. They reduce the size of the RD_PATH information by listing each RDI only once, regardless of how many times it may have appeared in the multiple RD_PATHS that were aggregated.

An RD_SET implies that the destinations listed in the NLRI can be reached through paths that traverse at least some of its constituent RDs. RD_SETs provide sufficient information to avoid routeing loops; however, their use may prune potentially useful paths, since such paths are no longer listed individually as in the form of RD-SEQUENCES. In practice this is not likely to be a problem, since once an NPDU arrives at the edge of a group of RDs, the BIS at that point is likely to have more detailed path information and can distinguish individual paths to destinations.

8.17.2 Aggregating Routeing Information

Aggregation is the process of combining the characteristics of several different routes (or components of a route such as an individual path attribute) in such a way that a single route can be advertised. Aggregation can occur as part of the decision process to reduce the amount of information that will be placed in the Adj-RIBs-Out. For example, at the boundary of a routeing domain confederation an exit BIS can aggregate several intra-confederation routes into a single route that will be advertised externally.

Aggregation reduces the amount of information that BISs must store and exchange with each other. Routes can be aggregated by applying the following procedures separately to path attributes of like type and to the NLRI information.

8.17.2.1 Route Aggregation

Several routes shall not be aggregated into a single route unless the Distinguishing Attributes of each of these route are the same. In particular,

- a) Feasible routes may be aggregated with feasible routes.
- b) Routes that contain the DIST_LIST_INCL attribute may not be aggregated with routes that contain the DIST_LIST_EXCL attribute.

Routes that have the following attributes shall not be aggregated unless the corresponding attributes of each route are identical: MULTI-EXIT_DISC, LOC_PREF, and NEXT_HOP.

- 2 An aggregated route is constructed from one or more
- 2 component routes. If a component of an aggregated
- 2 route that has been advertised in an UPDATE PDU
- 2 becomes unfeasible, then all component routes that
- 2 comprise the aggregated route, except for the unfea-
- 2 sible component, shall be advertised again, either as
- 2 separate routes or as a new aggregated route. If the
- 2 new aggregated route has the same NLRI as the pre-
- 2 vious aggregated route, then no further actions are
- 2 necessary, since advertisement of the new aggre-
- 2 gated route implicitly marks the old aggregated route
- 2 as having been withdrawn from use. In all other
- 2 cases, the original aggregated route must be with-
- 2 drawn explicitly by means of the *Withdrawn Routes*
- 2 field of an UPDATE PDU.

8.17.2.2 NLRI Aggregation

The aggregation of the NLRI fields from several routes is straightforward: if a shorter NSAP address prefix can be used to represent the NSAPs (and only those NSAPs) listed in several individual NSAP address prefixes, this may be done. However, a shorter NSAP prefix which would be associated with more NSAPs than were associated with the individual prefixes being aggregated shall not be used.

8.17.2.3 Path Attribute Aggregation

Path attributes that have different type codes can not be aggregated together. Path attributes of the same type code may be aggregated, according to the following rules:

- EXT_INFO attributes:** If at least one route among routes that are aggregated has the EXT_INFO attribute, then the aggregated route must have the EXT_INFO attribute as well.
- RD_PATH attributes:** If routes to be aggregated have identical RD_PATH attributes, then the aggregated route has the same RD_PATH attribute as each individual route.
- RD_PATH attributes:** If routes to be aggregated have identical RD_PATH attributes, then the aggregated route has the same RD_PATH attribute as each individual route.
- If a route has an RD_PATH attribute that contains a confederation entry marker (see 8.13.4), then that route shall not be aggregated with any other route.
- If routes to be aggregated have different RD_PATH attributes, then the rules for determining the RD_PATH attribute of the aggregated route are specified by the following algorithm:

- a) Set RD_PATH of the aggregated route to empty.
- b) Determine the longest common final sequence of the RD_PATH segments among all the routes to be aggregated.
- c) Append this longest common final sequence (as is) to the RD_PATH of the aggregated route, as its final segment.
- d) Delete this longest common final sequence from the RD_PATH attribute of each of the routes to be aggregated.
- e) Determine the longest different final sequences of the RD_PATH segments among all the routes to be aggregated.
- f) Combine all RDIs in these sequences into an RD_SET path segment and prepend it to the RD_PATH attribute of the aggregated route.
- g) Delete sequences determined in step e from the RD_PATH attribute of each of the routes to be aggregated.
- h) Repeat steps b to g until there all the input routes have empty RD_PATH segments.

DIST_LIST_INCL attributes: The DIST_LIST_INCL attribute of the aggregated route is formed by the set intersection of the RDIs listed in the DIST_LIST_INCL attributes of the individual routes. If the set intersection consists of an empty set, then the routes shall not be aggregated.

DIST_LIST_EXCL attributes: The DIST_LIST_EXCL attribute of the aggregated route is formed by the set union of the RDIs listed in the DIST_LIST_EXCL attribute of the individual routes. A route that contains no DIST_LIST_EXCL attribute is treated as if it contained a DIST_LIST_EXCL attribute that lists no RDIs.

Transit Delay attributes: The value of the Transit Delay attribute of the aggregated route is set to the maximum value of the Transit Delay attribute of the individual routes that are aggregated.

Residual Error attributes: The value of the Residual Error attribute of the aggregated route is set to the maximum value of the Residual Error attribute of the individual routes that are aggregated.

Expense attributes: The value of the Expense attribute of the aggregated route is set to the maximum value of the Expense attribute of the individual routes that are aggregated.

Source Specific QOS attributes: The rules for determining the value of the Source Specific QOS attribute of the aggregated route are determined by the authority responsible for assigning the Source Specific QOS.

Destination Specific QOS attributes: The rules for determining the value of the Destination Specific QOS attribute of the aggregated route are determined by the authority responsible for assigning the Destination Specific QOS.

HIERARCHICAL RECORDING attributes: If any of the routes to be aggregated contains the HIERARCHICAL_RECORDING attribute, the aggregated route shall also contain a HIERARCHICAL_RECORDING attribute:

- If the routes to be aggregated contain different values for this attribute, then it shall be set to a value of 0 in the aggregated route
- If all routes to be aggregated contain the same value for this attribute, then it shall be set to that value in the aggregated route.

RD_HOP COUNT The value of the RD_HOP_COUNT of the aggregated route shall be set equal to the largest RD_HOP_COUNT that was contained in the routes being aggregated.

Source Specific Security attribute: The rules for determining which source specific security levels can be aggregated, and the rules for doing so, are determined by the authority responsible for originating the attribute.

Destination Specific Security attributes: The rules for determining which destination specific security levels can be aggregated, and the rules for doing so, are determined by the authority responsible for originating the attribute.

- 2 **Priority:** The value of the PRIORITY attribute of the aggregated route shall be equal to the maximum value of the PRIORITY fields in the routes being aggregated. A route that does not contain the PRIORITY attribute shall be treated as if it contained a priority value of 0.

Capacity The value of the CAPACITY attribute of the aggregated route shall be equal to the minimum value of the CAPACITY fields in the routes being aggregated.

2 8.18 Maintenance of the Forwarding Information Bases

- 2 As summarized in Table 1, the Forwarding Information Bases contain the following information for a given RIB-Att (set of distinguishing attributes) and set of destinations (NLRI): the NET of the next-hop BIS, the local SNPA used by the local BIS to forward traffic to the next-hop BIS, and, if available, the SNPA in the next-hop BIS upon which traffic will be received.

- 2 The RIB-Att of the Loc-RIB which contains a route is also the RIB-Att of the corresponding FIB; the NLRI for the associated FIB is the same as the NLRI of the corresponding route that is stored in the Loc-RIB.
- 2 Therefore, the destination address of an incoming

- 2 NPDU and its NPDU-derived distinguishing attributes (see 9.3) allow a BIS to determine the FIB which contains the forwarding information to be used for this NPDU.

- 2 The forwarding information consists of three parts.

- 2 a) **Net of Next-hop BIS:** For each route in the Loc-RIB, the next-hop BIS has been determined, and is carried as a tag, as described in 8.16.2. The same tag is then carried over into the corresponding entry in the FIB. This information is always present.
- 2 b) **Output SNPA:** The SNPA that will be used by the local BIS for forwarding traffic to the destinations identified in the NLRI field of the FIB is established locally, and is one of the SNPAs identified in managed object **LocalSNPAs**.
- 2 c) **Input SNPA:** The SNPA that will be used by the remote BIS to receive traffic that is the NEXT_HOP attribute of the corresponding route stored in the Loc-RIB. If the NEXT_HOP attribute contains an empty SNPA list, or if the NEXT_HOP attribute itself is not included in the route, then the Input SNPA field in the FIB will be empty.

8.19 Receive Process

Within the Network layer, the IDRP protocol is located above the ISO 8473 protocol. Therefore, IDRP relies upon ISO 8473 to perform the initial processing of incoming PDUs. After processing the input NPDU, the ISO 8473 protocol machine that resides within the receiving BIS will deliver:

- a) BISPDUs to the IDRP Update-Receive Process, or
- b) ISO 8473 NPDUs to the IDRP CLNS Forwarding Process.

- 2 The ISO 8473 protocol machine shall process inbound NPDUs according to the appropriate ISO 8473 functions.

- 2 If the NPDU contains an SPI that identifies IDRP, and the NPDU's source address identifies any system listed in managed objects **INTERNAL-BIS** or **EXTERNAL-BIS-NEIGHBORS**, then the data part of the NPDU contains a BISPDUs. This BISPDUs shall be passed to the IDRP protocol machine for further processing by its Update-Receive Process (clause 8.14) and error handling processes (clause 8.20), as appropriate.

- 2 However, if the source address of the NPDU identifies a system that is not listed in these managed objects, then the NPDU shall be rejected by the receiving BIS, and the BIS shall send a notification ("packetbomb") to system management.

- 2 If the SPI identifies ISO 8473, decapsulate the inner PDU and pass it back to the ISO 8473 protocol

- 2 machine. (This step permits iterations on multiply
- 2 encapsulated NPDUs, which may occur, for example,
- 2 as described in clause 9.4, item b2.)

8.20 Error Handling for BISPDU

This section describes actions to be taken when errors are detected while processing BISPDU.

8.20.1 BISPDU Header Error Handling

If BIS-BIS connection was established using authentication code 2 (checksum plus authentication) and the validation pattern in the BISPDU header does not match the locally computed pattern, then the BISPDU shall be discarded without any further actions.

If any of the following error conditions are detected, the BISPDU shall be discarded, and the appropriate error event shall be logged by the receiving BIS:

- a) Length field of a PDU header less than 30 octets or greater than the Segment Size specified by the remote system's OPEN PDU,
- b) Length field of an OPEN PDU less than minimum length of an OPEN_PDU
- c) Length field of an UPDATE PDU less than minimum length of an UPDATE PDU
- d) Length field of KEEPALIVE PDU not equal to 30
- e) The BIS-BIS connection was established using authentication code 1 (checksum without authentication) and the validation pattern in the BISPDU header does not match the locally computed pattern
- f) Type field in the BISPDU is not recognized

8.20.2 OPEN PDU Error Handling

The following errors detected while processing the OPEN PDU shall be indicated by sending an IDRP ERROR PDU with error code OPEN_PDU_Error. The error subcode of the IDRP ERROR PDU shall elaborate on the specific nature of the error.

- a) If the version number of the received OPEN PDU is not supported, then the error subcode of the IDRP ERROR PDU shall be set to **Unsupported_Version_Number**. The Data field of the IDRP ERROR PDU is a 2-octet unsigned integer, which indicates the highest supported version number. less than the version of the remote BIS peer's bid (as indicated in the received OPEN PDU).
- b) If the Maximum PDU Size field of the OPEN PDU is less than **MinPDULength** octets, the error subcode of the IDRP ERROR PDU is set to **Bad_Maximum_PDU_Size**. The Data field of the IDRP ERROR PDU is a 2 octet unsigned integer

which contains the erroneous Maximum PDU Size field.

- c) If the Outstanding PDUs field of the OPEN PDU is zero, the error subcode of the IDRP ERROR PDU is set to **Bad_Outstanding_PDUs**. The Data field of the IDRP ERROR PDU is a 1 octet unsigned integer which contains the erroneous Outstanding PDUs field.
- d) If the Routeing Domain Identifier field of the OPEN PDU is not the expected one, the error subcode of the IDRP ERROR PDU is set to **Bad_Peer_RD**. The expected values of the Routeing Domain Identifier may be obtained by means outside the scope of this protocol (usually it is a configuration parameter).
- e) If a BIS receives an OPEN PDU from a BIS located in the same RD, and the RIB-AttsSet field contained in that PDU is different from the receiving BIS's managed object **RIB-AttsSet**, then the error subcode of the IDRP ERROR PDU shall be set to **Bad-RIB-AttsSet**.
- f) If the value of the Authentication Code field of the OPEN PDU is any value other than 1 or 2, the error subcode of the IDRP ERROR PDU is set to **Unsupported_Authentication_Code**.
- g) If a given BIS receives an OPEN PDU from another BIS located in the same routeing domain, then the RDIs reported in the Confed-IDs field of the OPEN PDU (received from the remote BIS) should match the Confed-IDs of the local BIS. If they do not match exactly, then an IDRP ERROR PDU shall be issued, indicating an OPEN PDU error with an error subcode of **RDC_Mismatch**. The data field of the IDRP ERROR PDU shall report the offending Confed-IDs field from the rejected OPEN PDU.

NOTE: The following condition does not constitute an OPEN PDU error:

- a) The managed object **RDC-Config** for the local BIS indicates that it is a member of several RDCs with a given nesting relationship
- b) An incoming OPEN PDU reports that the remote BIS is a member of some but not all of the same RDCs.

This may be a transient condition which will only exist during the formation or deletion of an RDC; hence, the local BIS should continue with normal processing of the OPEN PDU, and should be required to notify network management of the situation.

8.20.3 UPDATE PDU Error Handling

All errors detected while processing the UPDATE PDU are indicated by sending an IDRP ERROR PDU with error code UPDATE_PDU_Error. The error subcode of the IDRP ERROR PDU elaborates on the specific nature of the error.

- a) If the Total Attribute Length is inconsistent with the Length field of the PDU header, then the error

- subcode of the IDRP ERROR PDU shall be set to `Malformed_Attribute_List`. No further processing shall be done and all information in the UPDATE PDU shall be discarded.
- b) If any recognized attribute has attribute flags that conflict with the attribute type code, then the error subcode of the IDRP ERROR PDU shall be set to `Attribute_Flags_Error`. The Data field of the IDRP ERROR PDU shall contain the incorrect attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - c) If any recognized attribute has a length that conflicts with the expected length (based on the attribute type code), then the error subcode of the IDRP ERROR PDU shall be set to `Attribute_Length_Error`. The Data field of the IDRP ERROR PDU contains the incorrect attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - d) If any of the mandatory well-known attributes are not present, then the error subcode of the IDRP ERROR PDU shall be set to `Missing_Well-known_Attribute`. The Data field of the IDRP ERROR PDU contains the attribute type code of the missing well-known attribute.
 - e) If any well-known attribute (so designated by the attribute flags) is not recognized, then the error subcode of the IDRP ERROR PDU shall be set to `Unrecognized_Well-known_Attribute`. The Data field of the IDRP ERROR PDU shall report the unrecognized attribute (type, length and value). In both cases no further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - f) If the `NEXT_HOP` attribute field is invalid, then the error subcode of the IDRP ERROR PDU shall be set to `Invalid_NEXT_HOP_Attribute`. The Data field of the IDRP ERROR PDU contains the incorrect attribute (type, length and value). No further processing shall be done and all information in the UPDATE PDU shall be discarded.
 - g) The sequence of RD path segments shall be checked for RD loops. RD loop detection shall be done by scanning the complete list of RD path segments (as specified in the `RD_PATH` attribute) and checking that each RDI in this list occurs only once. If an RD loop is detected, then the error subcode of the IDRP ERROR PDU shall be set to `RD_Routeing_Loop`. The Data field of the IDRP ERROR PDU shall report the incorrect attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - h) If the non-empty set of distinguishing attributes of an UPDATE PDU received from a BIS located in a different routeing domain does not match any of the RIB-Atts that the local (receiving) BIS had advertised to that neighbor in the RIB-AttsSet field of its OPEN PDU, then the receiving BIS shall send an IDRP Error PDU that reports an error subcode of `Malformed_Attribute_List`. All information in the UPDATE PDU shall be discarded, and no further processing shall be done.
 - i) If the UPDATE PDU contains both the `DIST_LIST_INCL` and the `DIST_LIST_EXCL` attributes, then the error subcode of the IDRP ERROR PDU shall be set to `Malformed_Attribute_List`. No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - j) If a BIS receives an UPDATE PDU that has a `DIST_LIST_EXCL` attribute with the RDI of the receiving BIS or the RDI of any RDC to which the BIS belongs, the subcode of the IDRP ERROR PDU shall be set to `Malformed_Attribute_List`. The Data field of the IDRP ERROR PDU shall report the incorrect attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - k) If a BIS receives an UPDATE PDU that has a `DIST_LIST_INCL` attribute without the RDI of the receiving BIS or the RDI of any RDC to which the BIS belongs, the subcode of the IDRP ERROR PDU shall be set to `Malformed_Attribute_List`. The Data field of the IDRP ERROR PDU shall report the incorrect attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
- NOTE:** It is permissible for an UPDATE PDU to contain neither the `DIST_LIST_INCL` nor the `DIST_LIST_EXCL` attributes. According to clause 8.12.5, the absence of both the `DIST_LIST_INCL` and `DIST_LIST_EXCL` attributes implies that all RDs and all RDCs may receive the routeing information.
- l) If an optional attribute is recognized, then the value of this attribute shall be checked. If an error is detected, the attribute shall be discarded, and the error subcode of the IDRP ERROR PDU shall be set to `Optional_Attribute_Error`. The Data field of the IDRP ERROR PDU shall report the attribute (type, length and value). No further processing shall be done, and all information in the UPDATE PDU shall be discarded.
 - m) If RDCs are supported and any of the error conditions noted in clause 8.13.6 occur, no further processing of the UPDATE PDU shall be done, all information in the UPDATE PDU shall be discarded, and the error code of the NOTIFICATION PDU shall be set to `Misconfigured_RDCs`.

8.20.4 IDRP ERROR PDU Error Handling

If a BIS receives an IDRP ERROR PDU with a correct validation pattern but which contains an unrecognized error code or error subcode, the local BIS shall close the connection as described in clause 8.6.2.

NOTE: Any error (such as unrecognized Error Code or Error Subcode, or an incorrect Length field in the PDU header) should be logged locally and brought to the attention of the administration of the peer. The means to do this are, however, outside the scope of this protocol.

8.20.5 Hold Timer Expired Error Handling

If a system does not receive successive KEEPALIVE or UPDATE PDUs within the period specified in the Hold Time field of the OPEN PDU, then an IDRP ERROR PDU with error code Hold_Timer_Expired shall be sent, and the FSM for the associated BIS-BIS connection shall enter the CLOSE-WAIT state.

8.20.6 KEEPALIVE PDU Error Handling

The KEEPALIVE PDU consists of only the BISPDU Header. Error conditions are handled according to 8.20.1.

8.20.7 CEASE PDU Error Handling

The CEASE PDU consists of only the BISPDU Header. Error conditions are handled according to 8.20.1

8.20.8 RIB REFRESH PDU Error Handling

If any of the following error conditions are detected, the BIS shall issue an IDRP ERROR PDU with the following error indications:

- a) Invalid OpCode not in Range 1 to 3: indicate RIB REFRESH error with error subcode "Invalid OpCode"
- b) Receipt of an OpCode 3 (RIB Refresh End) without prior receipt of OpCode 2 (Rib Refresh Start): indicate FSM Error
- c) Receipt of an unsupported RIB-Att in the Rib-Atts variable length field in the RIB REFRESH PDU for a RIB Refresh Start OpCode: indicate RIB REFRESH error with error subcode "Unsupported RIB-Atts"

9. Forwarding Process for CLNS

The forwarding process for CLNS operation is driven by the header information carried in an ISO 8473 NPDU:

- a) If the NPDU contains an ISO 8473 complete source route parameter, then further forwarding of this NPDU shall be handled by the ISO 8473 protocol, not by the mechanisms defined in this international standard.
- b) If the NPDU contains an ISO 8473 partial source route parameter, the NPDU shall be forwarded on a path to the next system listed in the partial source route parameter.
- c) If the NPDU does not contain an ISO 8473 source route parameter, the NPDU shall be forwarded on a path to the system listed in the destination address field of the NPDU.

Having determined the system to which a path is needed, the BIS shall proceed as follows:

- a) If the destination system is located in its own RD, the local BIS shall proceed as defined in clause 9.1.
- b) If the destination system is located in a different RD, the local BIS shall perform the following actions:
 - 1) It shall determine the *NPDU-Derived Distinguishing Attributes* of the NPDU, according to clause 9.2.
 - 2) It shall next apply the procedures of clause 9.3 to determine if the NPDU-derived Attributes match any of the RIB-Atts of the information base(s) supported by the local BIS:
 - i) If there is no match, then the local BIS shall perform the ISO 8473 "Discard PDU Function" (see clause 6.9 of ISO 8473), and it shall generate an ER PDU with the parameter value set to "Unsupported Option not Specified".
 - ii) If there is a match, then the local BIS shall proceed as described in clause 9.4.

9.1 Forwarding to Internal Destinations

If the destination address of an incoming NPDU depicts a system located within the routing domain of the receiving BIS, then it shall forward that NPDU to any of the ISs listed in managed object **INTRA-IS**. That is, any further forwarding of the NPDU is the responsibility of the intra-domain routing protocol.

9.2 Determining the NPDU-derived Distinguishing Attributes

Table 4. NPDU-Derived Attribute Set. Some NPDU-derived Distinguishing attributes are by examining the QOS Maintenance Parameter octet for 1 or 0 in the bit positions shown below. The symbol “-” indicates that the corresponding bit does not enter into the determination.

QOS Maintenance Parameter								NPDU-Derived Attributes
b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	
1	1	-	-	-	1	0	0	Transit Delay
1	1	-	-	-	1	0	1	Transit Delay
1	1	-	-	-	0	0	0	Expense
1	1	-	-	-	0	1	0	Expense
1	1	-	-	-	0	1	1	Residual Error
1	1	-	-	-	1	1	1	Residual Error
1	1	-	-	-	0	0	1	Capacity
1	1	-	-	-	1	1	0	Capacity
0	1	0	0	0	0	0	0	Source Specific QOS
1	0	0	0	0	0	0	0	Destination Specific QOS

As the first step in forwarding an NPDU to a destination located in another routing domain, the receiving BIS shall determine the *NPDU-derived Distinguishing Attributes* of the incoming ISO 8473 NPDU. This determination shall be based on an examination of the priority parameter, security parameter, and QOS maintenance parameter in the NPDU's header:

- The 8473 priority parameter corresponds to the PRIORITY path attribute.
- The first two high order bits of the 8473 security parameter are decoded. If they equal to '01', then the security parameter corresponds to the SOURCE SPECIFIC SECURITY path attribute; if equal to '10', it corresponds to the DESTINATION SPECIFIC SECURITY path attribute.
- The remainder of the NPDU-Derived Distinguishing Attributes are derived by decoding the first octet of the QOS maintenance parameter, as shown in Table 4

If examination of the 8473 header shows that no NPDU-Derived Distinguishing Attributes are present, then the NPDU shall be associated with the Empty Distinguishing Attribute.

9.3 Matching RIB-Att to NPDU-derived Distinguishing Attributes

Within the BIS, each of its FIB(s) has an unambiguous RIB-Att (see clause 8.10.1) which is constructed from the set of Distinguishing Attributes that the local BIS supports. The set of NPDU-derived Distinguishing Attributes matches a given RIB-Att (which is itself a set of Distinguishing Attributes) when all of the following conditions are satisfied:

- a) Both sets contain the same number of attributes.
- b) Each instance of a type-specific attribute in the NPDU-derived Distinguishing Attributes must have an equivalent instance in the FIB-Att. The type-specific path attributes supported by IDRP are:
 - Capacity
 - Transit delay
 - Residual error
 - Expense
 - Priority
- c) Each instance of a type-value specific attribute in the NPDU-derived Distinguishing Attributes must have a corresponding instance in the FIB-Att with the same type and an equivalent value. The type-value specific attributes supported by IDRP are:
 - Source Specific QOS
 - Destination Specific QOS
 - Source Specific Security
 - Destination Specific Security

For the source and destination specific attributes, the NPDU-derived Distinguishing Attribute will report only a single relevant address, while in general the corresponding FIB-Att attribute apply to multiple addresses. A match occurs when the attribute types are identical and the address carried in the 8473 parameter is a subset of the address information contained in the FIB-Att. In searching for a match, the address information in the FIB-Att attribute shall be processed in order of descending length: that is a match with a

longer prefix shall take precedence over a match with a shorter prefix.

9.4 Forwarding to External Destinations

If the destination address of the incoming NPDU depicts a system located in a different routing domain from the receiving BIS, then the receiving BIS shall use the FIB identified by the FIB-Att that matches the NPDU-derived Distinguishing Attributes of the incoming NPDU. The incoming NPDU shall be forwarded based on the longest address prefix that matches (as in 8.1.2.2) the destination NSAP address of the incoming NPDU, as follows:

- a) If the entry in the inter-domain FIB that corresponds to the destination address of the incoming NPDU contains a NEXT_HOP entry that identifies a BIS which is located on at least one common subnetwork with the local BIS, then the NPDU shall be forwarded directly to the BIS indicated in the NEXT_HOP entry.
- b) If the entry in the inter-domain FIB that corresponds to the destination address of the incoming NPDU contains a NEXT_HOP entry that identifies a BIS which is not located on at least one common subnetwork with the local BIS, then the local BIS has the following options:

- 1) **Encapsulate the NPDU:** The local BIS may encapsulate the NPDU, using its own NET as the source address and the NET of the next-hop BIS as the destination address. The QOS parameter of the encapsulating (outer) NPDU shall be set equal to the QOS parameter of the encapsulated (inner) NPDU. The encapsulated NPDU shall then be handed over to the intra-domain routing protocol.

NOTE: It is a local responsibility to insure that the NPDU is encapsulated appropriately for the RD's intra-domain protocol. Since this international standard does not mandate the use of a specific intra-domain protocol, encapsulation details for specific intra-domain protocols are outside its scope.

- 2) **Use Paths Provided by the Intra-domain Routing Protocol:** The local BIS may query the intra-domain FIB to ascertain if the intra-domain protocol is aware of a route to the destination system⁹. If there is an intra-domain route that supports the QOS Maintenance parameter of the NPDU and will deliver the NPDU to the appropriate next-hop BIS, then the NPDU may be forwarded along this route.

NOTE: This case makes use of the intra-domain protocol's knowledge of suitable paths through the local RD which support the specified QOS parameter. It does not require encapsulation of the NPDU.

Details of the mapping between the QOS parameters of used by a given intra-domain protocol and the QOS Maintenance parameter of the NPDU must be determined by the intra-domain routing protocol; this mapping is not within the scope of IDRP.

10. Interface to ISO 8473

- 2 For the purposes of its interface to ISO 8473, a BIS shall be regarded as being comprised of a co-resident ES and IS. The protocol described in this international standard, although still within the Network layer, shall access the service provided by ISO 8473 through an NSAP in the ES part of the BIS. Hence, a BIS's NET shall, for the purposes of communication using ISO 8473, be regarded as NSAP address.

The protocol described in this international standard interfaces at its lower boundary to ISO 8473 using the ISO 8473 primitives shown in Table 5.

Table 5. IDRP-CL Primitives

Primitive	Parameters
N-UNITDATA Request	Destination NSAP Address Source NSAP Address QOS Userdata
N-UNITDATA Indication	Destination NSAP Address Source NSAP Address QOS Userdata

The parameters associated with the N-UNITDATA request primitive are:

- 2 — *Destination NSAP Address* is the NET of the BIS to which this BISPDU is being sent (that is, the remote BIS)
- 2 — *Source NSAP Address* is the NET the BIS that is sending this BISPDU (that is, the local BIS)

⁹ For example, if ISO 10589 were used as the intra-domain routing protocol, it would be able to calculate path segments through the RD for systems contained in its "reachable address prefixes".

- QOS is the quality of service parameter for this BIS-BIS connection
- *Userdata* is an ordered sequence of octets which comprise the BISPDU to be sent to the remote BIS

The parameters associated with the N-UNITDATA indication primitive are:

- 2 — *Destination NSAP Address* is the NET of the BIS to which this BISPDU is being sent
- 2 — *Source NSAP Address* is the NET the BIS that is sending this BISPDU (that is, the remote BIS)

- QOS is the quality of service parameter for this BIS-BIS connection
- *Userdata* is an ordered sequence of octets which comprise the BISPDU being delivered to the local BIS

11. Constants

This constants used by the protocol defined in this international standard are enumerated in Table 6.

Table 6. Architectural Constants of IDRP		
Name of Constant	Value	Description
Inter-domain Routeing Protocol Identifier	X'83'	The SPI for the protocol described in this international standard
MinBISPDULength	30	The size in octets of the smallest allowable BISPDU.
MinRDOriginationInterval	15 minutes	The minimum time between successive UPDATE PDUs advertising routeing information about the local RD
Jitter	0.25	The factor used to compute jitter according to clause 8.16.3.3.
MaxCPUOverloadPeriod	1 hour	Maximum time in which a BIS can remain CPU-overloaded before terminating its BIS-BIS connections.

12. System Management and GDMO Definitions

The operation of the inter-domain routing functions in a BIS may be monitored and controlled using System Management. This clause contains management specification for IDRP, expressed in the GDMO notation defined in ISO 10165-4.

12.1 Name Binding

idrpConfig-networkEntity NAME BINDING

```
SUBORDINATE OBJECT CLASS idrpConfig AND SUBCLASSES;
NAMED BY SUPERIOR OBJECT CLASS "ISO/IEC 10733 : 19xx":networkEntity;
WITH ATTRIBUTE "ISO/IEC xxxx": idrpConfigId;
CREATE
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;
REGISTERED AS { IDRP.nboi idrpConfig-networkEntity(1) };
```

adjacentBIS-idrpConfig NAME BINDING

```
SUBORDINATE OBJECT CLASS adjacentBIS AND SUBCLASSES;
NAMED BY SUPERIOR OBJECT CLASS idrpConfig AND SUBCLASSES;
WITH ATTRIBUTE bisNet;
BEHAVIOUR IDRPConfig-adjacentBIS-idrpConfig-B BEHAVIOUR
    DEFINED AS This name binding attribute identifies a BIS to BIS con-
    nection information block. One of these blocks of data should exist
    per remote BIS that this local BIS exchanges BISPDU with.;
REGISTERED AS { IDRP.nboi adjacentBIS-idrpConfig(2) };
```

12.2 Local BIS Managed Objects for IDRP

idrpConfig MANAGED OBJECT CLASS

```
DERIVED FROM "Rec. X.721 | ISO/IEC 10165-2 : 1992":top;
CHARACTERIZED BY idrpConfigPkg PACKAGE
BEHAVIOUR IDRPBasicImportedAlarmNotifications-B
    BEHAVIOUR DEFINED AS Imports the communicationsAlarm notification
    from ISO/IEC 10165-2. It is used to report the following protocol
    events:
```

errorBISPDUsent: generated when a BISPDU is received with an error in its format. In addition to the parameters specified by ISO/IEC 10733, the following information will be reported in the AdditionalInformation field for the BIS Connection on which the error BISPDU was received:

- a) RemoteBIS-NET for BIS-BIS connection--using the notificationRemoteBIS-NET parameter
- b) BISPDU error code (see 7.4 and 8.20)--this reports the error code that will be sent in the ERROR PDU using the parameter "notificationBISPDUErrortype".
- c) BIS error subcode (see 7.4 and 8.20)--this reports the subcode that will be sent using the parameter "notificationBISErrorsubcode".
- d) BISPDU error information (see 7.4 and 8.20)--this reports the data from the received BISPDU that will be used to diagnose the problem for the Notification. The parameter

"notificationBISpduerrorinfo" will be used to report this information.

OpenBISpduRDCError: generated when an OPEN BISPDU is received from another BIS in the same routing domain, and the remote BIS is not a member of identically the same confederations as the local BIS. In addition to the parameters specified by ISO/IEC 10733, the following information will be reported by the AdditionalInformation field for the BIS Connection on which this OPEN PDU was received:

- a) Remote BIS NET for this BIS-BIS connection--using the "notificationRemoteBIS-NET" parameter.
- b) Remote BIS Routing Domain Confederation (RDC) information using the "notificationRemoteRDCconfig" parameter.
- c) Local BIS Routing Domain Confederation (RDC) information using the "notificationLocalRDCconfig" parameter.

errorBISpduconnectionclose: generated when an ERROR PDU has been received from a remote BIS. In addition to the parameters specified by ISO/IEC 10733, the following information will be reported by the AdditionalInformation field for the BIS Connection on which this OPEN PDU was received:

- a) RemoteBIS-NET for BIS-BIS connection--using the "notificationREMOTEBIS-NET" parameter
- b) BISPDU error code (see 7.4 and 8.20)--this reports the error code that will be sent in the ERROR PDU using the parameter "notificationBISpduerrorcode".
- c) BIS error subcode (see 7.4 and 8.20)--this reports the subcode that will be sent using the parameter "notificationBISerrorsubcode".
- d) BISPDU error information (see 7.4 and 8.20)--this reports the data from the received BISPDU that will be used to diagnose the problem for the Notification. The parameter "notificationNBISPDUerrorinfo" will be used to report this information.

CorruptAdjRIBIn: generated when the local method of checking the Adj-RIB-In has found an error. All Adj-RIBs-In are being purged. In addition to the parameters specified by ISO/IEC 10733, the following information will be reported by the AdditionalInformation field:

- a) MaxRIBIntegrityCheck attribute for this BIS
- b) The remote BIS associated with this Adjacent RIB in the parameter "notificationRemoteBIS-NET".

PacketBomb: generated when the local BIS has been presented with a BISPDU whose source is not one of the BISs adjacent to the local BIS. Such BISPDU are rejected by the local BIS. In addition to the parameters specified by ISO/IEC 10733, the following information will be reported by the AdditionalInformation field with the parameters:

- a) notificationSourceBIS
- b) notificationSourceBISrdi
- c) notificationSourceBISrdc

These parameters are created from the OPEN PDU values:

- a) notificationSourceBIS--NET of remote BIS sending packet bomb

- b) notificationSourceBISrdi--RDI of remote BIS sending packet bomb
- c) notificationSourceBISrdc--RDC information for remote BIS sending packet bomb

EnterFSMStateMachine: generated when the IDRP FSM state machine used to communicate with another BIS is started. The RemoteBis-NET is reported in the additionalInformation field using the "notificationRemoteBIS-NET" parameter. The significance of the sub-parameter of each item of AdditionalInformation shall be set to the value FALSE (that is, not significant) so that a managing system that receives the event report will be less likely to reject it.;

iDRPBasicImportedInfoNotifications-B

BEHAVIOUR DEFINED AS Imports the communicationsInformation notification from ISO/IES 10165-2. It is used to report the following protocol events:

EnterFSMState: generated when a BIS starts the IDRP FSM state machine to establish a connection with a remote BIS. The RemoteBis-NET is reported in the AdditionalInformation field using the "notificationRemoteBIS-NET" parameter. The significant subparameter of each item of AdditionalInformation shall be set to "false" (that is, not significant) so that a managing system receiving the event report will be less likely to reject it.

FSMStateChange: generated when the IDRP FSM used to communicate with another BIS transitions from one state to another. The RemoteBis-NET is reported in the AdditionalInformation field using the "notificationRemoteBIS-NET" parameter. The significant sub-parameter of each item of AdditionalInformation shall be set to "false" (that is, not significant) so that a managing system receiving the event report will be less likely to reject it.;

ATTRIBUTES

internalBIS GET-REPLACE,
 intraIS GET-REPLACE,
 externalBISNeighbor GET-REPLACE,
 internalSystems GET-REPLACE,
 localRDI GET-REPLACE,
 rdcConfig GET-REPLACE,
 localSNPA GET-REPLACE,
 multiExit GET-REPLACE,
 routeServer GET-REPLACE,
 maximumPDUSize GET,
 holdTime GET,
 outstandingPDUs GET,
 authenticationCode GET,
 retransmissionTime GET,
 closeWaitDelayPeriod GET,
 rdTransitDelay GET,
 rdLRE GET,
 locExpense GET,
 ribAttsSet GET,
 capacity GET,
 priority GET,
 version GET,
 maxRIBIntegrityCheck GET;

ACTIONS

startEvent,
stopevent;

NOTIFICATIONS "REC X.721 | ISO/IEC 10165-2:1992": communicationsAlarm

notificationRemoteBIS-NET
notificationBISpduerrorcode
notificationBISerrorssubcode
notificationBISpduerrorinfo
notificationRemoteRDCconfig
notificationLocalRDCconfig
notificationMaxAdjRibIntegritycheck
notificationSourceBis,

"REC X.723 | ISO/IEC 10165-5: 1992": communicationsInformation

notificationRemoteBIS-NET;;;

REGISTERED AS {IDRP.moi iDRPConfig (1)};

12.3 Adjacent BIS Peer Managed Objects

adjacentBIS MANAGED OBJECT CLASS

DERIVED FROM "Rec. X.721 | ISO/IEC 10165-2 : 1992":top;

CHARACTERIZED BY adjacentBISpkg PACKAGE

ATTRIBUTES

bisNet GET,
bisRDI GET,
bisRDC GET,
bisNegotiatedVersion GET,
isPeerSNPAs GET,
authenticationType GET,
state GET,
lastSeqNoSent GET,
lastSeqNoRecv GET,
lastAckSent GET,
lastAckRecv GET,
updatesIn GET,
updatesOut GET,
totalBISPDUsIn GET,
totalBISPDUsOut GET,
keepAlivesSinceLastUpdate GET;

ATTRIBUTE GROUPS

counters
 updatesIn
 updatesOut
 totalBISPDUsIn
 totalBISPDUsOut
 keepAlivesSinceLastUpdate,
stateInfo
 state
 lastSeqNoSent
 lastSeqNoRecv
 lastAckSent
 lastAckRecv;;;

REGISTERED AS { moi adjacentBIS(2) }; h2 id=attdef.Attribute Definitions

internalBIS ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.BISGroup;

MATCHES FOR EQUALITY;

BEHAVIOUR internalBIS-B

BEHAVIOUR DEFINED AS The set of NETs which identify the BISs in this
routing domain;;

REGISTERED AS { atoi internalBIS(1) };

intraIS ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.BISGroup;
 MATCHES FOR EQUALITY;
 BEHAVIOUR intraIS-B
 BEHAVIOUR DEFINED AS The set of NETs of the ISs to which the local
 BIS may deliver an inbound NPDU whose destination lies within the
 BIS's routing domain. These ISs must be located on the same common
 subnetwork as this local BIS, and must be capable of delivering NPDUs
 to destinations that are located within the local BIS's routing
 domain.;;
 REGISTERED AS { atoi intraIS(2) };

externalBISNeighbor ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.BISGroup;
 MATCHES FOR EQUALITY;
 BEHAVIOUR externalBISNeighbor-B
 BEHAVIOUR DEFINED AS The set of NETs which identify the BISs in adja-
 cent routing domain that are reachable via a single subnetwork hop.;;
 REGISTERED AS { atoi externalBISNeighbor(3) };

internalSystems ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.SystemIdGroup;
 MATCHES FOR EQUALITY;
 BEHAVIOUR internalSystems-B
 BEHAVIOUR DEFINED AS The set of NETs and NSAPS which identify the
 systems in this routing domain which the BIS uses to construct
 network layer reachability information;;
 REGISTERED AS { atoi internalSystems(4) };

localRDI ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Rdi;
 MATCHES FOR EQUALITY;
 BEHAVIOUR localRDI-B
 BEHAVIOUR DEFINED AS The Routing Domain Identifier for the routing
 domain where this BIS is located;;
 REGISTERED AS { atoi localRDI(5) };

rdcConfig ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Rdcgroup;
 MATCHES FOR EQUALITY;
 BEHAVIOUR rdcConfig-B
 BEHAVIOUR DEFINED AS All of the Routing Confederations to which the
 RD of this BIS belongs and the nesting relationships that are in
 force between them. The nesting relationships are described as a
 sequence of sets of RDC Identifiers;;
 REGISTERED AS { atoi rdcConfig(6) };

localSNPA ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.LocalSNPAs;
 MATCHES FOR EQUALITY;
 BEHAVIOUR localSNPA-B
 BEHAVIOUR DEFINED AS The list of SNPAs of this BIS;;
 REGISTERED AS { atoi localSNPA(7) };

multiExit ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Boolean;
 MATCHES FOR EQUALITY;
 BEHAVIOUR multiExit-B
 BEHAVIOUR DEFINED AS The indication whether this BIS will use the
 MULTI_EXIT_DISC attribute to decide between otherwise identical
 routes. The multiExit parameter is used as the default value for the
 "multi_exit_disc" function in policy decisions.;;

REGISTERED AS { atoi multiExit(8) };

routeServer ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Boolean;

MATCHES FOR EQUALITY;

BEHAVIOUR routeServer-B

BEHAVIOUR DEFINED AS The indication whether this BIS may set the "IDRP_Server_Allowed" field in the NEXT_HOP attribute to X"FF" for BIS to BIS UPDATE BISPDU's. If this variable is true then in accordance with local policy, the IDRP_Server_Allowed field may be set on some UPDATE BISPDU's that this BIS sends. If this attribute is set to false, then no UPDATE BISPDU's will be sent by this BIS with NEXT_HOP attributes containing an "IDRP_Server flag" equal to X"FF".;

REGISTERED AS { atoi routeServer(9) };

maximumPDUSize ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.MaximumPDUSize;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR maximumPDUSize-B

BEHAVIOUR DEFINED AS The maximum number of octets that this BIS is able to handle in an incoming BISPDU;

REGISTERED AS { atoi maximumPDUSize(10) };

holdTime ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Holdtime;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR holdTime-B

BEHAVIOUR DEFINED AS The maximum number of seconds that may elapse between the receipt of two successive BISPDU's of any of the following types: KEEPALIVE, UPDATE, RIB CHECKSUM PDU's or RIB REFRESH PDU's;;

REGISTERED AS { atoi holdTime(11) };

outstandingPDU's ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.OutstandingPdu's;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR outstandingPDU's-B

BEHAVIOUR DEFINED AS The maximum number of BISPDU's that may be sent to this BIS without receiving an acknowledgement;;

REGISTERED AS { atoi outstandingPDU's(12) };

authenticationCode ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.AuthenticationCode;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR authenticationCode-B

BEHAVIOUR DEFINED AS Indication of which authentication mechanism will be used;;

REGISTERED AS { atoi authenticationCode(13) };

retransmissionTime ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.RetransmissionTime;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR retransmissionTime-B

BEHAVIOUR DEFINED AS The Number of seconds of between KEEPALIVE messages if no other traffic is sent;;

REGISTERED AS { atoi retransmissionTime(14) };

closeWaitDelayPeriod ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.CloseWaitDelayPeriod;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR closeWaitDelayPeriod-B

BEHAVIOUR DEFINED AS The number of seconds the local system shall stay in the CLOSE-WAIT state prior to changing to the CLOSED state.;;

REGISTERED AS { atoi closeWaitDelayPeriod(15) };

rdTransitDelay ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.RDTransitDelay;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR rdTransitDelay-B

BEHAVIOUR DEFINED AS The estimated average delay across a Routeing Domain in units of 500ms.;;

REGISTERED AS { atoi rdTransitDelay(16) };

rdLRE ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Rdlre;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR rdLRE-B

BEHAVIOUR DEFINED AS The average error rate of a Routeing Domain in units of an integer which if divided by $2^{32}-1$ will provided the actual probability of the error.;;

REGISTERED AS { atoi rdLRE(17) };

locExpense ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.LocExpense;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR locExpense-B

BEHAVIOUR DEFINED AS The monetary expense of transiting this Routeing Domain. The attribute contains an indication of cost and the units in which it is calculated.;;

REGISTERED AS { atoi locExpense(18) };

ribAttsSet ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.RibAttsSet;

MATCHES FOR EQUALITY;

BEHAVIOUR ribAttsSet-B

BEHAVIOUR DEFINED AS The set of Rib Attributes supported by this BIS.;;

REGISTERED AS { atoi ribAttsSet(19) };

capacity ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Capacity;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR capacity-B

BEHAVIOUR DEFINED AS The traffic carrying capacity of this Routeing Domain.;;

REGISTERED AS { atoi capacity(20) };

priority ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Priority;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR priority-B

BEHAVIOUR DEFINED AS The lowest value of ISO 8473 priority parameter that this RD will provide forwarding services for.;;

REGISTERED AS { atoi priority(21) };

bisNet ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.BisNet;

MATCHES FOR EQUALITY;

BEHAVIOUR bisNet-B

BEHAVIOUR DEFINED AS The NET of the remote BIS of this BIS to BIS connection.;;

REGISTERED AS { atoi bisNet(22) };

bisRDI ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Rdi;

MATCHES FOR EQUALITY;
 BEHAVIOUR bisRDI-B
 BEHAVIOUR DEFINED AS The RDI of the remote BIS of this BIS to BIS
 connection.;;
 REGISTERED AS { atoi bisRDI(23) };

bisRDC ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.Rdcgroup;
 MATCHES FOR EQUALITY;
 BEHAVIOUR bisRDC-B
 BEHAVIOUR DEFINED AS "The RDC the remote BIS belongs to in this BIS
 to BIS connection.";;
 REGISTERED AS { atoi bisRDC(24) };

bisNegotiatedVersion ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.BisNegotiatedVersion;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR bisNegotiatedVersion-B
 BEHAVIOUR DEFINED AS The negotiated version of IDRP protocol this
 BIS to BIS connection is using.;;
 REGISTERED AS { atoi bisNegotiatedVersion(25) };

bisPeerSNPAs ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.BisPeersSNPAs;
 MATCHES FOR EQUALITY;
 BEHAVIOUR bisPeerSNPAs-B
 BEHAVIOUR DEFINED AS The SNPAs announced by the remote BIS of this
 BIS to BIS connection.;;
 REGISTERED AS { atoi bisPeerSNPAs(26) };

authenticationType ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.Authtype;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR authenticationType-B
 BEHAVIOUR DEFINED AS The authentication type the remote BIS sent in
 the OPEN BISPDU in this BIS to BIS connection.;;
 REGISTERED AS { atoi authenticationType(27) };

state ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.State;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR state-B
 BEHAVIOUR DEFINED AS The current state of BIS to BIS communication in
 the local BIS.;;
 REGISTERED AS { atoi state(28) };

lastSeqNoSent ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR lastSeqNoSent-B
 BEHAVIOUR DEFINED AS The last sequence number sent to the remote BIS
 from this local BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi lastSeqNoSent(29) };

lastSeqNoRecv ATTRIBUTE
 WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR lastSeqNoRecv-B
 BEHAVIOUR DEFINED AS The last sequence number received from the
 remote BIS by this local BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi lastSeqNoRecv(30) };

lastAckSent ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR lastAckSent-B
 BEHAVIOUR DEFINED AS The number of the last ack sent to the remote
 BIS from this local BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi lastAckSent(31) };

lastAckRecv ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR lastAckRecv-B
 BEHAVIOUR DEFINED AS The number of the last ack received from the
 remote BIS by this local BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi lastAckRecv(32) };

updatesIn ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR updatesIn-B
 BEHAVIOUR DEFINED AS The number of UPDATE BISPDU's received by this
 BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi updatesIn(33) };

updatesOut ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR updatesOut-B
 BEHAVIOUR DEFINED AS The number of UPDATE BISPDU's sent by this BIS on
 this BIS to BIS connection.;;
 REGISTERED AS { atoi updatesOut(34) };

totalBISPDU'sIn ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR totalBISPDU'sIn-B
 BEHAVIOUR DEFINED AS The number of BISPDU's received by this BIS from
 the remote BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi totalBISPDU'sIn(35) };

totalBISPDU'sOut ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR totalBISPDU'sOut-B
 BEHAVIOUR DEFINED AS The number of BISPDU's received by this BIS from
 the remote BIS on this BIS to BIS connection.;;
 REGISTERED AS { atoi totalBISPDU'sOut(36) };

keepAlivesSinceLastUpdate ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.NonWrappingCounter;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR keepAlivesSinceLastUpdate-B
 BEHAVIOUR DEFINED AS The number of KEEPALIVE BISPDU's received by this
 BIS from the remote BIS since this last UPDATE BISPDU.;;
 REGISTERED AS { atoi keepAlivesSinceLastUpdate(37) };

version ATTRIBUTE

WITH ATTRIBUTE SYNTAX IDRP.Version;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR version-B
 BEHAVIOUR DEFINED AS The version of IDRP protocol this machine
 defaults to using.;;

```

REGISTERED AS { atoi version(38) };

maxRIBIntegrityCheck ATTRIBUTE
  WITH ATTRIBUTE SYNTAX IDRP.MaxRIBIntegrityCheck;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR maxRIBIntegrityCheck-B
    BEHAVIOUR DEFINED AS The maximum time in seconds between checking of
    the Adj-RIBs-In by a local mechanism. If corrupt Adj-RIB-In is
    detected, the BIS shall purge the offending Adj-RIB-In;;
  REGISTERED AS { atoi maxRIBIntegrityCheck(39) };

maxRIBIntegrityTimer ATTRIBUTE
  DERIVED FROM timer;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR maxRIBIntegrityTimer-B
    BEHAVIOUR DEFINED AS The timer that measures in seconds the time
    remaining until the Adj-RIBs-In must be checked by a local mechanism.
    If a corrupt Adj-RIB-In is detected, the BIS shall purge the
    offending Adj-RIB-In;;
  REGISTERED AS { atoi maxRIBIntegrityTimer(40) };

closeWaitDelayTimer ATTRIBUTE
  DERIVED FROM timer;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR closeWaitDelayTimer-B
    BEHAVIOUR DEFINED AS The timer that measures in seconds the time that
    has elapsed since the BIS FSM entered the CLOSE-WAIT state. Upon
    timer expiration, the BIS FSM will enter the CLOSED state;;
  REGISTERED AS { atoi closeWaitDelayTimer(41) };

keepAliveTimer ATTRIBUTE
  DERIVED FROM timer;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR keepAliveTimer-B
    BEHAVIOUR DEFINED AS The timer that measures in seconds the time that
    has elapsed since the previous KEEPALIVE PDU was received by the
    local BIS. Upon its expiration, the BIS will send a BISPDU to its
    peer BIS;;
  REGISTERED AS { atoi keepAliveTimer(42) };

minRouteAdvertisementTimer ATTRIBUTE
  DERIVED FROM timer;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR minRouteSelectionTimer-B
    BEHAVIOUR DEFINED AS The timer that measures in seconds the time that
    has elapsed since the advertisement by the local BIS of a better
    route that was received from a BIS located in another routing
    domain. See clause 8.16.3.1. The minimum value is 5 seconds,
    and the maximum value is 1800 seconds.
  REGISTERED AS { atoi minRouteSelectionTimer(43) };

minRDOriginationTimer ATTRIBUTE
  DERIVED FROM timer;
  MATCHES FOR EQUALITY, ORDERING;
  BEHAVIOUR minRDOriginationTimer-B
    BEHAVIOUR DEFINED AS The timer that measures in seconds the time that
    has elapsed since the advertisement by the local BIS of an UPDATE PDU
    that reported changes within the local BIS's routing domain. See
    clause -- Heading 'MINORG' unknown --;;
  REGISTERED AS { atoi minRDOriginationTimer(44) };

maxCPUOverloadTimer ATTRIBUTE

```

DERIVED FROM timer;
 MATCHES FOR EQUALITY, ORDERING;
 BEHAVIOUR maxCPUOverloadTimer-B
 BEHAVIOUR DEFINED AS The timer that measures in seconds the time that
 has elapsed since the local BIS has detected that its CPU has become
 overloaded. See Annex -- Heading 'CPUOLD' unknown --;;
 REGISTERED AS { atoi maxCPUOverloadTimer(45) };

12.4 Action Definitions

startEvent ACTION
 BEHAVIOUR
 startEvent-B BEHAVIOUR
 DEFINED AS The request to start communication with a remote BIS peer;;
 MODE CONFIRMED;
 WITH INFORMATION SYNTAX IDRP.ActionInfo;
 WITH REPLY SYNTAX IDRP.StartEventreply;
 PARAMETERS remoteBISNetEventInfo;;
 REGISTERED AS { acoi startEvent(1) };

stopEvent ACTION
 BEHAVIOUR
 stopEvent-B BEHAVIOUR
 DEFINED AS The request to stop communication with a remote BIS peer;;
 MODE CONFIRMED;
 WITH INFORMATION SYNTAX IDRP.ActionInfo;
 WITH REPLY SYNTAX IDRP.StopEventreply;
 PARAMETERS remoteBISNetEventInfo;;
 REGISTERED AS { acoi stopEvent(2) };

12.5 Notification Definitions

enterFSMStateMachine NOTIFICATION
 BEHAVIOUR enterFSMStateMachine-B
 BEHAVIOUR DEFINED AS The indication of starting the FSM state machine
 to establish a connection with a remote BIS.;;
 PARAMETERS remoteBISNetEventInfo;
 WITH INFORMATION SYNTAX IDRP.NotificationInfo;
 REGISTERED AS { noi enterFSMStateMachine(1) };

fsmStateChange NOTIFICATION
 BEHAVIOUR fsmStateChange-B
 BEHAVIOUR DEFINED AS The indication of transiting from one state to
 another in the IDRP connection state machine in communication with
 another BIS.;;
 PARAMETERS remoteBISNetActionReply;
 WITH INFORMATION SYNTAX IDRP.NotificationInfo;
 REGISTERED AS { noi fsmStateChange(2) };

errorBISPDUsent NOTIFICATION
 BEHAVIOUR errorBISPDUsent-B
 BEHAVIOUR DEFINED AS The indication of an error in the format of
 BISPDU.;;
 PARAMETERS remoteBISNetEventInfo, bisPDUErrorCode,
 bisPDUErrorSubCode, bisPDUErrorInfo;
 WITH INFORMATION SYNTAX IDRP.NotificationInfo;
 REGISTERED AS { noi errorBISPDUsent(3) };

openBISPDURDCerror NOTIFICATION
 BEHAVIOUR openBISPDURDCerror-B
 BEHAVIOUR DEFINED AS The indication that an OPEN PDU has been

received with the RDC Config for remote BIS and this BIS do not indicate that the two BIS trying to establish a connection are a part of the same confederations;;

PARAMETERS remoteBISNetEventInfo, remoteRDCCConfig, localRDCCConfig;
WITH INFORMATION SYNTAX IDRP.NotificationInfo;
REGISTERED AS { noi errorPDURDCError(4) };

errorBISPDUConnectionClose NOTIFICATION

BEHAVIOUR errorBISPDUConnectionClose-B

BEHAVIOUR DEFINED AS The indication that an ERROR BISPDU has been received from a remote BIS;;

PARAMETERS remoteBISNetEventInfo, bisPDUErrorCode,
bisPDUErrorSubCode,bisPDUErrorInfo;
WITH INFORMATION SYNTAX IDRP.NotificationInfo;
REGISTERED AS { noi errorBISPDUConnectionClose(5) };

corruptAdjRIBIn NOTIFICATION

BEHAVIOUR corruptAdjRIBIn-B

BEHAVIOUR DEFINED AS The indication that the local method of checking the Adj-RIB-In has found an error. All Adj-RIBs-In are being purged.;;

PARAMETERS maxRIBIntegrityCheck;
WITH INFORMATION SYNTAX IDRP.NotificationInfo;
REGISTERED AS { noi corruptAdjRIBIn(6) };

packetBomb NOTIFICATION

BEHAVIOUR packetBomb-B

BEHAVIOUR DEFINED AS The indication that the local BIS has been presented with a BISPDU whose source is not one of the BISs adjacent to the local BIS. Such BISPDUs are rejected by the local BIS.;;

WITH INFORMATION SYNTAX IDRP.NotificationInfo;
REGISTERED AS { noi packetBomb(7) };

12.6 Parameter Definitions

remoteBISNetActionReply PARAMETER

CONTEXT ACTION-REPLY;

WITH SYNTAX IDRP.RemoteBISNetActionReply;

BEHAVIOUR remoteBISNetActionReply-B

BEHAVIOUR DEFINED AS The NET of the Remote BIS that this local BIS is starting IDRP protocol communication with.;;

REGISTERED AS { poi remoteBISNetActionReply(1) };

remoteBISNetEventInfo PARAMETER

CONTEXT EVENT-INFO;

WITH SYNTAX IDRP.RemoteBISNetActionReply;

BEHAVIOUR remoteBISNetEventInfo-B

BEHAVIOUR DEFINED AS The NET of the Remote BIS that this local BIS is starting IDRP protocol communication with.;;

REGISTERED AS { poi remoteBISNetEventInfo(2) };

stateEventInfo PARAMETER

CONTEXT EVENT-INFO;

WITH SYNTAX IDRP.State;

BEHAVIOUR stateEventInfo-B

BEHAVIOUR DEFINED AS The state of the local BIS Finite state machine.;;

REGISTERED AS { poi stateEventInfo(3) };

bisPDUErrorCode PARAMETER

CONTEXT EVENT-INFO;

WITH SYNTAX IDRP.BispduErrorCode;
 BEHAVIOUR bisPDUErrCode-B
 BEHAVIOUR DEFINED AS The error code indicating what type of error
 occurred in the BIS PDU.;;
 REGISTERED AS { poi bisPDUErrCode(4) };

bisPDUErrSubCode PARAMETER
 CONTEXT EVENT-INFO;
 WITH SYNTAX IDRP.BispduErrorSubcode;
 BEHAVIOUR bisPDUErrSubCode-B
 BEHAVIOUR DEFINED AS The error code indicating what type of error
 within the major error type occurred in the BIS PDU.;;
 REGISTERED AS { poi bisPDUErrSubCode(5) };

bisPDUErrInfo PARAMETER
 CONTEXT EVENT-INFO;
 WITH SYNTAX IDRP.BispduErrorInfo;
 BEHAVIOUR bisPDUErrInfo-B
 BEHAVIOUR DEFINED AS The additional information from original PDU
 that indicated an error in the BIS PDU.;;
 REGISTERED AS { poi bisPDUErrInfo(6) };

remoteRDCCConfig PARAMETER
 CONTEXT EVENT-INFO;
 WITH SYNTAX IDRP.RemoteRDCCConfig;
 BEHAVIOUR remoteRDCCConfig-B
 BEHAVIOUR DEFINED AS The Routing Domain Confederation (RDC) informa-
 tion from the remote BIS on this BIS to BIS communication.;;
 REGISTERED AS { poi remoteRDCCConfig(7) };

localRDCCConfig PARAMETER
 CONTEXT EVENT-INFO;
 WITH SYNTAX IDRP.LocalRDCCConfig;
 BEHAVIOUR localRDCCConfig-B
 BEHAVIOUR DEFINED AS The Routing Domain Confederation (RDC) informa-
 tion from this local BIS on this BIS to BIS communication.;;
 REGISTERED AS { poi localRDCCConfig(8) };

maxAdjRIBIntegrityCheck PARAMETER
 CONTEXT EVENT-INFO;
 WITH SYNTAX IDRP.MaxRIBIntegrityCheck;
 BEHAVIOUR maxAdjRIBIntegrityCheck-B
 BEHAVIOUR DEFINED AS The maximum number of integrity checks
 detected before reporting the event to systems management;;
 REGISTERED AS { poi maxAdjRIBIntegrityCheck(9) };

12.7 Attribute Groups

counters ATTRIBUTE GROUP
 DESCRIPTION The group of all counter per BIS connection;
 REGISTERED AS { agoi counters(1) };

stateInfo ATTRIBUTE GROUP
 DESCRIPTION The group of all state information per BIS connection;
 REGISTERED AS { agoi stateinfo(2) };

12.8 ASN.1 MODULES

```

IDRP
DEFINITIONS::=BEGIN
  -- object identifier definitions
  sc6 OBJECT IDENTIFIER ::= {joint-iso-ccitt standard 10747 }
  -- value to be assigned by SC21 secretariat
  idrpoi OBJECT IDENTIFIER ::= {joint-iso-ccitt standard
    10733 iDRP(1)} -- value to be assigned by SC6 secretariat
  sseoi OBJECT IDENTIFIER ::= {idrpoi standSpecificExtensions(0)}
  moi OBJECT IDENTIFIER ::= {idrpoi objectClass (3)}
  poi OBJECT IDENTIFIER ::= {idrpoi package (4)}
  proi OBJECT IDENTIFIER ::= {idrpoi parameter(5)}
  nboi OBJECT IDENTIFIER ::= {idrpoi nameBinding (6)}
  aoj OBJECT IDENTIFIER ::= {idrpoi attribute (7)}
  agoi OBJECT IDENTIFIER ::= {idrpoi attributeGroup (8)}
  acoi OBJECT IDENTIFIER ::= {idrpoi action (9)}
  noi OBJECT IDENTIFIER ::= {idrpoi notification (10)}

  --
  --object identifiers for notification parameters
  --

  se OBJECT IDENTIFIER ::= {sseoi specificProblems(3)}

  errorBISPDUsent OBJECT IDENTIFIER ::= {se errorBISPDU0(0)}
  openBISpduRDCerror OBJECT IDENTIFIER ::= {se errorBISPDU1(1)}
  errorBISPDUconnectionclose OBJECT IDENTIFIER ::= {se
    errorBISPDU2(2)}
  corruptAdjRIBIn OBJECT IDENTIFIER ::= {se errorBISPDU3(3)}
  packetBomb OBJECT IDENTIFIER ::= {se errorBISPDU4(4)}
  enterFSMstate OBJECT IDENTIFIER ::= {se errorBISPDU5(5)}
  fSMStateChange OBJECT IDENTIFIER ::= {se errorBISPDU6(6)}

  --
  --ASN1 Types and Values
  --

  ActionInfo ::=SET OF Parameter
  ActionReply ::= SEQUENCE {
    responseCode OBJECT IDENTIFIER,
    responseArgs SET OF Parameter OPTIONAL}
  AuthenticationCode ::=ENUMERATED{
    integrityOnly(0),
    integrityPlusAuthentication(1)}
  Authtype ::=AuthenticationCode
  BISGroup ::= SET OF NetworkEntityTitle
  BisNet ::= NetworkEntityTitle
  BisNegotiatedVersion ::=Version
  BispduErrorCode::= ENUMERATED {
    oPENPDUErrors (1),
    uDATEPDUErrors (2),
    holdtimerExpired (3)}
  BispduErrorSubcode ::= CHOICE {
    operr [ ] IMPLICIT Openerrorsubcode,
    uperr [1] IMPLICIT Updateerrorsubcode}

  BispduErrorInfo ::=OCTET STRING(SIZE(1..50))
  --50 bytes of original message are saved

  --EDITOR'S NOTE: Comment is requested on the amount of
  --      data that should be saved.

```

```

BisPeersSNPAs ::= SNPAAddresses
Boolean ::= BOOLEAN
Capacity ::= INTEGER(1..255)
CloseWaitDelayPeriod ::= INTEGER(150)
Destinationspecificqos ::= Ribattsec
Destinationspecificsecurity ::= Ribattsec
EndSystemNSAP ::= OCTET STRING(SIZE(1..20))
Expensevalue ::= Locexpense
Holdtime ::= INTEGER(1..65535)
KeepaliveSincelastupdate ::= INTEGER(1..4294967295)
Keepalivetimer ::= Timer
Lastseqnosent ::= INTEGER(1..4294967295)
Lastseqnorecv ::= INTEGER(1..4294967295)
Lastacksent ::= INTEGER(1..4294967295)
Lastackrecv ::= INTEGER(1..4294967295)
LocExpense ::= INTEGER(1..65535)
LocalRDCCConfig ::= Rdcgroup
LocalSNPAs ::= SNPAAddresses
MaximumPDUSize ::= INTEGER(1..65535)
MaxRIBIntegrityCheck ::= INTEGER(1..65535)
Metriclength ::= INTEGER(1..255)
Metricvalue ::= OCTET STRING(SIZE(1..255))
NSAPprefixLength ::= INTEGER(1..160)
NSAPprefix ::= BIT STRING(SIZE(1..160))
NetworkEntityTitle ::= OCTET STRING(SIZE(1..20))
NonWrappingCounter ::= INTEGER(0..18446744.73709551616)
--0..(2 exp 64)-1
NotificationInfo ::= SET OF Parameter
Openerrorsubcode ::= ENUMERATED {
    unsupportedVersionnumber (1),
    badMaxPDUSize (2),
    badOutstandingPDUs (3),
    badPeerRD (4),
    unsupportedAuthenticationcode (5),
    authenticationFailure (6),
    badRIB-AttrsSet (7),
    rDCmismatch (8)}
OutstandingPdus ::= INTEGER(0..255)
Parameter ::= SEQUENCE {
    paramID OBJECT IDENTIFIER,
    paramInfo ANY DEFINED BY paramID}
Priority ::= INTEGER(0..14)
QOSlength ::= INTEGER(1..255)
QOSvalue ::= OCTET STRING(SIZE(1..255))
Rdi ::= OCTET STRING(SIZE(1..20))
--assigned from the NSAP address space
Rdcgroup ::= SEQUENCE{confed Rdcsetid, members SET OF Rdi}
Rdcsetid ::= INTEGER(1..255)
RDTransitDelay ::= INTEGER(0..65535)
Rdlre ::= INTEGER(0..4294967295)
RetransmissionTime ::= INTEGER(0..65535)
RemoteBIS-NET ::= NetworkEntityTitle
RemoteRDCCConfig ::= Rdcgroup
RemoteBISNetActionReply ::= SEQUENCE{
    responseCode OBJECT IDENTIFIER,
    responseArgs SET OF Parameter OPTIONAL}
RibAttsSet ::= SEQUENCE { confed Ribsetid,
    count Ribsetcount,
    attribs SET OF Ribattributes}
Ribsetid ::= INTEGER(1..255)
Ribsetcount ::= INTEGER(0..255)
Ribattributes ::= SEQUENCE {

```

```

    attrib Ribattribute,
    value Ribvalue OPTIONAL}
Ribattribute ::= ENUMERATED {
    tRANSITDELAY (9),
    rESIDUALERROR (10),
    eXPENSE (11),
    sourceSpecificQOS (12),
    destinationSpecificQOS (13),
    sourceSpecificSecurity (17),
    destinationSpecificSecurity(18),
    capacity (19),
    priority (20)}

Ribvalue ::= SEQUENCE {length Ribattlength,
    attr Ribattributes}
Ribattlength ::= INTEGER
Ribattvalue ::= CHOICE {
    transitdelayvalue [0] IMPLICIT INTEGER,
    residualerrorvalue [1] IMPLICIT INTEGER,
    expensevalue [2] IMPLICIT INTEGER,
    sourcespecificqos [3] IMPLICIT INTEGER,
    destinationspecificqos [4] IMPLICIT INTEGER,
    sourcespecificsecurity [5] IMPLICIT INTEGER,
    destinationspecificsecurity [6] IMPLICIT INTEGER,
    capacityvalue [7] IMPLICIT INTEGER,
    priorityvalue [8] IMPLICIT INTEGER}
Ribattqos ::= SEQUENCE {
    preflgth NSAPPrefixLength,
    prefix NSAPPrefix,
    qoslgth QOSlength,
    qosval QOSvalue,
    metriclgth Metriclength,
    metricval Metricvalue}
Ribattsec ::= SEQUENCE {
    preflgth NSAPPrefixLength,
    prefix NSAPPrefix,
    seclgth Securitylength,
    secval Securitylevel}
RouteAdvertisementInterval ::=INTEGER(30..900)
--IS 10589 imposes minimum value of 30 seconds
--and maximum value of 900 seconds in clause
--12.2.3.4, part c)
Securitylength ::= INTEGER(0..255)
Securitylevel ::= OCTET STRING(SIZE(1..255))

SNPAaddress ::= OCTET STRING
    (FROM ('1'H|'2'H|'3'H|'4'H|'5'H|'6'H|'7'H|'8'H|'9'H|
        'A'H|'B'H|'C'H|'D'H|'E'H|'F'H))
    --integral number of hexadecimal digits
SNPAaddresses ::= SET OF SNPAaddress
State ::= ENUMERATED {
    closed (0),
    open-recv(1),
    established(2),
    open-sent(3),
    close-wait(4)}
StopEventreply ::= Parameter
StartEventreply ::=Parameter
SystemIdGroup ::= SEQUENCE {
    nETS SET OF NetworkEntityTitle,
    nSAPs SET OF EndSystemNSAP}
Updateerrorsubcode ::=ENUMERATED {

```



```
malformedAttributelist (1),
unrecognizedWell-knownAttribute (2),
missingWell-knownAttribute (3),
attributeFlagsError (4),
attributeLengthError (5),
rDRoutingLoop (6),
invalidNEXTHOPAttribute (7),
optionalAttributeerror (8),
invalidReachabilityInformation (9),
misconfiguredRDCs (10)}
Updatesin ::=INTEGER(1..4294967295)
Updatesout ::=INTEGER(1..4294967295)
Totalbispdusin ::=INTEGER(1..4294967295)
Totalbispdusout ::=INTEGER(1..4294967295)
Version ::=INTEGER (1..255)
Timer ::= Sequence{
    exponent [1] INTEGER(-62..63),
    mantissa [2] integer(0..65535)}
Waitdelattimer ::= Timer
END
```

13. Conformance

A Protocol Implementation Conformance Statement (PICS) shall be completed with respect to any claim for conformance of an implementation to this International Standard. The PICS shall be produced in accordance with the relevant PICS-proforma in Appendix A.

NOTE: Since it is only Boundary ISs that implement the elements of procedure of this international standard, this clause does not address deployment guidelines or addressing guidelines for systems in general. Since distribution of policies is outside the scope of this standard, this topic also is not addressed in the following conformance clauses.

13.1 Static Conformance for All BISs

Each IS claiming conformance to this international standard shall be capable of each of the following:

- a) generating and parsing BISPDU's with the structure and format of 7
- b) transmitting and receiving NSAP prefixes that have been encoded as single values according to 8.1.2.1
- c) generating, recognizing upon receipt, and updating each of the following well-known mandatory path attributes as described in the indicated clauses:
 - 2 — ROUTE-ID in 8.12.1
 - RD_PATH in 8.12.3
 - RD_HOP_COUNT 8.12.4
- d) recognizing upon receipt each of the following well-known discretionary path attributes that are contained in any incoming UPDATE PDU, as described in the indicated clauses:
 - EXT_INFO in 8.12.2
 - NEXT_HOP in 8.12.4
 - DIST_LIST_INCL in 8.12.5
 - DIST_LIST_EXCL in 8.12.6
 - LOCAL_PREF in 8.12.8
 - TRANSIT DELAY in 8.12.9
 - RESIDUAL ERROR in 8.12.10
 - EXPENSE in 8.12.11
 - SOURCE SPECIFIC QOS in 8.12.12
 - DESTINATION SPECIFIC QOS in 8.12.13
 - HIERARCHICAL RECORDING in 8.12.14
 - SOURCE SPECIFIC SECURITY in 8.12.16
 - DESTINATION SPECIFIC SECURITY in 8.12.17
 - CAPACITY in 8.12.18
 - PRIORITY in 8.12.19
- e) utilizing domain configuration information in the format described in 8.3
- f) providing reliable delivery of BISPDU's using the methods of 8.5
- g) establishing, closing, and maintaining BIS-BIS connections in accordance with the procedures of 8.6
- h) responding to error conditions in BISPDU's according to 8.20
- i) negotiating the protocol version number according to 8.7
- j) operating in a "fail-stop" manner in regard to corrupted routing information, according to 8.10.2
- k) maintaining RIBs as described in 8.10.
- l) handling incoming BISPDU's as described in 8.14.
- m) detecting inconsistent routing information in accordance with 8.14.2.
- n) receiving and recognizing information which has been reduced in size according to the methods of 8.17.1.
- o) distributing network reachability information within a routing domain according to 8.16.1.
- p) distributing network reachability information outside a routing domain according to 8.16.2.
- q) selecting routes according to 8.15
- r) forwarding ISO 8473 NPDUs according to 9.
- s) supporting the interface to ISO 8473 using the service primitives according to 10.
- t) providing the managed objects described in 12.
- u) supporting the authentication mechanism described in 8.9.

13.2 Conformance to Optional Functions

13.2.1 Generation of Information in Reduced Form

A BIS that claims to support generation of information in a reduced form shall be capable of producing information in accordance with the reduction techniques of clauses 8.17.1 and 8.17.2.

13.2.2 Supporting RDCs

A BIS that claims to support Routing Domain Confederations shall construct them and distribute their routing information in accordance with clause 8.13.

13.2.3 Generation of Well-known Discretionary Attributes

A BIS that claims to support generation of any of the following well-known discretionary attributes shall do so in accordance with the indicated clauses:

EXT_INFO: 8.12.2

NEXT HOP: 8.12.4

DIST LIST INCL: 8.12.5

DIST LIST EXCL: 8.12.6

LOCAL PREF: 8.12.8

TRANSIT DELAY: 8.12.9

RESIDUAL ERROR: 8.12.10.

EXPENSE: 8.12.11.

SOURCE SPECIFIC QOS: 8.12.12.

DESTINATION SPECIFIC QOS: 8.12.13.

HIERARCHICAL RECORDING: 8.12.14

SOURCE SPECIFIC SECURITY: 8.12.16

DESTINATION SPECIFIC SECURITY: 8.12.17

CAPACITY: 8.12.18

PRIORITY: 8.12.19

13.2.4 Propagation of Well-known Discretionary Attributes

A BIS that claims to support updating and propagation of any of the following well-known discretionary attributes after having received them in an UPDATE PDU shall do so in accordance with the indicated clauses:

EXT_INFO: 8.12.2

NEXT HOP: 8.12.4

DIST LIST INCL: 8.12.5

DIST LIST EXCL: 8.12.6

LOCAL PREF: 8.12.8

TRANSIT DELAY: 8.12.9

RESIDUAL ERROR: 8.12.10.

EXPENSE: 8.12.11.

SOURCE SPECIFIC QOS: 8.12.12.

DESTINATION SPECIFIC QOS: 8.12.13.

HIERARCHICAL RECORDING: 8.12.14

SOURCE SPECIFIC SECURITY: 8.12.16

DESTINATION SPECIFIC SECURITY: 8.12.17

CAPACITY: 8.12.18

PRIORITY: 8.12.19

Annex A. PICS Proforma

(Normative)

A.1 Introduction

The supplier of a protocol implementation which is claimed to conform to International Standard XXX shall complete the applicable Protocol Implementation Conformance Statement (PICS) proforma.

A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use:

- by the protocol implementer, as a check list to reduce the risk of failure to conform to the standard through oversight;
- by the supplier and acquirer—or potential acquirer— of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis of understanding provided by the standard PICS proforma;
- by the user—or potential user—or potential user—of the implementation, as a basis for initially checking the possibility of interworking unit another implementation (note that, while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSs);
- by a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A.2 Abbreviations and Special Symbols

A.2.1 Status Symbols

The following status symbols are used in the PICS:

Symbol	Meaning
M	mandatory
O	optional
O.<n>	optional, but support of at least one of the group of options labelled by the same numeral <n> is required
X	prohibited
c.<cid>	conditional requirement, according to the condition identified by <cid>
<item>	simple-predicate condition, dependent on the support marked for <item>
—	not applicable (N/A)

A.3 Instructions for Completing the PICS Proforma

A.3.1 General Structure of the PICS Proforma

The first part of the PICS proforma—Implementation Identification and Protocol Summary—is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma is a fixed-format questionnaire divided into subclauses, each containing a group of individual items. Answers to the questionnaire items are to be provided in the right-most column, either by simply marking an answer to indicate a restricted choice (usually *Yes* or *No*), or by entering a value or a set or range of values. (Note that there are some items where two or more choices from a set of possible answers can apply: all relevant choices are to be marked.)

Each item is identified by an item reference in the first column; the second column contains the question to be answered; the third column contains the reference or references to the material that specifies the item in the main body of the standard; the remaining columns record the status of the item—whether support is mandatory, optional, or conditional—and provide space for the answers: see also A.3.4 below.

A supplier may also provide—or be required to provide—further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labelled A<i> or X<i>, respectively, for cross-referencing purposes, where <i> is any unambiguous identification for the item (e.g., simply a numeral): there are no other restrictions on its format and presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformance Statement for the implementation in question.

NOTE: Where an implementation is capable of being configured in more than one way, a single PICS may be able to describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering some subset of the implementation's configuration capabilities, in case that makes for easier and clearer presentation of the information.

A.3.2 Additional Information

Items of Additional Information allow a supplier to provide further information intended to assist the interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS can be considered complete without any such information. Examples might be an outline of the

ways in which a (single) implementation can be set up to operate in a variety of environments and configurations, or a brief rationale—based perhaps upon specific application needs—for the exclusion of features which, although optional, are nonetheless commonly present in implementations of the protocol.

References to items of Additional Information may be entered next to any answer in the questionnaire, and may be included in items of Exception Information.

A.3.3 Exception Information

It may occasionally happen that a supplier will wish to answer an item with mandatory or prohibited status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No pre-printed answer will be found in the Support column for this: instead, the supplier is required to write into the Support column an X<i> reference to an item of Exception Information, and to provide the appropriate rationale in the Exception item itself.

An implementation for which an Exception item is required in this way does not conform to ISO/IEC XXX.

NOTE: A possible reason for the situation described above is that a defect in the standard has been reported, a correction for which is expected to change the requirement not met by the implementation.

A.3.4 Conditional Status

A.3.4.1 Conditional Items

The PICS proforma contains a number of conditional items. These are items for which the status that applies—mandatory, optional, or prohibited—is dependent upon whether or not certain other items are supported, or upon the values supported for other items.

In many cases, whether or not the item applies at all is conditional in this way, as well as the status when the item does apply.

Individual conditional items are indicated by a conditional symbol in the Status column as described in A.3.4.2 and A.3.4.3 below. Where a group of items is subject to the same condition for applicability, a separate preliminary question about the condition appears at the head of the group, with an instruction to skip to a later point in the questionnaire if the "Not Applicable" answer is selected.

A.3.4.2 Conditional Symbols and Conditions

A conditional symbol is either of the form C.<n> or C.G<n>, where <n> is a numeral or is an abbreviated condition of the form described in A.3.4.3 below. For the C.<n> form, the numeral identifies a condition appearing in a list as the end of the subclause

containing the item. For the C>G<n> form, the numeral identifies a condition appearing in a list of global conditions.

A simple condition is of the form

if <p1> then <s1> else <s2>

where <p1> is a predicate (see A.3.4.4 below) and <s1> and <s2> are either basic status symbols (M, O, O.<n>, or X) or the symbol "—".

An extended condition is of the form

if <p1> then <s1>
else if <p2> then <s2>
[else if ...]
else <sn>

where the quantities <px> are predicates, and the quantities <sx> are either basic status symbols or "—".

The symbol (either a basic status symbol or "—") applicable to an item governed by a simple condition is <s1> if the predicate of the condition is true, and is <s2> otherwise. The symbol applicable to an item governed by an extended condition is <si> where <pi> is the first true predicate, if any, in the sequence <p1>, <p2>, ...; and it is <sn> if no predicate is true.

A.3.4.3 Abbreviated Conditions

The abbreviated condition <item>:<s> in the status column is equivalent to a conditional symbol with corresponding condition *if <item> then <s> else "—"*.

A.3.4.4 Predicates

A simple predicate in a condition is either:

- a single item reference; or
- a relation containing a comparison operator (=, <, etc.) with one (or both) of its operands being an item reference for an item taking numerical values as its answer. In case (a), the predicate is true if the item referred to is marked as supported, and false otherwise. In case (b), the predicate is true if the relation holds when each item reference is replaced by the value entered in the Support column as the answer to the item referred to.

Compound predicates are boolean expressions constructed by combining simple predicates using the boolean operators AND, OR, and NOT, and parentheses, in the usual way. A compound predicate is true if and only if the boolean expression evaluates to "true" when the simple predicates are interpreted as described above.

Item	Questions/Features	References	Status	N/A	Support
H3	Is ... supported?	42.3(d)	C.2	—	M: Yes ___ O: Yes___ No___

C.2: If A4 then M else if D1 or (B52>2) then O else N/A

Figure 9. Illustrative PICS for Conditional Items

Items whose references are used in predicates are indicated by an asterisk in the Item column.

A.3.4.5 Answering Conditional Items

To answer a conditional item, the predicate(s) of the condition is (are) evaluated as described in A.3.4.4 above, and the applicable symbol is determined as described in A.3.4.2. If the result is "N/A", the Not Applicable answer column is to be marked; otherwise, the Support column is to be completed in the usual way.

When two or more status symbols appear in the condition for an item, the Support column for the item contains one line for each such symbol, labelled by the relevant method. The answer for the item is to be marked in the line labelled by the symbol selected according to the condition (unselected lines may be crossed out for added clarity).

For example, in Figure 9, the N/A column would be marked if neither predicate of C.2 was true; the answer line labelled "M:" would be marked if item A4 was marked as supported; and the answer line labelled "O:" would be marked if item A4 was not marked as supported but item D1 was supported.

A.4 Identification

A.4.1 Implementation Identification: IDRP

Table 7. PICS Proforma for IDRP: Implementation Identification	
Supplier	
Contact point for queries about this PICS	
Implementation Name(s) and Version(s)	
Other information necessary for full identification (e.g., Name's and Version(s) for machines and operating systems, System Name(s))	

NOTES:

- Only the first three items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification.
- The terms *Name* and *Version* should be interpreted appropriately to correspond with a supplier's terminology (using, e.g., Type, Series, MODEL).

A.4.2 Protocol Summary: IDRP

Table 8. PICS Proforma for IDRP: Protocol Summary	
Protocol Version	
Addenda Implemented (if applicable)	
Amendments Implemented	
Have any Exception items been required? (See A.3.3.)	Yes__ No__ NOTE: The answer Yes means that the implementation does not conform to this international standard.)

A.4.3 PICS Proforma: IDRP General

Table 9. PICS Proforma for IDRP: General				
Item	Questions/Features	References	Status	Support
BASIC	Are all basic BIS functions implemented?	13.1	M	Yes __
MGT	Is this system capable of being managed by the specified management information?	12 .	M	Yes __
VER	Does this BIS support version negotiation?	8.7	M	Yes__
RTID	Does this BIS support the ROUTE-ID attribute?	8.12.1	M	Yes__
HOPS	Does this BIS support the RD_HOP_COUNT attribute?	8.12.15	M	Yes__
PATH	Does this BIS support the RD_PATH attribute?	8.12.3.	M	Yes__
FSM	Does this BIS manage BIS-BIS connections according to the BIS FSM description?	8.6	M	Yes__
ERROR	Does this BIS handle error handling for IDRP?	8.20	M	Yes__
RIBCHK	Does this BIS operate in a "fail-stop" manner with respect to corrupted routeing information?	8.10.2	M	Yes__

A.4.4 PICS Proforma: IDRP Update Send Process

Table 10. PICS Proforma for IDRP: Update-Send				
Item	Questions/Features	References	Status	Support
INT	Does this BIS provide the internal update procedures?	8.16.1	M	Yes __
RTSEL	Does this BIS support the MinRouteSelectionInterval timer?	8.16.3.1	M	Yes __
RTORG	Does this BIS support the MinRDOrginationInterval timer?	8.16.3.2	M	Yes __
JITTER	Does this BIS provide jitter on its timers?	8.16.3.3	M	Yes __

A.4.5 PICS Proforma: IDRP Update Receive Process

Table 11. PICS Proforma for IDRP: Update-Receive				
Item	Questions/Features	References	Status	Support
INPDU	Does the BIS handle inbound BISPDU's correctly?	8.14	M	Yes __
INCONS	Does this BIS detect inconsistent routing information?	8.14.2	M	Yes__

A.4.6 PICS Proforma: IDRP Decision Process

Table 12. PICS Proforma for IDRP: Decision				
Item	Questions/Features	References	Status	Support
TIES	Does the BIS break ties between candidate routes correctly?	8.15.2.1	M	Yes __
RIBUPD	Does this BIS update the correct Loc-RIBs?	8.15.2	M	Yes__
AGGRT	Does this BIS support route aggregation?	8.17.2 to 8.17.2.3	O	Yes__ No __
LOCK	Does this BIS provide interlocks between its decision process and the updating of the information in its Adj-RIBs-In?	8.15.4	M	Yes__

A.4.7 PICS Proforma: IDRP Receive Process

Table 13. PICS Proforma for IDRP: Receive				
Item	Questions/Features	References	Status	Support
RCV	Does the BIS process incoming BISPDU's and respond correctly to error conditions?	8.14, 8.20	M	Yes __
OSIZE	Does the BIS accept incoming OPEN PDU's whose size in octets is between minBISPDULength and 3000?	7.2, 8.20	M	Yes __
MXPDU	Does the BIS accept incoming UPDATE, IDRP ERROR and RIB REFRESH PDU's whose size in octets is between minBISPDULength and maxBISPDULength ?	7.2, 8.20	M	Yes __

A.4.8 PICS Proforma: IDRP CLNS Forwarding

Table 14. PICS Proforma for IDRP: CLNS Forwarding				
Item	Questions/Features	References	Status	Support
PSRCRT	Does the BIS correctly handle 8473 NPDU's that contain a partial source route?	9	M	Yes __
DATTS	Does the BIS correctly extract the NPDU-derived Distinguishing Attributes from an 8473 NPDU?	9.2	M	Yes __
MATCH	Does the BIS correctly match the NPDU-derived Distinguishing Attributes with the corresponding FIB-Atts?	9.3	M	Yes __
EXTF	Does the BIS correctly forward NPDU's with destinations outside its own routing domain?	9.4	M	Yes __
INTF	Does the BIS correctly forward NPDU's with destinations inside its own routing domain?	9.1	M	Yes __

A.4.9 PICS Proforma: IDRP Optional Transitive Attributes

Table 15. PICS Proforma for IDRP: Optional Transitive Attributes				
Item	Questions/Features	References	Status	Support
MEXIT	Does the BIS support use of the MULTI-EXIT DISC attribute?	8.12.7	O	Yes ___ No ___

A.4.10 PICS Proforma: IDRP Well Known Discretionary Attributes

Table 16. PICS Proforma for IDRP: Generating Well Known Discretionary Attributes				
Item	Questions/Features	References	Status	Support
EXTG	Does the BIS support generation of the EXT_INFO attribute?	8.12.2	O	Yes ___ No ___
NHRS	Does the BIS support generation of the NEXT_HOP attribute in support of route servers?	8.12.4	O	Yes ___ No ___
NHSN	Does the BIS support generation of the NEXT_HOP attribute to advertise SNPsAs?	8.12.4	O	Yes ___ No ___
DLI	Does the BIS support generation of the DIST_LIST_INCL attribute?	8.12.5	O	Yes ___ No ___
DLE	Does the BIS support generation of the DIST_LIST_EXCL attribute?	8.12.6	O	Yes ___ No ___
LPRF	Does the BIS support generation of the LOCAL_PREF attribute?	8.12.8	O	Yes ___ No ___
TDLY	Does the BIS support generation of the TRANSIT DELAY attribute?	8.12.9	O	Yes ___ No ___
RERR	Does the BIS support generation of the RESIDUAL ERROR attribute?	8.12.10	O	Yes ___ No ___
EXP	Does the BIS support generation of the EXPENSE attribute?	8.12.11	O	Yes ___ No ___
SQOS	Does the BIS support generation of the SOURCE SPECIFIC QOS attribute?	8.12.12	O	Yes ___ No ___
DQOS	Does the BIS support generation of the DESTINATION SPECIFIC QOS attribute?	8.12.13	O	Yes ___ No ___
HREC	Does the BIS support generation of the HIERARCHICAL RECORDING attribute?	8.12.14	O	Yes ___ No ___
SSEC	Does the BIS support generation of the SOURCE SPECIFIC SECURITY attribute?	8.12.16	O	Yes ___ No ___
DSEC	Does the BIS support generation of the DESTINATION SPECIFIC SECURITY attribute?	8.12.17	O	Yes ___ No ___
CAPY	Does the BIS support generation of the CAPACITY attribute?	8.12.18	O	Yes ___ No ___
PRTY	Does the BIS support generation of the PRIORITY attribute?	8.12.19	O	Yes ___ No ___

Table 17. PICS Proforma for IDRP: Propagating Well Known Discretionary Attributes				
Item	Questions/Features	References	Status	Support
EXTGP	Does the BIS support propagation of the EXT_INFO attribute?	8.12.2	M	Yes ___ No ___
NHRSP	Does the BIS support propagation of the NEXT_HOP attribute in support of route servers?	8.12.4	O	Yes ___ No ___
NHSP	Does the BIS support propagation of the NEXT_HOP attribute to advertise SNPs?	8.12.4	O	Yes ___ No ___
DLIP	Does the BIS support propagation of the DIST_LIST_INCL attribute?	8.12.5	O	Yes ___ No ___
DLEP	Does the BIS support propagation of the DIST_LIST_EXCL attribute?	8.12.6	O	Yes ___ No ___
LPRFP	Does the BIS support propagation of the LOCAL_PREF attribute?	8.12.8	O	Yes ___ No ___
TDLYP	Does the BIS support propagation of the TRANSIT DELAY attribute?	8.12.9	O	Yes ___ No ___
RERRP	Does the BIS support propagation of the RESIDUAL ERROR attribute?	8.12.10	O	Yes ___ No ___
EXPP	Does the BIS support propagation of the EXPENSE attribute?	8.12.11	O	Yes ___ No ___
SQOSP	Does the BIS support propagation of the SOURCE SPECIFIC QOS attribute?	8.12.12	O	Yes ___ No ___
DQOSP	Does the BIS support propagation of the DESTINATION SPECIFIC QOS attribute?	8.12.13	O	Yes ___ No ___
HRECP	Does the BIS support propagation of the HIERARCHICAL RECORDING attribute?	8.12.14	O	Yes ___ No ___
SSECP	Does the BIS support propagation of the SOURCE SPECIFIC SECURITY attribute?	8.12.16	O	Yes ___ No ___
DSECP	Does the BIS support propagation of the DESTINATION SPECIFIC SECURITY attribute?	8.12.17	O	Yes ___ No ___
CAPYP	Does the BIS support propagation of the CAPACITY attribute?	8.12.18	O	Yes ___ No ___
PRTYP	Does the BIS support propagation of the PRIORITY attribute?	8.12.19	O	Yes ___ No ___

Table 18. PICS Proforma for IDRP: Receiving Well Known Discretionary Attributes				
Item	Questions/Features	References	Status	Support
EXTR	Does the BIS recognize upon receipt the EXT_INFO attribute?	8.12.2	M	Yes ___ No ___ X:
NHRSR	Does the BIS recognize upon receipt the NEXT_HOP attribute?	8.12.4	M	Yes ___ No ___ X:
DLIR	Does the BIS recognize upon receipt the DIST_LIST_INCL attribute?	8.12.5	M	Yes ___ No ___ X:
DLER	Does the BIS recognize upon receipt the DIST_LIST_EXCL attribute?	8.12.6	M	Yes ___ No ___ X:
LPRFR	Does the BIS recognize upon receipt the LOCAL_PREF attribute?	8.12.8	M	Yes ___ No ___ X:
TDLYR	Does the BIS recognize upon receipt the TRANSIT DELAY attribute?	8.12.9	M	Yes ___ No ___ X:
RERRR	Does the BIS recognize upon receipt the RESIDUAL ERROR attribute?	8.12.10	M	Yes ___ No ___ X:
EXPR	Does the BIS recognize upon receipt the EXPENSE attribute?	8.12.11	M	Yes ___ No ___ X:
SQOSR	Does the BIS recognize upon receipt the SOURCE SPECIFIC QOS attribute?	8.12.12	M	Yes ___ No ___ X:
DQOSR	Does the BIS recognize upon receipt the DESTINATION SPECIFIC QOS attribute?	8.12.13	M	Yes ___ No ___ X:
HRECR	Does the BIS recognize upon receipt the HIERARCHICAL RECORDING attribute?	8.12.14	M	Yes ___ No ___ X:
SSECR	Does the BIS recognize upon receipt the SOURCE SPECIFIC SECURITY attribute?	8.12.16	M	Yes ___ No ___ X:
DSECR	Does the BIS recognize upon receipt the DESTINATION SPECIFIC SECURITY attribute?	8.12.17	M	Yes ___ No ___ X:
CAPYR	Does the BIS recognize upon receipt the CAPACITY attribute?	8.12.18	M	Yes ___ No ___ X:
PRTYR	Does the BIS recognize upon receipt the PRIORITY attribute?	8.12.19	M	Yes ___ No ___ X:

Annex B. IDRP Checksum Generation Algorithm

(Normative)

This annex describes the IDRP checksum algorithm, which accepts an a message of arbitrary length as its input and produces a 128-bit digital signature as its output. It is based upon the message digest algorithm described in RFC 1186.

B.1 Mathematical Notation

In this annex, the following notation is used:

Symbol Meaning

$X+Y$ Addition of two quantities, modulo 2^{32}

$X \ll s$ Left rotation (circular shifting) of the binary pattern X by " s " bit positions.

$\neg X$ Bitwise complement of the binary pattern X

$X \oplus Y$ Bitwise EXCLUSIVE-OR function of X and Y

XY Bitwise AND-function of X and Y

$X \vee Y$ Bitwise OR-function of X and Y

B.2 Algorithm Description

The input data stream, M , operated upon by this algorithm is assumed to be b binary digits in length. The first (leftmost) bit of M is labelled m_1 , the second is labelled m_2 , ..., and the last (rightmost) bit is labelled m_b .

The following steps shall be performed to compute the message digest of the message:

a) *Append padding bits*

From 1 to 512 padding bits shall be appended to the back of the original message M so that its length in bits is congruent to 448, modulo 512. If the original message length, b is already congruent to 448 modulo 512, then 512 bits of padding shall be added. The first padding bit shall be 1, and all others shall be 0.

b) *Append the length field*

When the value of b is less than or equal to 2^{64} , it shall be expressed as a 64-bit binary integer. If the quantity b is greater than 2^{64} , then only the low-order 64 bits of its binary representation shall be used. The 64-bit long binary encoded quantity shall then be appended to the back of the result obtained in the first step. Call this quantity Q .

After completing these two steps, the quantity Q will have a length which is an exact multiple of 512 bits. That is, Q consists of N 32-bit words, where N is a multiple of 16. Let $Q[1]$ represent the first (leftmost) 32-bit word of Q , ..., and $Q[N]$ represent the last (rightmost) 32-bit word of Q .

c) *Initialize the Checksum Buffer*

The checksum is accumulated in 4 32-bit buffers (A, B, C, and D). Each shall be initialized to the following values, expressed in hexadecimal notation:

Word A initial value: 01 23 45 67

Word B initial value: 89 AB CD EF

Word C initial value: FE DC BA 98

Word D initial value: 76 54 32 10

d) *Process Q in Blocks of 16 32-bit words*

Three auxiliary functions are defined that each take three 32-bit words as input and produce one 32-bit word as output;

$$f(X,Y,Z) = XY \vee (\neg X)Z$$

$$g(X,Y,Z) = XY \vee XZ \vee YZ$$

$$h(X,Y,Z) = X \oplus Y \oplus Z$$

Do the following:

For $i = 0$ to $N/16$ do /* process each 16-word block */

For $j = 1$ to 16 do: /* copy block i into X */
set $X[j]$ to $M[i*16+j]$.

end /* of loop on j */

Save A as AA, B as BB, C as CC, and D as DD.

[Round 1]:

Let $[K \ L \ M \ P \ t \ s]$ denote the operation

$$K = (K + f(L,M,P) + X[t]) \ll s.$$

Do the following 16 operations in the order indicated:

```
[A B C D 0 3]
[D A B C 1 7]
[C D A B 2 11]
[B C D A 3 19]
[A B C D 4 3]
[D A B C 5 7]
[C D A B 6 11]
[B C D A 7 19]
[A B C D 8 3]
[D A B C 9 7]
[C D A B 10 11]
[B C D A 11 19]
[A B C D 12 3]
[D A B C 13 7]
[C D A B 14 11]
[B C D A 15 19]
```

[Round 2]:

Now let $[K \ L \ M \ P \ t \ s]$ denote the operation

$$K = (K + g(L,M,P) + X[t] + 5A827999) \ll s.$$

(The value 5A827999 is a hexadecimal 32-bit constant.)

Do the following 16 operations in the order indicated:

```
[A B C D 0 3]
[D A B C 4 5]
[C D A B 8 9]
[B C D A 12 13]
[A B C D 1 3]
[D A B C 5 5]
[C D A B 9 9]
[B C D A 13 13]
[A B C D 2 3]
[D A B C 6 5]
[C D A B 10 9]
[B C D A 14 13]
[A B C D 3 3]
[D A B C 7 5]
[C D A B 11 9]
[B C D A 15 13]
```

[Round 3]:

Now let $[K \ L \ M \ P \ t \ s]$ denote the operation

$$K = (K + h(L,M,P) + X[t] + 6ED9EBA1) \ll s.$$

(The value 6ED9EBA1 is a hexadecimal 32-bit constant.)

Do the following 16 operations in the order indicated:

```

[A B C D 0 3]
[D A B C 8 9]
[C D A B 4 11]
[B C D A 12 15]
[A B C D 2 3]
[D A B C 10 9]
[C D A B 6 11]
[B C D A 14 15]
[A B C D 1 3]
[D A B C 9 9]
[C D A B 5 11]
[B C D A 13 15]
[A B C D 3 3]
[D A B C 11 9]
[C D A B 7 11]
[B C D A 15 15]

```

Then perform the following additions:

```

A = A + AA
B = B + BB
C = C + CC
D = D + DD

```

(That is, each register is incremented by the value it had when processing on this block was started.)

end /* of loop on i */

e) *Output*

After completing the last loop on *i*, the checksum is the concatenation of the final values of A, B, C, and D.

Annex C. Jitter Algorithm

(Informative)

When BISPDUs are transmitted as a result of timer expiration, there is danger that the timers of individual systems could become synchronized. To minimize the likelihood of this occurring, all periodic timers whose expiration can cause the transmission of a BISPDUs shall have jitter introduced. An example algorithm that satisfies the requirements of clause 8.16.3.3. is shown in Figure 10.

Annex D. Computing a Checksum for an Adj-RIB

(Informative)

To compute the checksum for a given Adj-RIB-Out or Adj-RIB-In, the following procedure can be used:

- Unfeasible routes will not enter into the computation.
- A sequence number will be associated with each feasible route in the information base. For an Adj-RIB-Out, it will be the locally generated sequence number of the UPDATE PDU that was used to advertise the route; for an Adj-RIB-In, it will be the sequence number of the neighbor BIS's UPDATE PDU that advertised the route.
- The feasible routes within the information base will be sorted in a non-decreasing order of their sequence numbers.
- Within each route, path attributes will be sorted in a non-decreasing order based on their type codes, followed by the Network Layer Reachability Information sorted in lexicographical order, based on the binary value of its NSAP address prefixes.
- The checksum will be generated by applying the procedures of annex Appendix B to the octet stream which is composed from the concatenation of the sorted feasible routes.

Annex E. RIB Overload

(Informative)

A BIS is said to experience a RIB-overload condition when it does not have enough memory available to store the routing information needed for its Adj-RIBs-In, Loc-RIBs, and Adj-RIBs-Out. Since the routing information that a BIS chooses to maintain is in fact a local policy, this international standard does not prescribe methods for handling overload conditions.

Methods for handling RIB overload can be considered as specific instances of local policies, and therefore are not specified by this international standard. However, some examples of approaches that may be used to control RIB overload are suggested below.

Since the Loc-RIBs contain only a subset of the routing information held in the Adj-RIBs-In, the size of a BIS's Adj-RIBs-In will be greater than or equal to the size of its Loc-RIBs. Therefore, the first step to alleviate the memory overload condition would be to reduce the amount of information that is stored in

```

CONSTANT
  Jitter=0.25 (* defined by architectural constant Jitter *)
  Resolution=100

PROCEDURE Random (max: Integer): Integer
  (* This procedure delivers a uniformly distributed random integer R,
  such that 0 < R < max *)

PROCEDURE
  DefineJitteredTimer (baseTimeValueInSeconds: Integer: expirationAction: Procedure);

VAR
  baseTimeValue, maximumTimeModifier, waitTime: Integer;
  nextexpiration: Time;

BEGIN
  baseTimeValue:=baseTimeValueInSeconds*1000/Resolution;
  maximumTimeModifier:=baseTimeValue*Jitter;
  WHILE running DO
    BEGIN
      (* First compute next expiration timer *)
      randomTimeModifier:=Random(maximumTimeModifier);
      waitTime:=baseTimeValue - randomTimeModifier;
      waitTime:=waitTime*Resolution/1000;
      nextexpiration:=CurrentTime + waitTime;
      (* Then perform expiration action *)
      expirationAction: WaitUntil(nextexpiration)
    END (* of loop *)
  END (* of DefinedJitterTimer *)

```

Figure 10. Jitter Algorithm for Timers

- 2 Adj-RIBs-In. To insure routing consistency within a
- 2 routing domain, the following steps can be applied to
- 2 an Adj-RIB-In that is associated with a peer BIS
- 2 located in an adjacent routing domain:
 - a) Remove routes that are not currently in any of the Loc-RIBs (that is, those routes that have not been selected by the Decision Process):
 - Any routes to destinations that are not contained in the routes stored in the Loc-RIB may be removed with no negative impact.
 - Any routes to destinations that are contained in the routes stored in the Loc-RIBs can also be removed. However, since they could later have been used as fallback routes (if the current route that is in the Loc-RIB becomes unfeasible), removing them may cause sub-optimal connectivity in the future.
 - b) If several Adj-RIBs-In (that have the same RIB attribute but are associated with different neighbor BISs) have routes to the same destination, then routes with higher degree of preference (as computed by the local BIS) should be retained, while routes with lower degree of preference may be deleted.
 - c) If a BIS unilaterally deletes a route, then any solicited RIB-refresh will reinstate the deleted route. Hence, if the condition persists, the memory-overloaded BIS should close the IDRP connection, and then take corrective action, such as re-opening it with an OPEN PDU that indicates support for a smaller **RIB-ATTsSet**, for example.
 - d) Terminate one or more of the IDRP sessions with other BISs. That would result in releasing the memory that was previously used to store the Adj-RIB-Ins and the Adj-RIBs-Out associated with that BIS. To ensure routing consistency within an RD this measure may be applied only to the IDRP sessions with BISs in adjacent RDs.
 - e) If all else fails to alleviate the memory overload condition, the local BIS can terminate all of its IDRP sessions.

Annex F. Processor Overload

(Informative)

A BIS is said to be CPU overloaded when there is not enough processing power to process incoming BISPDU's received from other BISs. In this situation BIS must continue to update the Adj-RIBs-In with information contained in BISPDU's received from other BISs, but may not run the Decision Process using this information except for routes received with the WITH-DRAW path attribute.

If a route received in the UPDATE_PDU has the WITH-DRAW path attribute, the local BIS checks whether this route is currently installed in one of its Loc-RIBs; if so, it removes it from the appropriate Loc-RIB, updates the appropriate Adj-RIBs-Out and FIB, and generates (if necessary) an UPDATE-PDU to inform other BIS's of the change in its Loc-RIBs and its Adj-RIBs-Out. The Decision Process on the local BIS does not select another to replace the one that becomes unfeasible.

Since this procedure decreases the size of the Loc-RIB, a long-lasting CPU overload condition can eventually deplete the entire Loc-RIB, thus making the BIS unavailable as an intermediate system. If the CPU overload condition disappears, then the Decision Process and Update Process should be run over all the new routes that were installed into the Adj-RIBs but have not yet been processed by the Decision Process. If the CPU overload condition persists for more than the predefined architectural constant **MaxCPUOverloadPeriod**, the local BIS terminates its IDRP sessions.

The order of termination of the IDRP sessions is significant. First the BIS should terminate one or more of the IDRP sessions with BISs in adjacent RDs. If after terminating IDRP sessions with all of the BISs in adjacent RDs the CPU overload still persists, the BIS terminates the rest of its IDRP sessions (with all the BISs within its own RD).

Annex G. Syntax and Semantics for Policy

(Informative)

A major task in using IDRP is to assign a degree of preference to each available path. This degree of preference will generally be a function of the number of RDs in the path, properties of the specific RDs in the path, the origin of the route, and properties of the specific border router to be used in the first hop. This annex presents an example of how a routing domain administrator might articulate this function.

In addition to controlling the selection of the best path to a given network, the network administrator must control the advertisement of this best path to neighboring RDs. Therefore, path selection and path distribution emerge as the two key aspects of policy expression in IDRP usage.

Since different aspects of one RD's policy interact, and since the policies of different RDs interact, it is important to facilitate the analysis of such interactions by means of high-quality and consistent tools. There is also a need for tools to translate the expression of the network administrator's policy to some technical mechanism within a BIS to implement that policy. These factors suggest that there should be a globally consistent way of describing policies. The syntax and semantics of these policies should be capable of expressing the path selection phase within the local RD as well as the path redistribution phase to other RDs.

Because it may be desirable to coordinate routing policy at an external level, it may prove worthwhile to create a language to describe this information in a globally consistent way. Policies expressed in such a language could conceivably be used by some high-level tools to analyze the interaction among the routing policies of different Routing Domains. The following defines one possible syntax and semantics for describing RD path policies from the point of view of the local RD. Alternative syntaxes with equivalent richness of functionality are not precluded. Other mechanisms may be needed to provide a fully functional configuration language.

A complete RD path, supplied by IDRP, provides the most important mechanism for policy enforcement. Assigning a degree of preference to a particular RD path can be modelled as a matching between this path and one or more predefined RD path patterns. Each predefined RD path pattern has a degree of preference that will be assigned to any RD path that matches it. Since patterns are naturally expressed by regular expressions, one can use regular expressions over the alphabet of RDIs to define RD path patterns and, therefore, to formulate policies.

Since certain constructs occur frequently in regular expressions, the following notational shorthand (operators) is defined:

- . matches any RDI. To improve readability, "." can be replaced by "any" so long as this does not introduce ambiguity.
- * a regular expression followed by * means zero or more repetitions
- + a regular expression followed by + means one or more repetitions
- ? a regular expression followed by ? means zero or one repetition

| alternation

() parentheses group subexpressions--an operator, such as * or +, works on a single element or on a regular expression enclosed in parentheses

{m,n} a regular expression followed by {m,n} (where m and n are both non-negative integers and $m \leq n$) means at least m and at most n repetitions.

{m} a regular expression followed by {m} (where m is a positive integer) means exactly m repetitions.

{m,} a regular expression followed by {m,} (where m is a positive integer) means m or more repetitions.

Any regular expression is generated by these rules.

The Policy Based Routing Language can then be defined as follows:

a) `<Policy-Based-Routing> ::= { <policy-statement> }`

Semantics: each policy statement might cause a given possible IDRP advertisement (possibility) to be installed into the routing table as the route to a given (set of) networks. Thus, an empty Policy-Based-Routing means that no possibilities will be accepted.

b) `<policy-statement> ::= <policy-expression>'='<locpref-expression>,<multi_exit_disc>;'`

Semantics: if a given possibility matches the policy-expression, then that possibility will be accepted with a degree of preference denoted by the integer value locpref-expression.

c) `<policy-expression> ::= <policy-term> | <policy-term> <policy-operator> <policy-term>`

d) `<policy-term> ::= <destination-list> <RD-path> <origin> <distribution-list> | '(' <policy-expression> ')' | NOT <policy-expression> | <>`

e) `<policy-operator> ::= OR | AND`

Semantics: the intersection of the Network Layer Reachability information of a possibility and the destination-list must be non-empty; the RD-path of the possibility must match the RD-path as a sequence; the origin of the possibility must be a member of the origin set; if these conditions are met, the route denoted by the possibility is accepted as a possible route to those NSAP address prefixes of the intersection of the possibility Network Layer Reachability information and the destination-list.

f) `<RD-path> ::= "regular expression over RD-path-segments"`

Semantics: the RD-path of the possibility must be generated by the regular expression `<RD-path>`.

g) `<destination-list> ::= '<' { NSAP-address-prefix destination-list } '>' | '<' ANY '>'`

Semantics: A non-empty sequence enumerates the NSAP address prefixes of the destination-list; ANY denotes the set of all NSAPs.

h) `<origin> ::= IDRP | EXTERNAL | ANY`

Semantics: origin enumerates the sources from which routing information can be learned by the protocol; ANY denotes the set of all origins.

i) `<distribution-list> ::= '<' { RDI } '>' | '<' ANY '>'`

Semantics: if a given possibility is accepted and installed into the routing table, then distribution-list is the set of (neighboring) RDIs to whose border routers we will distribute the IDRP-derived routes.

j) `<locpref-expression> ::= <locpref-term> | <locpref-term> '+' <locpref-term> | <locpref-term> '-' <locpref-term> | <locpref-term> '*' <locpref-term> | <locpref-term> '/' <locpref-term> | REJECT`

`<locpref-term> ::= <integer> | <function> | '(' <locpref-expression> ')'`

Semantics: if a possibility matches with the local preference REJECT, then that possibility will not be used. Otherwise, the integer value of the local preference indicates the degree of preference of the possibility, with higher values preferred over lower ones.

k) `<Multi_exit_disc_pref> ::= "True|False"`

Semantics: If the Multi_exit_disc_pref is set to true, and if two routes have the same path attributes except for MULTI_EXIT_DISC, and if the two policy expressions are evaluated to have the same preferences, then the MULTI_EXIT_DISC value received from the adjacent RD for each route is used to select between the two otherwise equally preferable routes. Clause 8.12.7 describes the usage of MULTI_EXIT_DISC, and Appendix J provides several illustrative examples.

White spaces can be used between symbols to improve readability. "<>" denotes the empty sequence.

There are two built-in functions, PathLength() and PathWeight(). PathLength() takes the RD path as an

argument and returns the number of RDIs in that path. PathWeight() takes the RD path and an RDI weight table as arguments and returns the sum of weights of the RDIs in the RD path as defined by the RDI weight table. In order to preserve determinism, the RDI weight table must always have a default weight which will be assigned to any RDI which is not in that table.

- 1 The RDI path, as used above, is constructed from left
- 1 to right (which is consistent with IDRP), so that the
- 1 most recent RD in the path occupies the rightmost
- 1 position. Each destination NSAP address prefix (and its associated complete RD path) received from other BIS neighbors is matched against local Routeing Policies.

If either no match occurs or the degree of preference associated with the matched policy is REJECT, then the received information is rejected. Otherwise, a degree of preference associated with the matched policy is assigned to that path. Notice that the process terminates on the first successful match. Therefore, policy-terms should be ordered from more specific to more general.

The semantics of a matched policy is as follows: If a destination in <destination-list> that was originally introduced into IDRP from <origin> is received via <RD-path>, that network should be redistributed to all RDs in <distribution-list>.

The interconnection of routeing domains shown in Figure 11 can be used as an example to illustrate how policy terms can be written. In this picture, there are four destination NSAPs, ten BIS's, and five Routeing Domains.

First, consider RD 2. It has no destination NSAPs attached, and models a transit routeing domain, such as a backbone network. It may have a very simple policy: it will carry any traffic between any two RDs, without further constraint. If RD 1 and RD 3 are neighboring domains, then its policy term could be written as:

- 1 RD 2: < ANY > < .* (1 | 3) > < IDRP > < 1 3 >
- 1 = 10

The first component in this policy, the destination NSAP list

< ANY >

says that any destination is subject to this policy. The second component, the RD path

- 1 < .* (1 | 3) >

says that routeing information that came from either RD 1 or RD 3 matches this policy, including routes

from RDs that lie beyond RD 1 and RD 3. The third component, the origin

< IDRP >

says that this route was learned via mechanisms defined in the IDRP protocol. This means that routes learned by other mechanisms would not match this policy. The fourth component, the distribution list

< 1 3 >

says that this route may be redistributed to both RD 1 and RD 3. Finally, the degree of preference assigned to any route which matches this policy is set to 10.

To improve readability, the above policy can be rewritten as:

RD 2: <ANY><(1|3) ANY*><IDRP><1 3> = 10

Next, consider RD 3. It is willing to provide transit service to RD 4 and RD 5, presumably due to multilateral agreements. RD 3 should set its policy as follows:

- RD 3: <ANY><(4|5)><IDRP><2 4 5> = 10
- 1 RD 3: <ANY><.* 2 ><IDRP><4 5> = 10
- RD 3: <ANY><3><ANY><2 4 5> = 10

This would allow RD 3 to distribute internal routes received from RDs 4 and 5 to RDs 2, 4, and 5, and all backbone routes through RD 2 would be distributed to RDs 4 and 5. RD 3 would advertise its own networks to RDs 2, 4, and 5. Destination NSAPs in RD 4 and RD 5 would be able to reach each other, as well as destination NSAPs in RDs 1 and 3 and anything beyond them. RD 3 allows any origin in routes from RD 2. This implies that RD 3 trusts RD 2 to impose policy on routes imported by means other than IDRP. Note that although the policy statement would appear to allow RD 3 to send RDs 4 and 5 their own routes, the IDRP protocol would detect this as a routeing loop and prevent it.

Now consider RD 1. RD 1 wishes to use the backbone service provided by RD 2, and is willing to carry transit traffic for RD 4. The policy statements for RD 1 might read:

- RD 1: <ANY><4><IDRP><2>=150
- 1 RD 1: <ANY><.* 2><ANY><4> = 150
- RD 1: <ANY><1><ANY><2 4> = 150

RD 1 will redistribute all routes learned from the RD 2 backbone to RD 4, and vice versa, and distribute routes to its own networks to both RD 2 and RD 4. The degree of preference assigned to any route which matches this policy is set to 150.

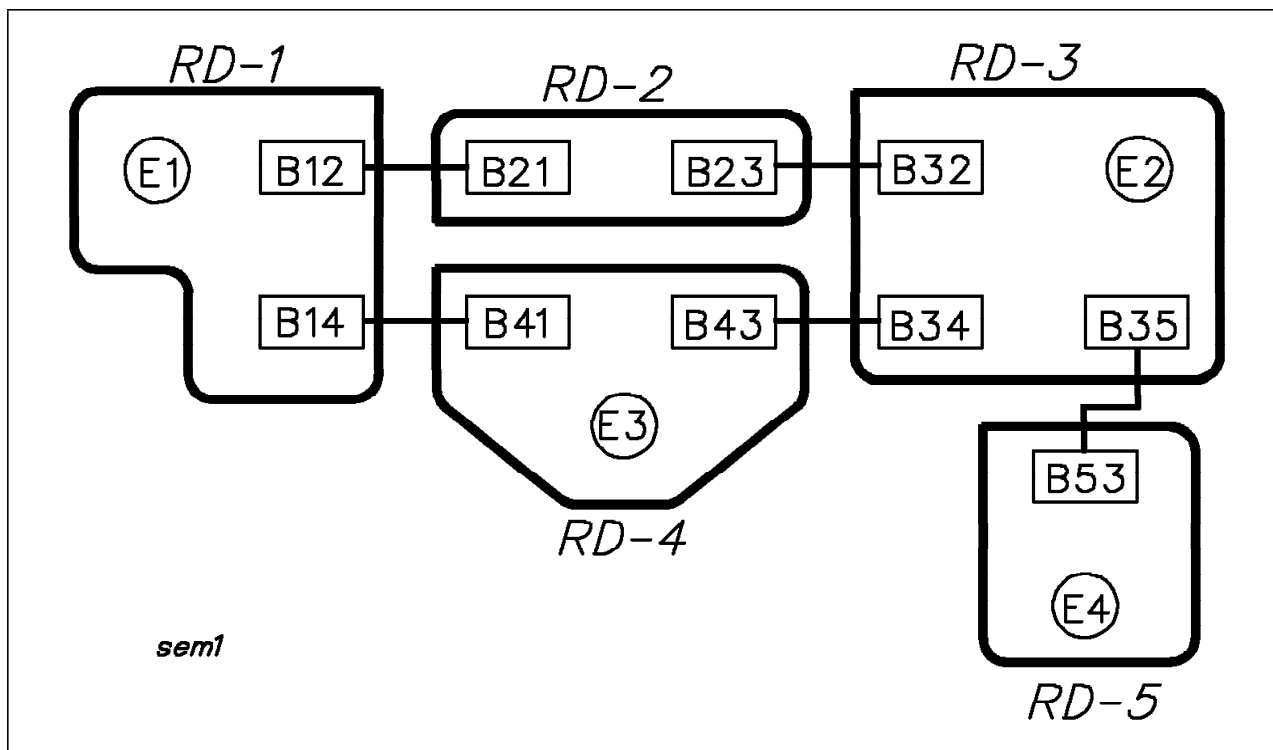


Figure 11. Example Interconnection of Routing Domains

RD 5 is a more interesting case. RD 5 wishes to use the backbone service, but is not directly connected to RD 2. Its policy statements could be as follows:

```
1 RD 5: <ANY><4 3><IDRP><> = 10
1 RD 5: <ANY><.* 3 2><.><> = 10
RD 5: <ANY><5><.><3> = 10
```

This policy imports routes through RD 2 and RD 3 into RD 5, and allows RD 5 and RD 4 to communicate through RD 3. Since RD 5 does not redistribute any routes other than its own, it is a stub ERD. Note that RD 5 does not trust RD 3 to advertise only routes through RD 2, and thus applies its own filter to ensure that it only uses the backbone. This lack of trust makes it necessary to add the second policy term.

RD 4 is a good example of a multihomed ERD. RD 4 wishes to use RD 3 as its primary path to the backbone, with RD 1 as a backup. Furthermore, RD 4 does not wish to provide any transit service between RDs 1 and 3. Its policy statement could read:

```
1 RD 4: <ANY><.* 3><ANY><> = 10
1 RD 4: <ANY><.* 1><ANY><> = 20
RD 4: <ANY><4><ANY><1 3> = 10
```

Paths to any destination through RD 3 are preferred, but RD 1 will be used as a backup if necessary. Note that since RD 4 trusts RD 3 to provide it with reasonable routes, it is not necessary to explicitly import routes from RD 5. Since the redistribution terms are

null except for networks within RD 4, RD 4 will never carry any transit traffic.

Given the topology and policies described above, it becomes apparent that two paths of equal preference would be available from RD 2 to any of the destinations in RD 4. Since ties are not allowed, an arbitrary tie-breaking mechanism would come into play (as described above), which might result in less than optimal routes to some destinations. An alternative mechanism that would provide optimal routes while still allowing fallback paths would be to provide destination-by-destination policies in specific cases, and explicit tie-breaking policies for the remaining destinations. For example, the policies for RD 2 could be rewritten as follows:

```
1 RD 2: <47:0005:000042><.* 1><IDRP><3> =
1 10
1 RD 2: <47:0005:000042> <.* 3><IDRP><1> =
1 20
1 RD 2: <.><.* 1><IDRP><3> = 20
1 RD 2: <.><.* 3><IDRP><1> = 10
```

Paths to destination NSAP prefix 47:0005:000042 through RD 1 would be preferred, with RD 3 as a fallback; paths to all other destinations through RD 3 would be preferred over those through RD 1. Such optimizations may become arbitrarily complex.

There may be other, simpler ways of assigning the degree of preference to an RD path. The simplest way to assign a degree of preference to a particular

path is to use the number of RDIs in the RD path as the degree of preference. This approach reflects the heuristic that shorter paths are usually better than longer ones. This policy can be implemented by using the PathLength() built-in function in the following policy statement:

```
<ANY><.*><ANY><ANY> =
    PathLength(RDpath)
```

This policy assigns to any network with an arbitrary RD path a degree of preference equal to the number of RDIs in the RD path; it then redistributes this information to all other BIS speakers. As an example, an RD path which traverses three different Routeing Domains will be assigned the degree of preference 3.

Another approach is to assign a certain degree of preference to each individual RD, and then determine the degree of preference of a particular RD path as the sum of the degree of preferences of the RDIs in that path. Note that this approach does not require the assignment of a specific degree of preference to every RDI in the global OSIE. For RDIs with an unknown degree of preference, a default can be used. This policy can be implemented by using the PathWeight() built-in function in the following policy statement:

```
<ANY><.*><ANY><ANY>
    = PathWeight(RDpath, RDWeightTable)
```

As an example, if Routeing Domains 145 and 55 have 10 and 15 as their weights in the RDWeightTable, and if the default degree of preference in the RDWeightTable is 50, then an RD path that traverses Routeing Domains 145, 164, and 55 will be assigned degree of preference 75.

The above examples demonstrate some of the simple policies that can be implemented with IDRP. In general, very sophisticated policies based on partial or complete RD path discrimination can be written and enforced. It should be emphasized that movement toward more sophisticated policies will require parallel effort in creating more sophisticated tools for policy interaction analysis.

Annex H. Importing Internal Reachability Information

(Informative)

EDITOR'S NOTE:

During the meeting at Sydney, several member bodies expressed an interest in seeing more detail

on how the inter-domain protocol (IDRP) and the intra-domain protocol (DIS 10589) can pass information cooperatively between each other.

As a result of the Sydney meetings, revised text will be circulated for both IDRP and for DIS 10589. However, since the editor has not yet had a chance to review the both texts in detail, he has chosen to delete the text from SC6 N6120 which addressed cooperation between these protocols, with the intention of developing replacement text after both revised documents are available for review.

The editor asks member bodies to consider the question of information exchange between these two protocols, and to submit suggestions as part of their comments on this document. Members should also consider whether this type of material can best be developed as an annex to IDRP, as an annex to DIS 10589, as a standard in its own right, as a Technical Report, etc.

Annex I. Formation of RDCs

(Informative)

Confederations exist in the knowledge configured into a given BIS. Since this knowledge must be added one BIS at a time, it is necessary to examine how a confederation can grow, and what happens in the interim when only some of the BISs are aware of the information regarding the confederation.

There are some potential problems that one should be aware of:

- Routes through a confederation might not work properly if BISs in the middle of the confederation do not know about the confederation.
- Routes may not work properly while a planned very large confederation with confederations nested inside is growing, but is not yet large enough to include all the confederations that eventually will be nested inside.
- Policies in distant BIS's must change when confederations are formed or dissolve.

If confederations are formed and dissolved carefully, then these problems can be avoided. The next sections describe the steps that should be taken for several common scenarios.

I.1 Forming a New Lower Level Confederation

Let's start with the simplest case—a newly formed confederation consisting of several RDs. The steps involved are:

- a) First warn all managers of all BISs whose RDs are contained in the new RDC that a new confederation will be formed consisting of the RDs in a particular set. If the new confederation is to be nested within an existing confederation, the existence of the new confederation will not be noticeable to any BISs outside the nesting confederation. Thus the affected BISs are those within the lowest level confederation in which this new confederation will be nested. If there are multiple overlapping confederations in which the confederation will be nested, with no smaller confederation nested within one of the overlapping confederations and in which the new confederation will be nested, BISs in all those confederations will be affected.
- b) A manager of a BIS that has policies regarding any of the RDs to be included in the new confederation must modify those policies since the RDs in the new RDC can no longer be differentiated. For example, if the previous policy was that some of those were all right to route through and others not, a new single policy for the new confederation would need to be formulated.
- c) The policy regarding the new confederation must be added to the existing set of policies (the confederation will not appear immediately).
- d) When ample time has elapsed so that managers will can modify the policies at their BISs, the managers of the BISs in the new confederation can start informing them about the new confederation.
- e) One by one, each BIS is informed that it is in the confederation. The order in which the BISs are modified is critical. At all times the set of BISs that have been informed about the confederation must be a connected set. Thus the confederation must be built gradually outwards until all the BISs have been modified.
- f) When all the BISs in the confederation have been modified, the managers of remote BISs can be informed that the confederation has been fully formed, and any old policies regarding the RDs in the confederation can now be safely deleted.

Note that the above rules apply as well if the new confederation is one that is nested within another confederation. The only difference that occurs when the new confederation to be formed is nested within a confederation X is that managers of BISs that are not contained within X do not need to be informed about the formation of X.

Also note that the BISs internal to X still need to retain their policies regarding the RDs and confederations within X.

I.2 Forming a Higher Level Confederation

Now assume it is desired to form a new confederation X that will have some number of already formed confederations nested within it, say Y and Z. The steps are:

- a) As above, warn all managers of all affected BISs (i.e. in the lowest level confederation(s) that X will be nested within (or all BISs, if X is a top level confederation)) that a new confederation X will be formed, and list all the RDs and confederations to be included (the ones that are currently visible externally, i.e., don't list the RDs in confederations to be included in X -- just list the top level confederations to be included)..
- b) As above, managers of BISs must figure out a reasonable policy for the new confederation.
- c) As above, the policy regarding the new confederation must be ADDED to the existing set of policies (the confederation will not appear immediately).
- d) As above, when ample time has elapsed so that managers will have been given an appropriate opportunity to modify the policies at their BISs, the managers of the BISs in the new confederation can start informing the BISs in the confederation about the confederation.
- e) As above, one by one, each BIS is informed that it is in the confederation, where the order in which the BISs are configured with the confederation information is critical—at all times the confederation must be connected.

The difference, though is in how the BISs are configured. Initially, the BISs are informed, one by one, that they are in X, but they are NOT informed that Y and Z are nested within X. Instead, they will be configured as though X is a lowest level confederation.

- f) After all BISs in the confederation have been configured to know they belong to X, they can one by one be modified to believe Y and Z are nested within X. In contrast to the knowledge that they belong to X, which must be configured in a careful order, the knowledge that Y and Z are nested within X can be configured within the BISs in X in any order.
- g) When all the BISs in the confederation have been twice modified (once to know about X, and once to know about the nesting rules), managers of remote BISs can be informed that the confederation has been fully formed, and the policies regarding RDs and confederations in the new confederation can now be safely deleted.

I.3 Deleting a Lowest Level Confederation

Now suppose there is a confederation X, with no confederations nested within it, that is being dissolved. The steps involved are:

- a) First warn all managers of all affected BISs (see point 1 in the previous 2 sections for a rigorous description of which BISs are affected), that X will be dissolved, and list all the RDs in X.
- b) A manager of a BIS that has policies regarding X needs to add the same policy many times, one for each RD in X. It is also possible at this time to make policies that are different for the RDs in X.
- c) When ample time has elapsed so that managers will have been given an appropriate opportunity to modify the policies at their BISs, the managers of the BISs in X can start informing the BISs in X to forget about X.
- d) One by one, each BIS in X is informed that it is not in X. The order in which the BISs are modified is critical. At all times the set of BISs that believe they are in X must be a connected set. Thus X must be shrunk gradually towards one point.
- e) When all the BISs in X have been modified, the managers of remote BISs (those in the confederation within which X had been nested, or all BISs if X was a top level confederation) can be informed that X no longer exists, and the policies regarding X can now be safely deleted.

I.4 Deleting a Higher Level Confederation

The steps involved are:

- a) As above, warn all managers of all affected BISs (see point 1 in the previous 3 sections) that confederation X will be dissolved, and list all the RDs and confederations included in X (i.e., the ones that will become visible when X is deleted.)
- b) As above, policies need to be ADDED regarding all the RDs and confederations that were included in X.
- c) As above, when ample time has elapsed so that managers will have been given an appropriate opportunity to modify the policies at their BISs, the managers of the BISs in the new confederation can start informing the BISs in X about X's impending dissolution.
- d) Now different from above, one by one (in any order) the BISs in X are informed that nothing is nested within X any more, though they retain knowledge of X.
- e) After all BISs in X have been configured to believe X is a bottom level confederation, knowledge of X can be carefully deleted from the BISs

(careful because the order is critical, as above, i.e. X must at all times be connected.)

- f) After all BISs previously in X have been twice modified (once to delete the nesting rules for X, and one to delete X), managers of remote BISs can be informed that X has been fully dissolved, and policies regarding the confederation can now be safely deleted.

Annex J. Example Usage of MULTI_EXIT_DISC Attribute

(Informative)

The MULTI-EXIT DISC attribute can be used to provide a limited form of multi-path (load-splitting), as is shown in the following examples.

— Example 1 (see Figure 12):

Consider the case when a BIS A located in routeing domain RD-A has two adjacent BISs (B1 and B2) that belong to the routeing domain RD-B. Assume that RD-B has Network Layer Reachability information about NSAPs N1, N2, ... Nk, and it wants to advertise this information to RD-A. By using the MULTI_EXIT_DISC attribute RD-B may do selective load splitting (based on NSAP addresses) between B1 and B2.

For example, BIS B1 advertises to BIS A Network Layer Reachability information N1, N2, ... Nm with the MULTI_EXIT_DISC set to X, and advertises N(m+1), ... Nk with the MULTI_EXIT_DISC set to X + 1.

Similarly, BIS B2 advertises to BIS A Network Layer Reachability information N1, N2, ... Nm with the MULTI_EXIT_DISC set to X + 1, and advertises N(m+1), ... Nk with the MULTI_EXIT_DISC set to X.

As a result, traffic from BIS A that destined to N1, N2, ... Nm will flow through BIS B1, while traffic from BIS A that destined to N(m+1), ... Nk will flow through BIS B2. This scenario illustrates the simplest way of doing limited multipath with IDRP.

— Example 2 (see Figure 13):

Next consider more complex case where there is a multihomed routeing domain RD-A that has only slow speed links. RD-A is connected at several points to a transit routeing domain RD-B that has only high speed links; BIS A1 is adjacent to BIS B1, and BIS A2 is adjacent to BIS B2. RD-A wants to minimize the distance that incoming NPDUs addressed to certain ESs—say ES(1) through ES(k)—will have to travel within RD-A.

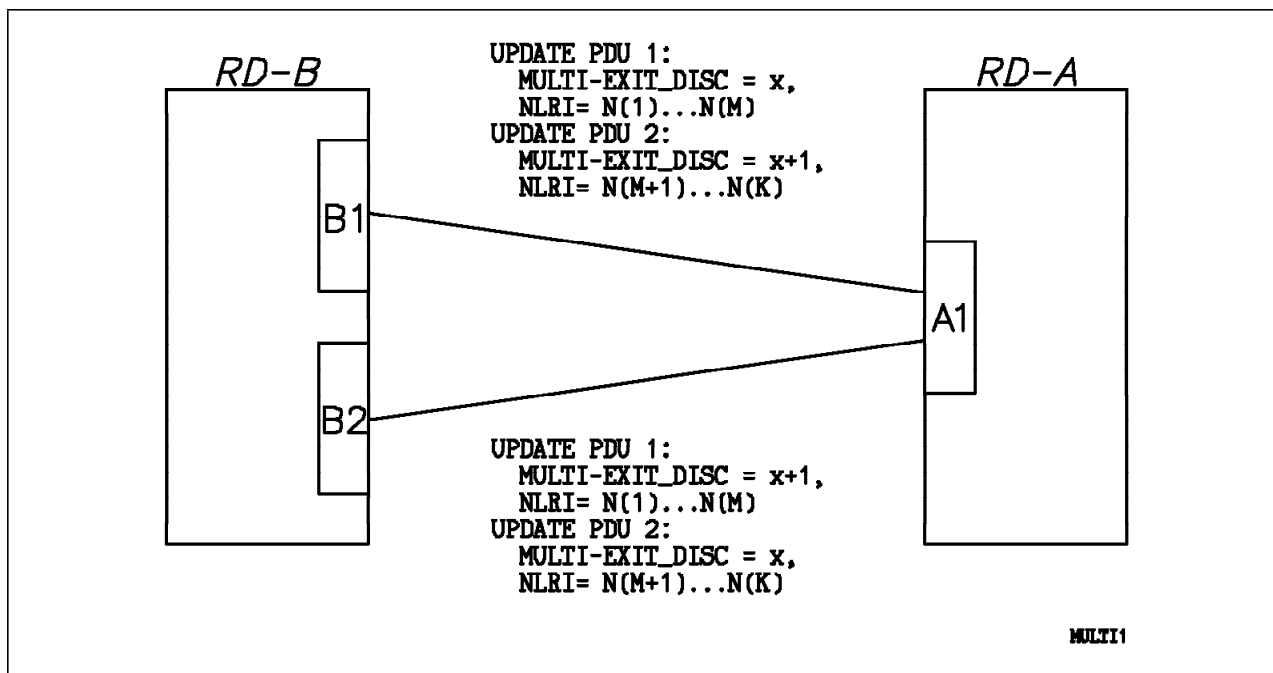


Figure 12. Example 1 Configuration

One way of doing this is by making BIS A1 to announce to BIS B1 destinations ES(1) – ES(k) with a lower MULTI_EXIT_DISC, as compared to the MULTI_EXIT_DISC that BIS A2 will use when announcing the same destinations to the BIS B2. Similarly, BIS A2 would announce to BIS B2 destinations ES(k+1) – ES(n) within the RD-A that are closer to the BIS A2 (than to the BIS A1) with the lower MULTI_EXIT_DISC, as compared to the

MULTI_EXIT_DISC that the BIS A1 will use when announcing the same destinations to the BIS B1.

When traffic that destined to some ES within RD-A enters RD-B on its way to RD-A via BIS X, X picks up the exit BIS that has the lowest MULTI_EXIT_DISC value for that destination. For example, X may pick up BIS A2 as an exit, even if the distance between A2 and X is greater than the distance between A1 and X.

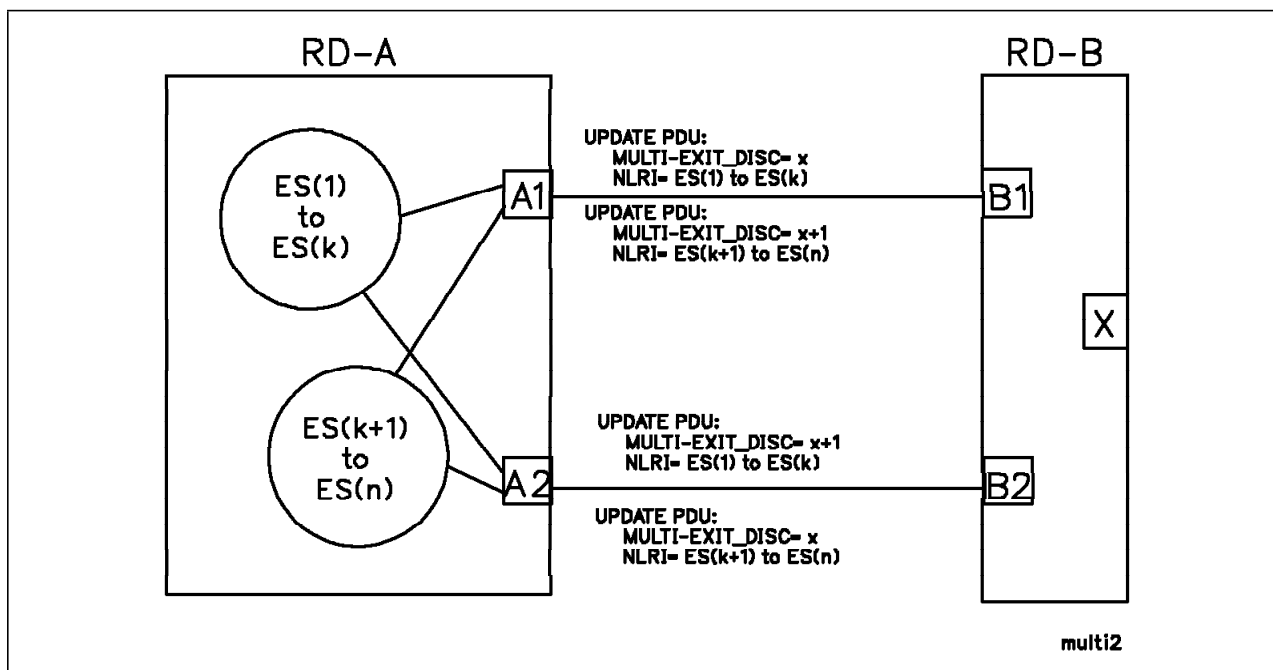


Figure 13. Example 2 Configuration

Annex K. Example Routeing Algorithm

(Informative)

This annex contains a description of an example routing algorithm that satisfies the requirements of the protocol described in this international standard:

```

/*
 * ROUTE_SELECT procedure takes the following as the arguments:
 * 1. route R with destination R.dest and N path attributes
 *   R.attrib[1], R.attrib[2], ... , R.attrib[N]
 * 2. K sets of Adj-RIBs-In (one per adjacent BIS)
 * 3. K sets of Adj-RIBs-Out (one per adjacent BIS)
 * 4. M sets of Loc-RIB's: Loc-RIB[1], Loc-RIB[2], ... , Loc-RIB[M].
 * 5. M sets of PIB's: PIB[1], PIB[2], ... , PIB[M].
 * 6. M sets of FIB's: FIB[1], FIB[2], ... , FIB[M].
 *
 * and updates the appropriate Adj_RIB-In, Adj_RIB_Out, Loc_RIB, R,
 * and propagates R to the adjacent BIS's.
 *
 * It calls function pa2riba() that performs mapping of path
 * attributes to RIB attribute.
 *
 * It calls function rt2dop() that takes route and PIB as an
 * input and return modified route with its degree of preference
 * as the LOCAL_PREF attribute.
 *
 * It calls function find_current_route() that takes Loc_RIB
 * and DESTINATION as arguments and return route to that
 * DESTINATION from the Loc_RIB (if present).
 *
 * It calls function route_feasible() that takes route
 * as an argument and returns TRUE if this route does
 * not have the WITHDRAW path attribute, and FALSE otherwise.
 *
 * It calls function unfeasible2feasible() that takes
 * route with the WITHDRAW path attribute and just
 * strips this attribute out
 *
 * It calls function find_new_route() that takes as arguments
 * Adj_RIBS, RIB attribute, destination, and returns route r
 * with the highest degree of preference found in one of Adj_RIBS
 * such that the RIB attribute supplied as an argument is
 * equal to the pa2riba(r).
 *
 * It calls function best_ext() that takes as arguments route, and
 * RIB attribute, and returns route r to a particular
 * destination that has the highest degree of preference among
 * all other routes to the same destination different from the
 * one supplied as an argument received external inter-domain
 * links.
 *
 * We assume that each route in Adj_RIB's or Loc_RIB's has field
 * "dop" which is the degree of preference associated with this route.
 */

procedure ROUTE_SELECT(r, adj_ribs_in, adj_ribs_out, loc_ribs, pibs)
  INT i, j;
  RIB_ATTRIBUTE riba;
  ROUTE cr;      /* current route */

```

```
ROUTE ext_r; /* best route received from BIS in an adjacent routeing domain */
```

```
switch (route_feasible(r)) {
  case TRUE: /* feasible route */

    store r in the appropriate adj_ribs;

    for (i = 1; i <= N; i++) {
      riba = pa2riba(r.attrib[i]);
      if (riba == NULL)
        continue;
      for (j = 1; j <= M; j++) {
        if (loc_ribs[j].attrib == riba) {
          cr = find_current_route(loc_ribs[j], r.dest);
          if (cr != NULL) { /* there are other routes */
            r = rt2dop(r, pibs[j]);
            if (cr.dop >= r.dop) { /* no changes to the LOC_RIB */
              if r was received from a BIS in an adjacent routeing domain {
                ext_r = best_ext(r, riba);
                if (ext_r != NULL && r.dop > ext_r.dop) {
                  mark r as "propagate to internal only";
                  mark r as "best external";
                  unmark cr as "best external";
                }
              }
            }
          }
          else { /* install better route */
            update loc_ribs[j] with r;
            update fibs[j] with r;
            mark r as "propagate";
            if r was received from a BIS in an adjacent routeing domain {
              mark r as "best external";
              ext_r = best_ext(r, riba);
              if (ext_r != NULL) {
                unmark ext_r as "best external";
              }
            }
          }
        }
      }
    }
  }
  else { /* no other routes */
    update loc_ribs[j] with r;
    update fibs[j] with r;
    mark r as "propagate";
    mark r as "best external";
  }
}
}
```

```
update appropriate adj_ribs_out
propagate all routes marked as "propagate";
```

```
propagate all routes marked as "propagate to internal only" to
all other BISs in the same domain;
```

```
break;
```

```
case FALSE: /* unfeasible route */
```

```
r = unfeasible2feasible(r);
```

```
remove r from the appropriate adj_ribs_in;
```



```

for (i = 1; i <= N; i++) {
    riba = pa2riba(r.attrib[i]);
    if (riba == NULL)
        continue;
    for (j = 1; j <= M; j++) {
        if (loc_ribs[j].attrib == riba) {
            cr = find_current_route(loc_ribs[j], r.dest);
            if (cr != r) { /* no changes to the Loc_RIB */
                if r was received from a BIS in an adjacent routeing domain {
                    if r marked as "best external" {
                        update appropriate adj_ribs_out
                        propagate r to all other BISs in the domain as unreachable;
                        ext_r = best_ext(r, riba);
                        if (ext_r != NULL) {
                            mark ext_r as "best external";
                            update appropriate adj_ribs_out
                            propagate ext_r to all other BISs in the domain;
                        }
                    }
                }
            }
        }
    }
}
else {
    remove r from loc_ribs[j];
    update fibs[j];
    update appropriate adj_ribs_out
    propagate r to the adjacent BIS's as unreachable;
    cr = find_new_route(adj_ribs, loc_ribs[j].attrib, r.dest);
    if (cr != NULL) {
        update loc_ribs[j] with cr;
        update fibs[j] with cr;
        update appropriate adj_ribs_out
        propagate cr to the adjacent BIS's;
        if r was received from a BIS in an adjacent routeing domain {
            ext_r = best_ext(r, riba);
            if (ext_r != NULL) {
                mark ext_r as "best external";
                if (ext_r != cr) {
                    update appropriate adj_ribs_out
                    propagate ext_r to all other BISs in the domain;
                }
            }
        }
    }
}
}
}
}
}
}
/* end switch */

```

end procedure

Index

A

administrative domain 1
 aggregating routes
 See route aggregation
 authentication 11, 13, 53

B

BISPDU 10—20
 boundary IS
 definition 3

C

CAPACITY 19, 42, 52, 56
 CEASE PDU 20, 26, 28, 29, 53
 checksum field
 in BISPDU header 11, 23, 53
 confederations
 See RDC (Routeing Domain Confederation)
 configuration information 22
 connection management between BISs 25—29

D

deployment guidelines
 for ESs and ISs 22
 for RDs 21
 DESTINATION SPECIFIC QOS 17, 40, 52, 56
 DESTINATION SPECIFIC SECURITY 18, 42, 52, 56
 distinguishing attributes
 derived from NPDU 55
 matching with RIB-Att 56
 distinguishing path attributes in UPDATE PDU 8
 as identifier of RIBs and FIBs 31
 type specific 35
 type-value specific 35
 DIST_LIST_EXCL 16, 38, 50, 51
 DIST_LIST_INCL 15, 37, 38, 50, 51

E

error handling 53
 EXPENSE 16, 40, 51, 56
 EXT_INFO 14, 35, 51

F

forwarding of NPDUs
 and NPDU-derived Distinguishing Attributes 55
 to external destinations 57
 to internal destinations 55

H

HIERARCHICAL RECORDING 18
 HIERARCHICAL_RECORDING 41
 hold time
 of OPEN PDU 12

I

IDRP ERROR PDU 19
 inter-domain link
 definition 3
 real links and inter-domain traffic 3
 virtual links and inter-domain traffic 3
 virtual links and intra-domain traffic 3
 intra-domain routeing protocol vi, 3, 10, 22
 ISO/IEC 10589 7

K

KEEPALIVE PDU 20

L

LOCAL_PREF 16, 39, 51

M

MULTI-EXIT_DISC 16, 39, 51
 multi-homed end routeing domain 3

N

NET (Network entity title)
 syntax and encoding 9
 network layer reachability information
 as encoded in UPDATE PDU 19
 as related to configuration information 22
 information reduction of 50
 NEXT_HOP 15, 36, 51
 NLRI
 See network layer reachability information
 Non-transitive attribute 14
 NSAP Addresses
 syntax and encoding 9

O

OPEN PDU 11

P

partial bit 14
 path attributes
 paths, RD
 as encoded in UPDATE PDU 15
 policies
 detecting inconsistencies 7, 45
 external inconsistencies 45
 internal inconsistencies 45
 indirect expression in UPDATE PDUs 7
 local setting of 7
 syntax and semantics of 89—93
 positioning of IDRP
 with respect to ISO 8473 vi
 within the Network layer vi, 5
 PRIORITY 19, 42, 52

R

RD Path

See paths, RD

RDC (Routeing Domain Confederation) 3, 7, 42

associated managed object 22

boundary of 43

configuration information for 43

definition 3

disjointed. 4

entry marker in RD_PATH attribute 43

formation and deletion of 93, 96

generating RD_PATH attributes 43

in OPEN PDU 13

nested 3

overlapping 3

permissible policies of 43

recursive nature of 7

updating a received RD_PATH attribute 44

use of RDI to identify 7, 9

RDI (routeing domain identifier)

conformance to ISO 8348/Add. 2 9

syntax and encoding 9

RD_PATH 15, 36, 51

RD_SEQUENCE

See paths, RD

RD_SET

See paths, RD

real link

See inter-domain link

reliable delivery of BISPDU 23

RESIDUAL ERROR 16, 39, 51, 56

route aggregation

of NLRI 51

of path attributes 50

routeing domains

adjacent (definition) 3

as part of global OSIE 5

end routeing domain 3, 7

transit routeing domain 3, 7

routes

as expressed in UPDATE PDU 13

definition of 7

destinations 7

path attributes 7

S

Size of BISPDU 11

maximum, of OPEN PDU 11, 12

of the entire BISPDU 11

SOURCE SPECIFIC QOS 17, 40, 51, 56

SOURCE SPECIFIC SECURITY 18, 41, 52, 56

stub end routeing domain 3

T

TRANSIT DELAY 16, 39, 51, 56

Transitive attribute 14

TR 9575 (Routeing Framework) vi, 5

U

UPDATE PDU 13

V

virtual link

See inter-domain link