International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.2771

(07/2014)

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

Next Generation Networks – Security

Framework for deep packet inspection

Recommendation ITU-T Y.2771

ITU-T

International Telecommunication Union

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	Y.1300–Y.1399
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Enhancements to NGN	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Packet-based Networks	Y.2600–Y.2699
Security	**Y.2700–Y.2799**
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
FUTURE NETWORKS	Y.3000–Y.3499
CLOUD COMPUTING	Y.3500–Y.3999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T Y.2771

Framework for deep packet inspection

Summary

Recommendation ITU-T Y.2771 provides a framework for deep packet inspection (DPI). The primary purpose of this framework is to describe a structured approach for designing, defining and implementing DPI solutions in support of service/application awareness for facilitating interoperability in the evolving networks. It serves to identify and assist in understanding the network issues, from primarily an architectural viewpoint. This Recommendation also provides DPI framework aspects from modelling and performance.

The purpose of such frameworks is especially to outline possible relationships between a DPI function and other network functions, to assist in identifying requirements for DPI functions (which would be the subject of other ITU-T Recommendations like, e.g., Recommendation ITU-T Y.2770) and to help for terminology work (e.g., when a definition would be related to a functional model).

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.2771	2014-07-18	13	11.1002/1000/12178

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T Y.2771

Framework for deep packet inspection

1 Scope

This Recommendation provides a framework for deep packet inspection (DPI) in packet-based networks. The primary purpose of this Recommendation is to describe the fundamental concepts, functional components and capabilities of DPI that can be used to identify information flows in packet-based networks by DPI entities, to support the specification of DPI requirements and to guide structured solutions for packet-based networks (such as NGNs).

This Recommendation provides high-level system information with regard to some fundamental concepts as typically relevant when realizing DPI entities. However, it is not expected to present all-inclusive detailed specifications for the DPI. Rather, this Recommendation provides an high-level information (i.e., a framework) and is intended to be used as background material for use by ITU Study Groups and other expert groups outside the ITU, e.g., as input for their development of detailed standards for DPI functionalities.

The scope of this Recommendation includes:

a) basic architectural principles that will be encountered in combining DPI in various network architectures;

b) protocol architectural aspects from the perspective of DPI;

c) example functional models and their application to DPI use case scenarios; and

d) performance frameworks in order to assist in DPI performance discussions like the identification of key performance indicators related to DPI.

Implementers and users of this ITU-T Recommendation shall comply with all applicable national and regional laws, regulations and policies. The mechanism described in this Recommendation may not be applicable to international correspondence in order to ensure the secrecy and sovereign national legal requirements placed upon telecommunications providers, as well as the ITU Constitution and Convention.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T E.800] Recommendation ITU-T E.800 (2008), *Definitions of terms related to quality of service*.

[ITU-T G.602] Recommendation ITU-T G.602 (1988), *Reliability and availability of analogue cable transmission systems and associated equipments*.

[ITU-T H.248.86] Recommendation ITU-T H.248.86 (2014), *Gateway control protocol: H.248 Support for deep packet inspection*.

[ITU-T X.200] Recommendation ITU-T X.200 (1994), *Information technology – Open Systems Interconnection – Basic Reference Model: The basic model*.

[ITU-T X.731] Recommendation ITU-T X.731 (1992), *Information technology – Open Systems Interconnection – Systems management: State management function*.

[ITU-T Y.2704] Recommendation ITU-T Y.2704 (2010), *Security mechanisms and procedures for NGN*.

[ITU-T Y.2770] Recommendation ITU-T Y.2770 (2012), *Requirements for deep packet inspection in next generation networks*.

[ETSI TS 132 410] ETSI TS 132 410 (2012), *Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Telecommunication management; Key Performance Indicators (KPI) for UMTS and GSM (3GPP TS 32.410 version 11.0.0 Release 11*.

[IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol – DARPA Internet Program Protocol Specification*.

[IETF RFC 2460] IETF RFC 2460 (1998), *Internet Protocol, Version 6 (IPv6) Specification*.

[IETF RFC 5101] IETF RFC 5101 (2008), *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 application [ITU-T Y.2770]: A designation of one of the following:

- An *application protocol type* (e.g., IP application protocols ITU-T H.264 video, or session initiation protocol (SIP));
- A *served user instance* (e.g., VoIP, VoLTE, VoIMS, VoNGN, and VoP2P) of an application type, e.g., "voice-over-Packet application";
- A "*provider specific application*" for voice-over-Packet, (e.g., 3GPP provider VoIP, Skype VoIP); and
- an application embedded in another application (e.g., application content in a body element of a SIP or an HTTP message).

An application is identifiable by a particular identifier (e.g., via a bit field, pattern, signature, or regular expression as "application level conditions", see also clause 3.2.2 of [ITU-T Y.2770]), as a common characteristic of all the above listed levels of applications.

3.1.2 availability [ITU-T E.800]: Availability of an item to be in a state to perform a required function at a given instant of time or at any instant of time within a given time interval, assuming that the external resources, if required, are provided.

3.1.3 application-descriptor (also known as application-level conditions) [ITU-T Y.2770]: A set of rule conditions that identify the application (according to clause 3.2.1 of [ITU-T Y.2770]).

This Recommendation addresses the application descriptor as an object in general, which is synonymous with application-level conditions. It does not deal with its detailed structure, e.g., syntax, encoding and data type.

3.1.4 deep packet inspection (DPI) [ITU-T Y.2770]: Analysis, according to the layered protocol architecture OSI-BRM [ITU-T X.200], of:

- payload and/or packet properties (see list of potential properties in clause 3.2.11 of [ITU-T Y.2770]) deeper than protocol layer 2, 3 or 4 (L2/L3/L4) header information, and
- other packet properties

in order to identify the application unambiguously.

NOTE – The output of the DPI function, along with some extra information such as the flow information is typically used in subsequent functions such as reporting or actions on the packet.

3.1.5 DPI engine [ITU-T Y.2770]: A subcomponent and central part of the DPI functional entity which performs all packet path processing functions (e.g., packet identification and other packet processing functions in Figure 6-1 of [ITU-T Y.2770]).

3.1.6 DPI policy condition (also known as DPI signature) [ITU-T Y.2770]: A representation of the necessary state and/or prerequisites that identify an application and define whether a policy rule's actions should be performed. The set of DPI policy conditions associated with a policy rule specifies when the policy rule is applicable (see also [b-IETF RFC 3198]).

A DPI policy condition must contain application level conditions and may contain other options such as state conditions and/or flow level conditions:

1) State condition (optional):
   a) network grade of service conditions (e.g., experienced congestion in packet paths); or
   b) network element status (e.g., local overload condition of the DPI-FE).
2) Flow descriptor/Flow level conditions (optional):
   a) packet content (header fields);
   b) characteristics of a packet (e.g., number# of MPLS labels);
   c) packet treatment (e.g., output interface of the DPI-FE).
3) Application descriptor/application level conditions:
   a) Packet content (application header fields and application payload).

NOTE – The condition relates to the "simple condition" in the formal descriptions of flow level conditions and application level conditions.

3.1.7 DPI policy decision functional entity (DPI-PDFE) [ITU-T Y.2770]: The function remote to the DPI-FE that decides the signature-based rules to be enforced in the DPI-FE. Some control and/or management functions may not necessarily be remote from the DPI-FE.

3.1.8 flow descriptor (also known as flow level conditions) [ITU-T Y.2770]: A set of rule conditions that is used to identify a specific type of flow (according to clause 3.1.3 of [ITU-T Y.2770]) from inspected traffic.

NOTE 1 – This definition of flow descriptor extends the definition in [b-ITU-T Y.2121] with additional elements as described in clause 3 of [ITU-T Y.2770].

NOTE 2 – For further normative discussion of the flow descriptor as used in [ITU-T Y.2770], see Annex A of [ITU-T Y.2770].

3.1.9 reliability [ITU-T E.800]: The probability that an item can perform a required function under stated conditions for a given time interval.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 DPI analyser: A subsequent entity in the DPI processing path (within a DPI policy enforcement function) with focus on comparison functions between the particular packet headers and payloads of preselected packet flows. The primary scope of the DPI analyser is related to the evaluation of DPI policy *conditions* against *preselected* incoming packets.

NOTE – The DPI analyser may be located after a DPI scanner (see clause 3.2.6). The DPI analyser may provide the functionality of an intrusion detection system (IDS) analyser.

3.2.2 DPI node: A network element or device that realizes the DPI related functions. It is thus a generic term used to designate the realization of a DPI physical entity.

NOTE – Functional perspective: the DPI node function (DPI-NF) comprises the DPI policy enforcement function (DPI-PEF) and the (optional) local policy decision function (L-PDF); hence, the DPI-NF is functionally equal to the DPI functional entity.

3.2.3 DPI policy action (action in short): Definition of what is to be done to enforce a policy rule, when the conditions of the rule are met. Policy actions may result in the execution of one or more operations to affect and/or configure network traffic and network resources; see also [b-IETF RFC 3198].

3.2.4 DPI policy enforcement function (DPI-PEF): A logical entity that enforces policy decisions, given by DPI policy rules.

3.2.5 DPI scanner (also used as "DPI scan function"): The first entity in the DPI processing path (within a DPI policy enforcement function) which provides a pre-selection (related to the subsequent DPI analyser, see clause 3.2.1) by checking *all* DPI policy *conditions* against *all* incoming packets.

3.2.6 DPI "1+N" redundancy group: The collection of DPI functional components (e.g., DPI node, DPI-PIB, DPI engine, etc.) following a "1+N" redundancy architecture (and $N \geq 1$), given by a single working component and N protection components.

NOTE – The above collection is used to provide extra reliability and improve availability for a DPI node or network deployed with one or DPI nodes.

4 Abbreviations and acronyms

4.1 General abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

A_{DPI}	DPI policy action
BRM	Basic Reference Model
CAM	Content Addressable Memory
C_{DPI}	DPI policy condition
DAI	Deep Application Identification
DHI	Deep Header Inspection
DiffServ	Differential Service
DPI	Deep Packet Inspection
DPI-AcEF	DPI Action Execution Function
DPI-AnF	DPI Analyser Function
DPI-FE	DPI Functional Entity
DPI_{InP}	In-Path DPI
DPI-NF	DPI Node Function
DPI_{OoP}	Out-of-Path DPI
DPI-PDFE	DPI Policy Decision Functional Entity
DPI-PE	DPI Physical Entity
DPI-PEF	DPI Policy Enforcement Function

DPI-PIB	DPI Policy Information Base
DPI-PIF	DPI Packet Identification Function
DPI-ScF	DPI Scan Function
DNNF	Determining Next Node Function
FIB	Forwarding Information Base
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HW	Hardware
IDS	Intrusion Detection System
IP	Internet Protocol
IPFIX	IP Flow Information Export
KPI	Key Performance Indicator
KPI_{DPI}	Key performance indicators for DPI entities
L-PDF	Local PDF
L2VPN	Layer 2 Virtual Private Network
L_XHI	Header Inspection of protocol Layer X
L_XPI	Payload Inspection of protocol Layer X
LX	(Protocol) Layer X
LX+	Higher (Protocol) Layer than LX
MIB	Management Information Base
MPI	Medium depth Packet Inspection
MPLS	Multi-Protocol Label Switch
MTBF	Mean Time Between Failures
MTTR	Mean Time To Repair
NA(P)T	Network Address (and Port) Translation
NGN	Next Generation Network
OSI-BRM	Open System Interconnection-Basic Reference Model
PDF	Policy Decision Function
PEP	Policy Enforcement Point
PFF	Packet Forwarding Function
PIB	Policy Information Base
QoS	Quality of Service
RACF	Resource and Admission Control Functions
R_{DPI}	DPI policy rule
R-PDF	Remote PDF (i.e., PDF remotely located from DPI node perspective)
RTSP	Real Time Streaming Protocol
SDU	Service Data Unit

SIP Session Initiation Protocol

SPI Shallow Packet Inspection

S_D-PDF Session-dependent PDF

S_I-PDF Session-independent PDF

SW Software

TCAM Ternary Content Addressable Memory

TCP Transmission Control Protocol

TOS Type of Service

VoIP Voice over IP

VoLTE Voice over Long Term Evolution

VoIMS Voice over Integrated Media System

VoNGN Voice over Next Generation Network

VoP2P Voice over Peer to Peer

4.2 Mathematical symbols

This Recommendation uses the following symbols (with name, unit and brief description):

ε_{DPI}	(DPI) error rate	–
$\varepsilon_{f\text{-}n}$	(DPI) false-negative error rate	–
$\varepsilon_{f\text{-}p}$	(DPI) false-positive error rate	–
$\phi_{P,In}$	(DPI) processing rate of incoming packets	[s^{-1}]
$\phi_{P,Out}$	(DPI) packet rate in outgoing direction	[s^{-1}]
$\phi_{P,Node,Out}$	Packet node throughput	–
$\phi_{P,Identified}$	Rate of successfully identified packets	–
$P_{Hit,BloomFilter}$	Estimated information certainty of probability	–
Ndb	The number of DPI policy rules	–
Sp	The packet size	–
N_{DPIeng}	The number of DPI engines	–
τ_{TD}	Node-internal transfer delay (of DPI node)	[ns]
$\underline{\underline{A}}$	Set of (DPI policy) rule actions	–
$\underline{\underline{C}}$	Set of (DPI policy) rule conditions	–
$\underline{\underline{R}}$	Set of (DPI policy) rules	–

5 Conventions

None.

6 Architectural framework

6.1 Network architecture framework – High-level network scenarios

This framework outlines the major boundary conditions of a DPI deployment in a network infrastructure. Some principle DPI framework scenarios may be identified by consideration of criteria like:

- **Network location level** (i.e., the location of a DPI entity in a packet network domain)
  - at the edge ("**border DPI**"); or
  - within the network ("**core DPI**");
  - between peering networks ("**peering DPI**").
- **Network packet (path) types** (i.e., the kind of packet types are inspected)[1]
  - user plane (or transport stratum; e.g., IP data-path, IP media-path, IP bearer-path, tunnel, MPLS LSR, pseudo wire, etc.); or
  - control plane (or service stratum; e.g., IP signalling-path); or
  - management plane; or
  - combinations.
- **Alignment level with other network architectures** (i.e., the level how a DPI entity is coupled with the underlying packet network architecture)
  - **isolated DPI** entity (i.e., the DPI entity is hidden from the packet network),

    Examples:
    - a very few DPI entities located on selected points in the network (without the goal of "full coverage"; something like a probe-style in-path DPI);
    - Out-of-path DPI entities.
  - **overlay DPI** network (i.e., there is a dedicated DPI network infrastructure, overlayed to the underlying packet network; both network infrastructures are separated from an operational perspective),

    Examples:
    - generic example: a network of in-path DPI entities which share, e.g., the user plane paths, but use separate control and/or management interfaces;
    - specific example: e.g., a DPI function with the purpose of intrusion detection.
  - **embedded DPI** entity (i.e., the DPI functional entity is embedded in a physical network element together with other functional entities with regard to non-DPI packet processing; such a physical entity should provide, e.g., a single OAM interface from a cost-effective operational perspective, which implies consequently a harmonized management model across all functional entities),

    Examples:
    - generic example: a DPI entity with a management information base which is aligned with the management base of the other, non-DPI functional entities of the same physical network element;
    - specific example: a DPI functional entity within RACF and a common management capability, but without sharing any RACF control interfaces.
  - **integral DPI** (i.e., a DPI entity "fully integrated" in "the packet network"),

[1] The notion of "packet type" could be made specific by referring to a dedicated "protocol" or a "protocol stack". Though, that level of detail is not required here.

Examples:

- generic example: an SDO-defined network reference model (architecture) which takes into account DPI entities;
- specific example: ITU-T RACF extended by DPI entities which may use existing reference points (e.g., "Rw-controlled DPI" or an ITU-T H.248-based Rw extended by support of [ITU-T H.248.86]) or may introduce new reference points).

There are consequently numerous use cases from the perspective of the network integration scenario of a DPI entity.

6.2 Protocol architectural framework – Packet inspection level for some example network applications

6.2.1 Principle

There are various packet inspection levels. Table 6-1 provides an overview of typical network applications with respect to required involvement of "packet inspection levels". The packet inspection level may be denoted either:

1) according to a basic reference model (BRM) for layered protocol architectures, here columns L_xHI and L_yPI; or
2) using "old" colloquial terms (which are described in clause 8.1 of [b-ITU-T Y-Sup.23]), here columns shallow packet inspection (SPI), medium packet inspection (MPI), deep header inspection (DHI) and deep application inspection (DAI).

See also clause 8 in [b-ITU-T Y-Sup.23] on the aspect of DPI in layered protocol architectures.

6.2.2 Distinction of cases DPI and non-DPI

The concept of DPI, from the perspective of layered protocol architectures, is fairly wide and includes even all protocol layers above layer 1 (see clause 3.2.5 of [ITU-T Y.2770]). However, the scope of packet inspection could be basically limited in case of specific network application, such as primarily related on link, network and/or transport layers only.

Such a limitation is/was typically motivated by service-, historical- or/and implementation-related aspects, for instance, economical trade-off decisions with respect to an achievable DPI service versus state-of-the-art technique. Such a kind of limited packet inspection is/was also known as shallow packet inspection, and medium depth packet inspection (SPI, MPI; see also clause 8.1 in [b-ITU-T Y-Sup.23]).

The coarse granular distinction between DPI and non-DPI according to [ITU-T Y.2770] is sufficient and also followed by this Recommendation. The notion of 'DPI' here means (loosely) policy inspection rules as supported by this Recommendation, and 'non DPI' relates more to legacy packet inspection at protocol layers 2, 3 and/or 4 (i.e., SPI, MPI).

6.2.3 Examples

Table 6-1 provides a list of example network applications versus packet inspection levels, as typically part of such network applications. It should be noted that the indications by Table 6-1 are only exemplary and not necessarily exhaustive.

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
Security:						
1.1	Network intrusion detection	–	X	X	X	There are different ID methods, a) anomaly detection b) misuse detection (here)
1.2	Security protection of network resources (network intrusion prevention, security attack prevention)	–	X	X	X	
1.3	Other security-related functions					

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
Identification:						
2.1	Subscriber, user	–	**X**	–	–	Identified by …? (e.g., network address)
2.2	Application type	–	–	**X**	**X**	Identified by …? (e.g., application layer protocol type)
2.3	Session	–	**X**	–	–	Identified by …? (e.g., IP connection, IP transport connection). See also clause 7 in [b-ITU-T Y-Sup.23]
2.4	Application control protocol (e.g., SIP, RTSP, HTTP, FTP…)	–	**X** (dependent on well-known port)	**X**	**X**	
Application data characteristics:						
2.5	Content	–	–	**X**	**X**	For example, illegal content

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
2.6	Media type (Application data type)	–	–	**X**	**X**	
2.7	Media format	–	–	**X**	**X**	
Modification (of protocol data units):						
3.1	Modification 'content': Removing of viruses	–	–	–	**X**	
3.2	Modification 'header': QoS marking	–	**X**	**X**	–	
3.3	Modification 'header & content': "ALG function"	–	**X**	**X**	–	Local NA(P)T function on L3 (& L4) and application layer

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
Usage parameter monitoring:						
4.1	Service level agreements	–	**X**	**X**	**X**	
4.2	Traffic parameter control Examples:	–	**X**	**X**	**X**	Dependent on traffic parameter type
	L3 byte rate policing (peak rate, sustainable rate)	–	**X**	–	–	
	L3 PDU size (min, max) policing	–	**X**	–	–	
	L3 burst size policing	–	**X**	–	–	
	L7 SDU size policing ("application payload")	–	**X**	**X**	–	
	L7 byte rate policing	–	**X**	**X**	–	

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
	("application volume")					
Quality of Service support:						
5.1	Traffic shaping	–	**X**	–	–	
	L3 byte rate shaping	–	**X**	–	–	See e.g., [b-ITU-T Y.1221] or [b-ITU-T H.248.53]
Network analysis:						
6.1	User behaviour	–	**X**	**X**	**X**	
6.2	Usage patterns	–	**X**	**X**	**X**	

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
Performance measurements ("key performance indicators" (KPIs)):						
7.1	Collection of remote measurements	–	**X**	**X**	**X**	
7.2	Generation of local measurements	–	**X**	**X**	**X**	
Charging/billing support:						
8.1	Time-based information	–	**X**	–	–	
8.2	Traffic volume-based information	–	**X**	–	**X**	Traffic volume related to IP byte rate (L3) or/and application data
8.3	Event-based information	–	**X**	**X**	**X**	Dependent on event type (e.g., event may be associated to content)

Table 6-1 – Packet inspection level for some example network applications

Network application (example)		Packet inspection level				Comments
		"Deep packet inspection" (DPI)				
		(Note)	"Deep header inspection" (DHI)		"Deep application identification" (DAI)	
			"Medium depth packet"			
			"Shallow packet inspection" (SPI)	"Inspection" (MPI)		
		L2 header inspection (L_2HI)	L3,4 header inspection ($L_{3,4}$HI)	L4+ header inspection (L_{4+}HI)	L7 payload inspection (L_7PI)	
Link-oriented DPI:						
9.1	DPI applications with possible layer 2-related policy conditions	**X**	**X**	**X**	**X**	See Note
NOTE – There is a principal distinction between link-oriented DPI and network-oriented DPI. Link-oriented DPI is limited to a L2 network domain, and network-oriented DPI is related to DPI signatures which cover protocol information on network layer (L3) and higher.						

7 Modelling framework

7.1 Functional models

Multiple functional models are presented, illustrating the spectrum of a packet forwarding path without any DPI (clause 7.1.2), unidirectional DPI (clause 7.1.3) till a bidirectional DPI model (clause 7.1.4).

All functional models in this clause are example functional models.

7.1.1 In-path DPI versus out-of-path DPI

There are two principle deployment scenarios of a DPI node function (DPI-NF), from the perspective of the end-to-end packet path:

- In-path DPI (DPI_{InP}): the DPI-NF is located in the end-to-end packet path, the DPI policy enforcement function (DPI-PEF) executes DPI policy rules directly on the packet traffic (also known as Online DPI); or
- Out-of-path DPI (DPI_{OoP}): the DPI-NF is *not* located in the end-to-end packet path, rather it is centralized in a packet network; the DPI-PEF thus executes DPI policy rules indirectly, e.g., on sampled packet traffic (also known as bypass DPI, offline DPI).

Both DPI modes are different from the physical node perspective, housing the DPI-NF: the DPI_{InP} may be located in a packet node, and the DPI_{OoP} would be in a node *without* any packet forwarding function (PFF; see next clause).

7.1.2 Generic packet forwarding

A packet node in a packet-based network may be abstracted (on a high-level) by a packet forwarding function (PFF) according to Figure 7-1. The PFF may be e.g., a switching function in case of MPLS label switching routers (LSRs) or Ethernet switches or bridges[2], or a forwarding/routing function in case of IPv4 [IETF RFC 791] and IPv6 [IETF RFC 2460] routers. The PFF must determine the next node (e.g., the next hop in IP networks) for each ingress packet sent in egress direction for unicast communications.

NOTE 1 – Multicasting would lead to the determination of multiple next nodes.

The information used by the determining next node function (DNNF) for that function is stored in a collocated database called forwarding information base (FIB), for instance, the IP forwarding table MIB according to [b-IETF RFC 4292] in case of IPv4 routers as defined by [b-IETF RFC 1812].

2 NOTE – The term “packet” would then be synonymous to the (L2) "frame".

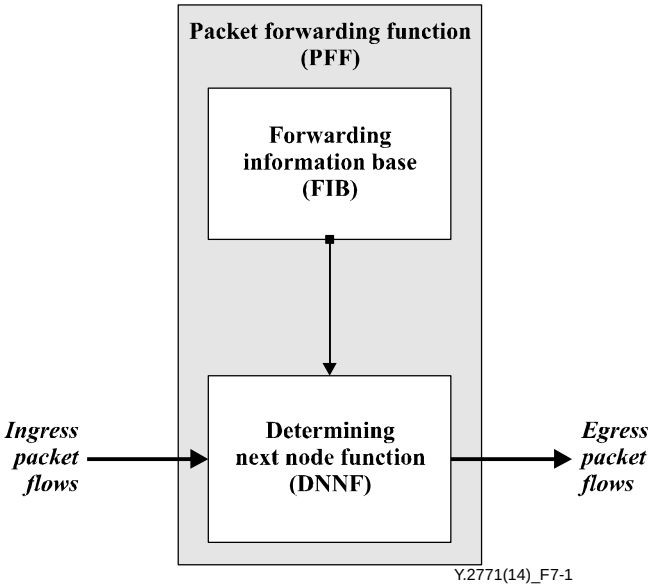


Figure 7-1 – Functional models "Generic packet forwarding"

It may be noted that the PFF is a unidirectional model. The DPI function (if present) is located in the packet path (i.e., the In-path DPI (DPI_{InP}) mode), typically before the PFF; see clause 7.1.3.

Any details and requirements for the PFF are outside the scope of this Recommendation, however the PFF is indicated in some functional models due to:

- designation of a possible DPI node behaviour without any enforced DPI policy rule (e.g., a temporary condition of an empty DPI-policy information base (DPI-PIB);
- showing that particular policy actions, like 'forward packet', would still include the involvement of the PFF; and
- an unambiguous specification baseline for some DPI node-related performance metrics (see clause 8).

It should be noted that the PFF may be void if there were just a single (egress) packet path (Note 2).

NOTE 2 – Example scenario: an in-path DPI node located between two L2 or L3 packet nodes, or an in-path DPI node in front of user equipment.

7.1.3 Unidirectional DPI

7.1.3.1 Components of the unidirectional DPI policy enforcement function

7.1.3.1.1 General, high-level functional model

Figure 7-2 provides the top-level functional model, based on the example architecture of a DPI function entity (DPI-FE) according to clause 6.2 of [ITU-T Y.2770].

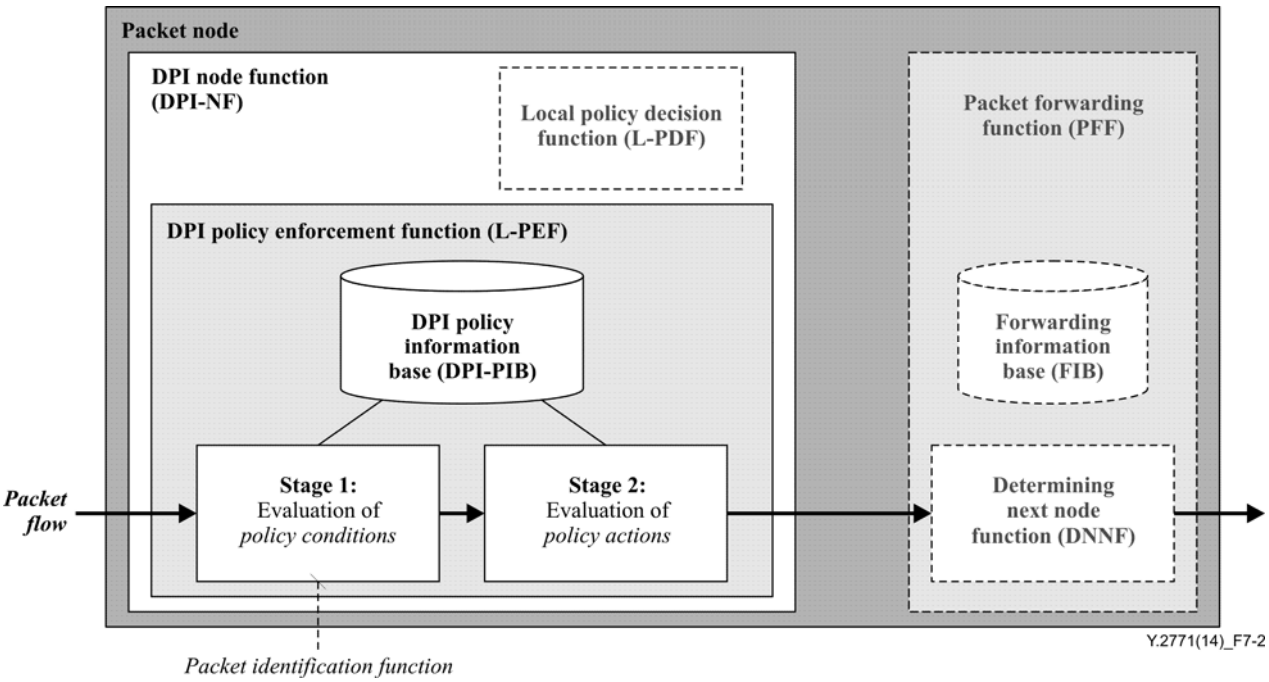


Figure 7-2 – General, high-level functional model

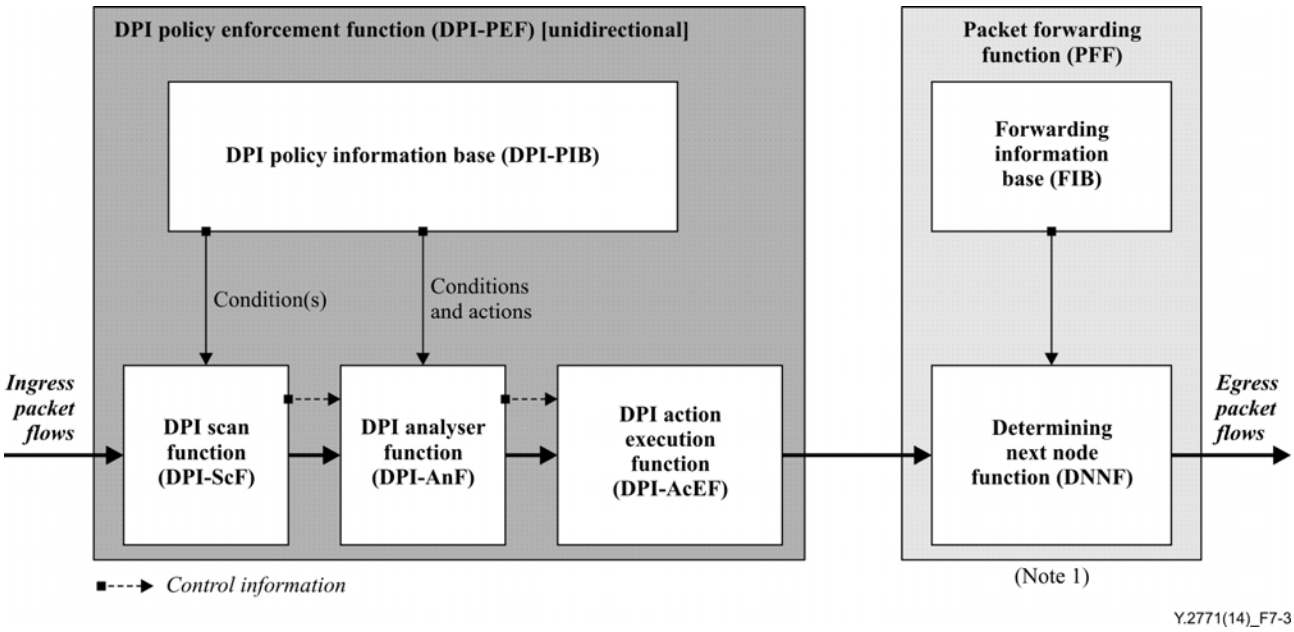
The unidirectional packet path is modelled as a staged process. The 1st stage represents the packet identification function (see also clause 7.3.2.1). That capability is a crucial function in the context of DPI, hence, further detailed (exemplified by functional decomposition models) in subsequent clauses.

7.1.3.1.2 Basic components in a serial processing topology

Figure 7-3 depicts an example of an unidirectional model (as a possible derivation from the top-level model of Figure 7-2). The DPI policy enforcement function (DPI-PEF) is located before the PFF: any incoming packet is firstly processed by the DPI-PEF and handled by the PFF. The DPI-PEF may be also structured in packet path functions plus an associated table as storage for the applied policy rules, called the DPI policy information base (DPI-PIB) or DPI signature library. In this example, the enforcement of DPI specific policy rules is the subject of:

- the DPI scan function (DPI-ScF);
- the DPI analyser function (DPI-AnF); and
- the DPI action execution function (DPI-AcEF).

These functional components are introduced and motivated in the next clause.



NOTE 1 – The PFF is out of scope of this Recommendation.
NOTE 2 – The PFF is only present for the in-path DPI mode.

Figure 7-3 – DPI models "Components of the unidirectional DPI policy enforcement function"

7.1.3.1.3 Additional components

7.1.3.1.3.1 Within the DPI-PEF

Additional components within the DPI-PEF are not yet described and left for further study.

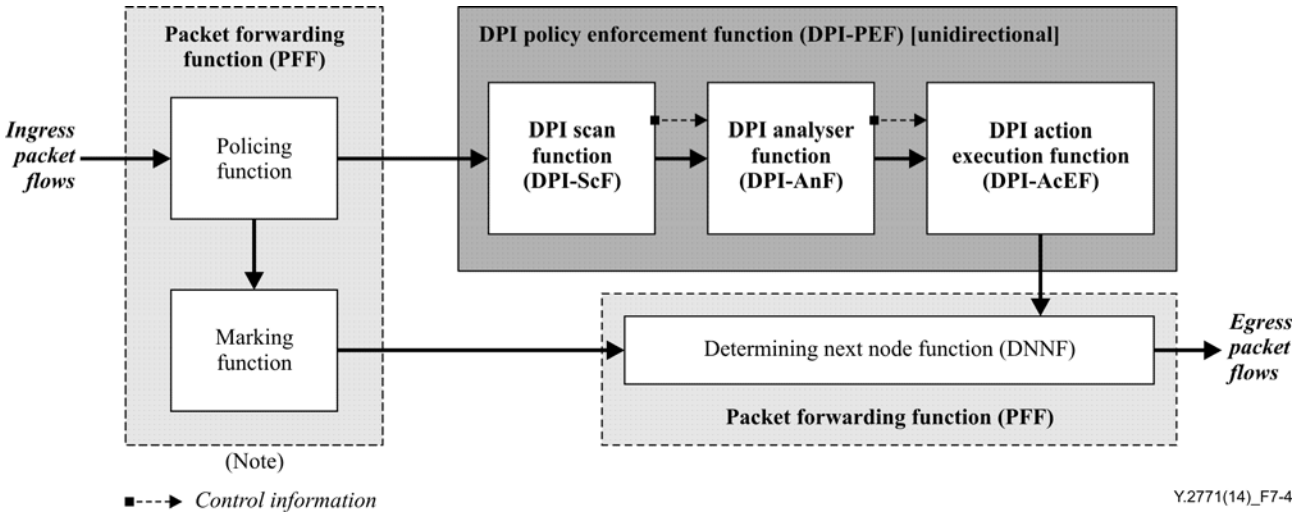
7.1.3.1.3.2 Within the PFF

The packet forwarding function may include sub-FEs such as queuing, encapsulation, shaping, policing, marking, switching and DNNF as well. These are, however, outside the scope of this Recommendation.

7.1.3.1.4 Structural aspects of packet processing path

Instead of a serial execution of packet processing functions (as illustrated in Figure 7-3) DPI node architectures with parallelism might exist also, e.g., the PFF could be also processed in parallel to the DPI-PEF.

In Figure 7-4, a packet processing path model with parallel packet processing is shown. Here the policing function monitors the packets coming into a certain ingress port, or packets with pre-defined criteria (i.e., a dedicated policy rule condition), such as an IPv4 TOS field marked with high priority (i.e., SPI-based policy rules). If those incoming packets or flows violate a bandwidth agreement, all or some of the packets may be marked accordingly and sent directly to the DNNF.



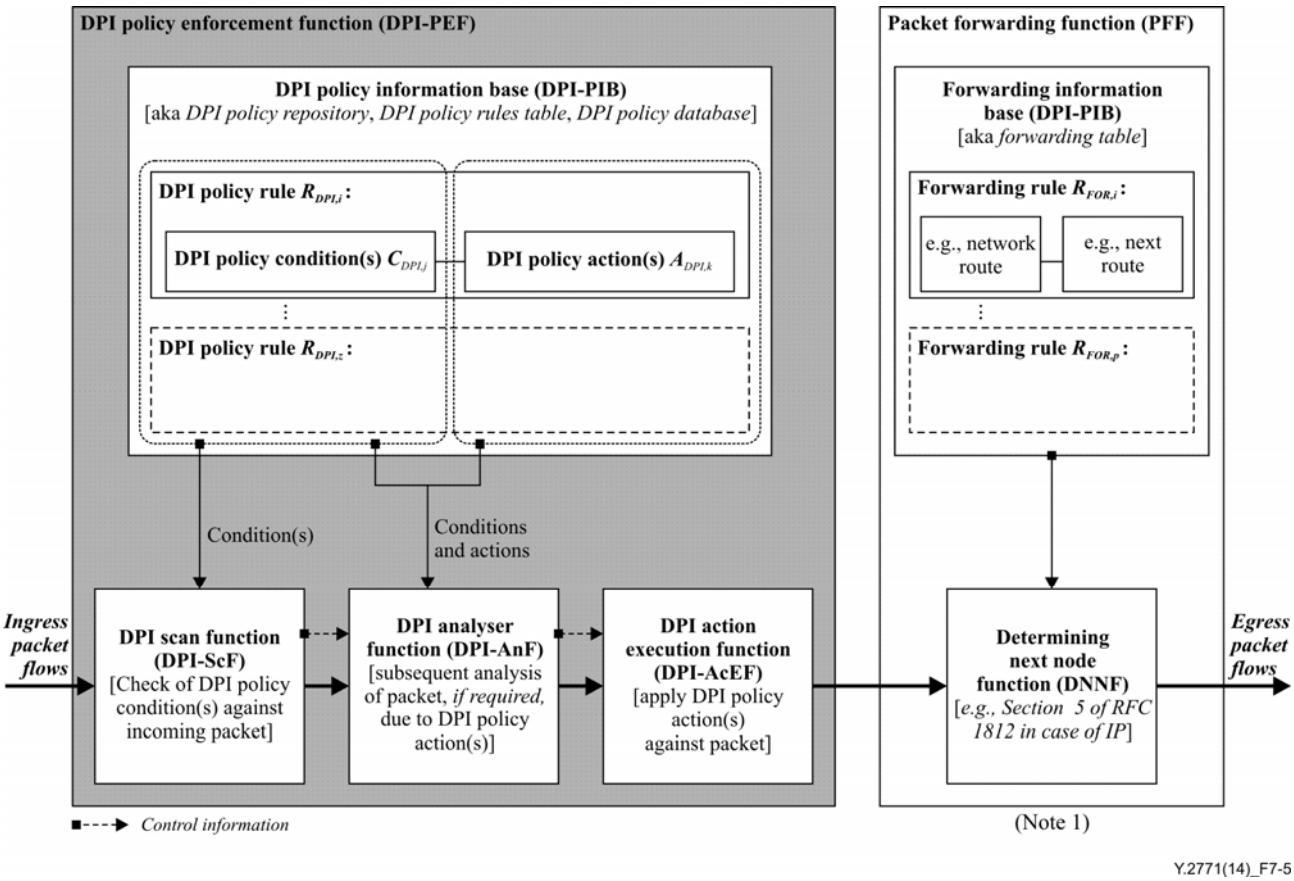
NOTE – The PFF is out of scope of this Recommendation.

Figure 7-4 – DPI model for the unidirectional DPI policy enforcement function with parallel packet processing

It should be noted that the examples in Figures 7-3 and 7-4 represent logical topologies, not physical ones. The actual implementation, however, should reflect the fact that the packet processing path a packet or a flow takes might be different even in a single DPI instance.

7.1.3.2 Structure of the DPI policy information base (DPI signature library)

More details and further structure of the DPI policy information base are provided by Figure 7-5, which is functionally adequate to Figure 7-3. The policy rule (*R*) relates to a binding of a set of actions (*A*) to a set of conditions (*C*). The conditions are evaluated to determine whether the actions are performed. The generic term 'policy rule' is also known as (specific) filter rule in case of actions related to packet filtering activities (see also clauses 7.3 and 7.6 in [b-ITU-T Y-Sup.23]).



NOTE 1 – The PFF is out of scope of this Recommendation.
NOTE 2 – The PFF is only present for the in-path DPI mode.

Figure 7-5 – DPI models "structure of the DPI policy information base"

Processing model for DPI policy rules R_{DPI}:

1) The DPI scan function (DPI-ScF) checks *all* (Note 1) DPI policy conditions C_{DPI} against an incoming packet;

   NOTE 1 – The scope of a DPI policy rule may cover the entire packet traffic forwarded by the node or be limited to a particular flow (see [ITU-T Y.2770]), given by a flow descriptor (see [ITU-T Y.2770]). The packet flow may be e.g., the subject of an end-to-end *session* (see clause 6.7 of [ITU-T Y.2770]) between application instances (e.g., in case of IP applications the end-to-end sessions could be sessions on HTTP, RTSP, SIP, FTP, etc. identified). The enforcement of session-specific policy rules is often called session-dependent policing, as opposed to session-independent policing (related to policy rules for the entire traffic aggregate of a policy enforcement node). The concept of flow and session is not further elaborated in this clause, because it is not relevant to the (high-level) functional models shown.

2) The DPI analysis function (DPI-AnF) is for further verification of policy conditions. The DPI-AnF is operated in the pipelined manner with the DPI-ScF, after the DPI-ScF initially screens every packet (Note 2). The DPI-AnF aims for performance enhancement.

NOTE 2 – For instance, the scan function may correlate an incoming packet to a specific (e.g., IP) application, and the analysis function may then provide an application-specific evaluation of the packet. The general principle behind the partitioning in DPI-ScF and DPI-AnF is related to a serial and/or hierarchical policy enforcement concept (e.g., in order to meet performance objectives). More detailed information about the DPI-AnF is for further study.

3) The DPI action execution function (DPI-AcEF) applies DPI policy actions A_{DPI} against the scanned and analysed packet.

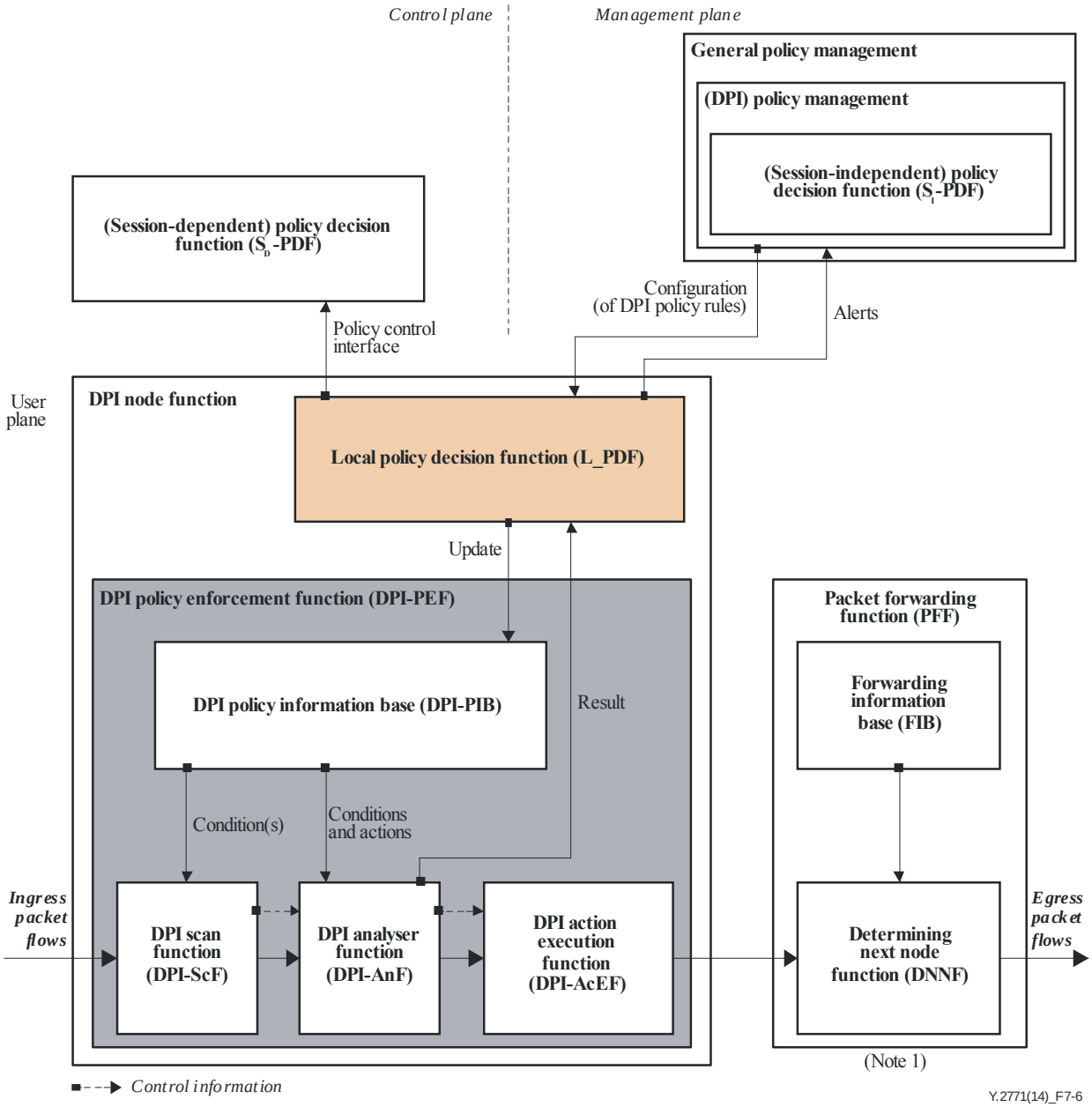
Every packet that successfully passed the DPI-PEF will be then handled by the regular PFF (see also clause 7.1.2) in case of the in-path DPI mode.

7.1.3.3 Policy decisions: modifying the DPI policy information base

The DPI-PIB provides a set of DPI policy rules $R_{DPI,i}$, which are determining the actual behaviour of the DPI-PEF. The DPI policy rules are created by a policy decision functional entity (DPI-PDFE). Figure 7-6 illustrates the example of remote PDFs, located in the control plane and management plane (Figure 7-7 depicts another example scenario, without any (direct) access from the control pane). The control plane PDF may be responsible for a session-dependent DPI policy decision (S_D-PDF). A possible session concept is mentioned in Note 1 of clause 7.1.3.2. [ITU-T H.248.86] defines such a policy control interface. The management plane PDF may be responsible for a session-independent DPI policy decision (S_I-PDF) in Figure 7-6. The policy management may principally define session-dependent and session-independent policy rules (as in the example of Figure 7-7).

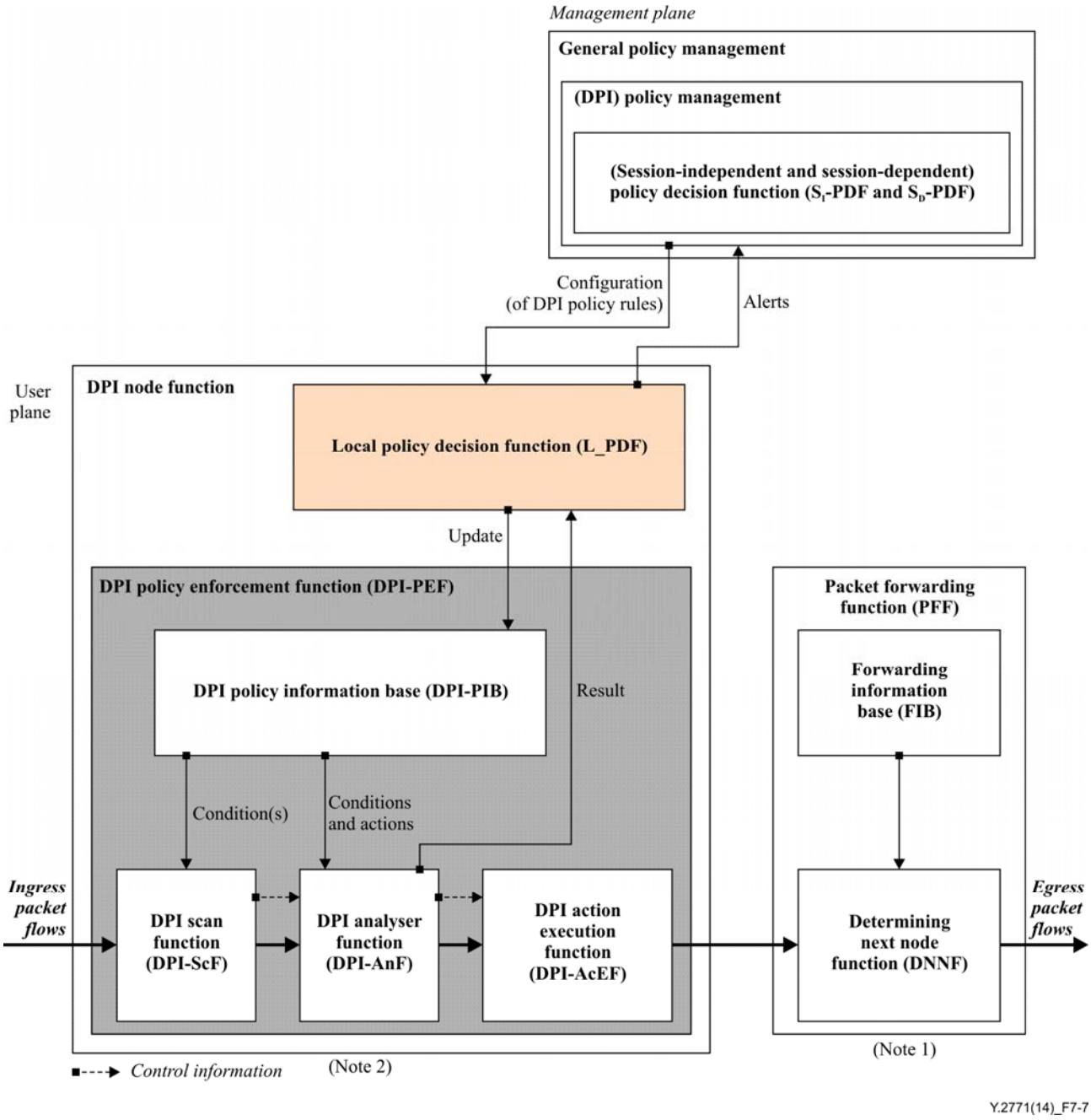
NOTE 1 – Session-dependent policing may be e.g., specific to a particular application, user, media type, etc., and session-independent policing may cover general security rules, e.g., updating on a daily basis. The (DPI) policy rules from the S_D-PDF and S_I-PDF are complementary. Figure 7-4 provides an example of just such a network configuration.

DPI policy management is typically part of a general policy management entity, responsible also for non-DPI policy rules like "legacy" shallow packet inspection (SPI) or medium depth packet inspection (MPI).



NOTE 1 – The PFF is out of scope of this Recommendation.
NOTE 2 – The PFF is only present for the in-path DPI mode.

Figure 7-6 – DPI models "modifying the DPI policy information base via the control and management plane"



NOTE 1 – The PFF is out of scope of this Recommendation.
NOTE 2 – The DPI node might not be connected to any other network element of the control plane.
NOTE 3 – The PFF is only present for the in-path DPI mode.

Figure 7-7 – DPI models "modifying the DPI policy information base via the management plane only"

The PDF(s) are typically located in geographically remote network elements, as indicated in Figure 7-6 (and also Figure 7-7) by the policy control interface for the S_D-PDF and policy management interface for the S_I-PDF. Any remote PDF may be temporarily out-of-service, motivating an additional, optional local PDF (L-PDF) in order to optimize DPI service availability in a network.

The L-PDF together with the DPI-PEF represents the DPI node function.

NOTE 2 – It relates to the local policy decision path in Figure 7-1 of clause 7.2.1 of [ITU-T Y.2770].

The L-PDF (if available) provides the external communication with remote PDF(s) and the internal interface to the DPI-PEF for updating the DPI-PIB and processing possible results from the DPI analyser function (DPI-AnF). The L-PDF may be also responsible for resolving possible rule interaction problems between the set of DPI policy rules.

NOTE 3 – The detection and resolution of rule interactions is a fundamental function for PDFs.

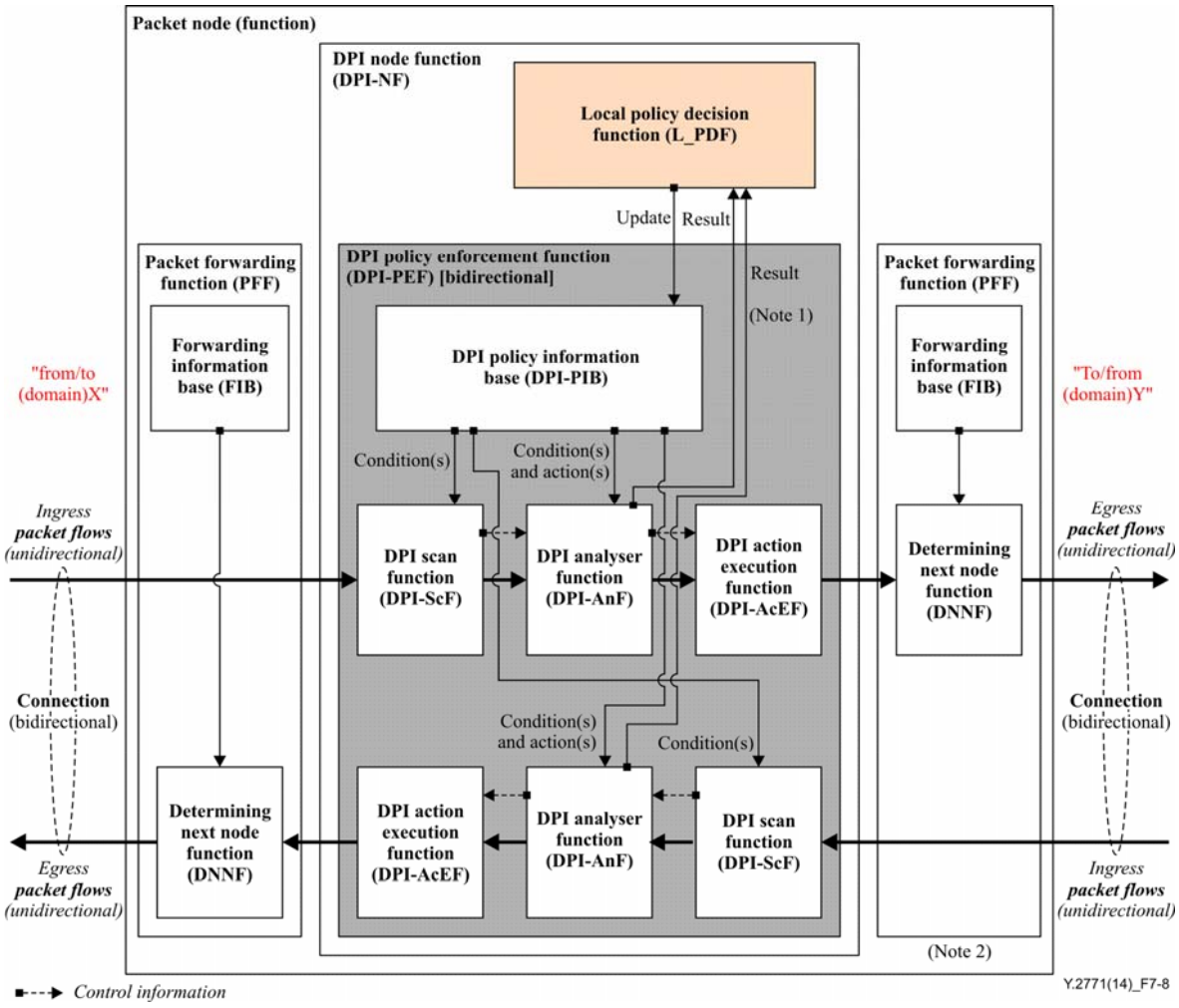
The feedback from the DPI analyser function (DPI-AnF) may lead to alerts towards policy management (e.g., the notification of a new security threat).

The fundamental policy enforcement model is unidirectional, but may be extended for bidirectional communication paths; see the following clause.

7.1.4 Bidirectional DPI

Figure 7-8 provides an example of the bidirectional DPI model (definition see clause 3.2.4 in [ITU-T Y.2770]).

A bidirectional packet connection consists of two unidirectional packet flows. The DPI-PIB is the binding element between both traffic directions from the DPI policy enforcement perspective. A "bidirectional" DPI policy rule would provide DPI policy conditions or/and actions relevant for both traffic directions.



NOTE 1 – The DPI-PIB may be internally organized in direction-dependent DPI-PIBs, e.g., DPIx ® y-PIB and DPIy ® x-PIB.
NOTE 2 – The PFF is out of scope of this Recommendation.
NOTE 3 – The PFFs are only present for the in-path DPI mode.

Figure 7-8 – DPI models "bidirectional DPI"

The L-PDF is responsible for the bidirectional DPI-PEF and provides a "mediation function" by post-processing possible results from the unidirectional analyser functions (DPI-AnF), which may then trigger the update of (local) policy rules or/and the notification of remote PDF(s).

7.1.5 Stateless and statefull DPI

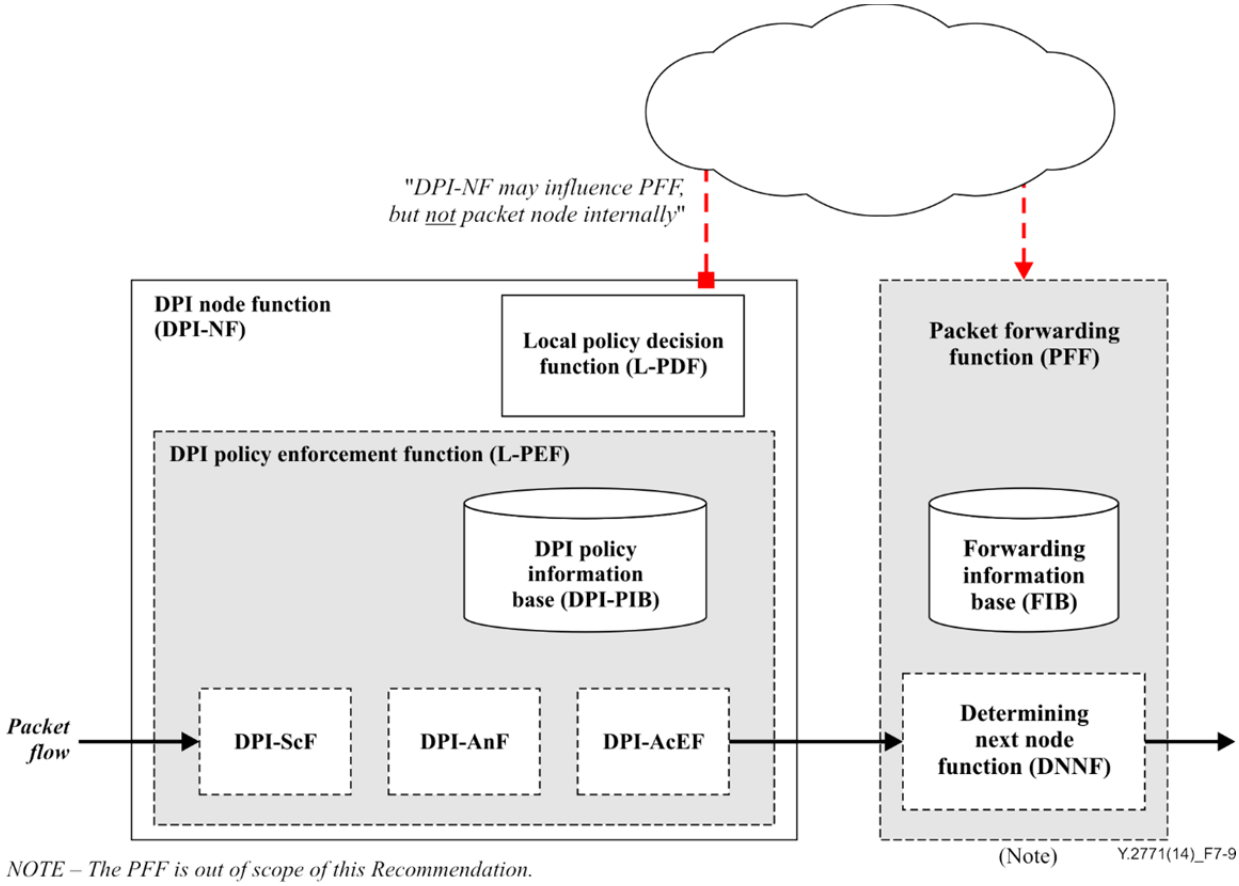
Stateless DPI relates to DPI policy rules which are enforced on each packet individually, without any correlation to other packets of a unidirectional packet flow or the bidirectional packet connection (see "state condition" in clause 3.2.11 in [ITU-T Y.2770]).

Statefull DPI implies such a correlation (Note), which may be modelled by a (finite) state machine. Such a functional model would be specific to concrete DPI policy rules and is thus outside the scope of this Recommendation.

NOTE – For instance, an IP application with TCP-based transport of application data. There might be dedicated policy conditions for the TCP connection establishment phase and other policy conditions for the subsequent active communication phase.

7.1.6 Impact of DPI on packet forwarding

The DPI node function may impact the subsequent packet forwarding function (PFF), under the condition that a PFF is available or is a "non-empty" PFF (see clause 7.1.2). However, the DPI-NF shall not be authorized for local packet node PFF modifications. This is rather an option, driven by external packet node remote network elements (see Figure 7-9). The specific impact is subject to policy decisions and/or dedicated DPI policy rules, and thus outside the scope of this Recommendation.



NOTE 1 – The PFF is out of scope of this Recommendation.
NOTE 2 – The PFF is only present for the in-path DPI mode.

Figure 7-9 – Possible impact of DPI on packet forwarding via a remote network entity

Examples:

- Update of FIB (network route information; see also Figure 7-5) due to observed network performance related to network topology;
- Update of FIB by blocking certain network routes (e.g., in reverse direction in case of bidirectional DPI).

It should be noted that any local modification of the PFF, driven by the DPI-NF and ordered by external network elements, must be in concert with the underlying network-wide framework with respect to packet forwarding, switching or/and routing concepts (e.g., given by an IPv6 DiffServ domain, an MPLS domain, a L2VPN topology, etc.).

7.1.7 DPI within packet-based networks and NGN environments

The DPI node function is embedded in a functional, physical or virtual packet node, which may interact with other functions in a packet-based network. Figure 7-10 provides an example environment. [ITU-T H.248.86] defines a DPI control technology when the DPI-PDFE and DPI-FE are mapped onto a decomposed gateway model.

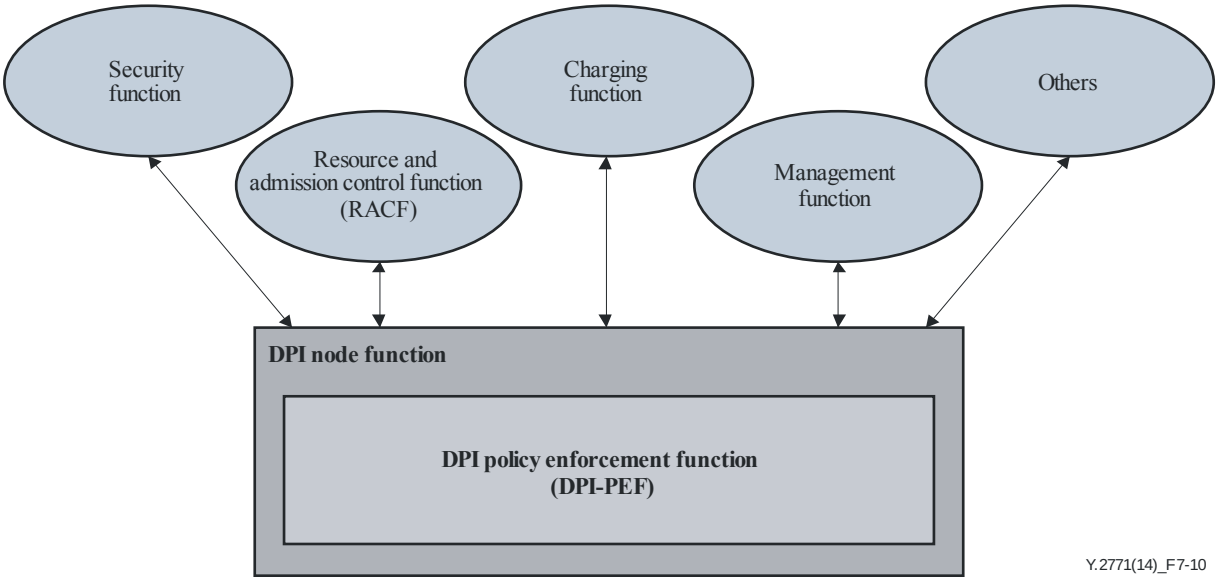


Figure 7-10 – DPI within packet-based networks and NGN environments

7.1.8 Functional models for support of IETF flow metering

7.1.8.1 Characteristics of IETF IPFIX flow metering function

The IETF IPFIX flow metering function is defined in [IETF RFC 5101] (see also clause 6.3.3.2 in [ITU-T Y.2770]). The notion of "flow" refers to a packet flow according to clause 3.1.3 of [ITU-T Y.2770] and "metering" to the measurement of performance metrics. The IPFIX flow metering function incorporates therefore a part with policy rule conditions for packet flow identification (based on the IPFIX flow identifier (see clause 3.2.17 in [ITU-T Y.2770]) and a part related to policy rule actions for measurements and reporting actions.

7.1.8.2 Embedded flow metering function

The IETF IPFIX flow process may be abstracted and described as a DPI policy rule, therefore directly mapped onto a DPI-FE. Figure 7-11 illustrates such an embedded flow metering function. The reporting of measurement results could be communicated across external interfaces e2 or e1.

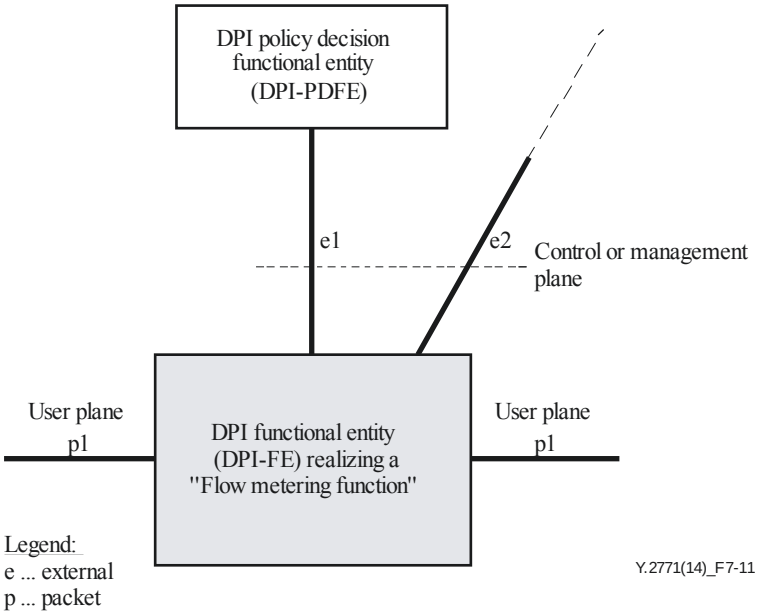


Figure 7-11 – Embedded flow metering function

7.1.8.3 Remote flow metering function

A flow metering function could be also remotely located in the packet path p1, before (see Figure 7-12) or after a DPI-FE. The network configuration with a remote flow metering function could be e.g., motivated when an existing network provides a physical entity with flow metering support and a DPI-PE is additionally deployed.

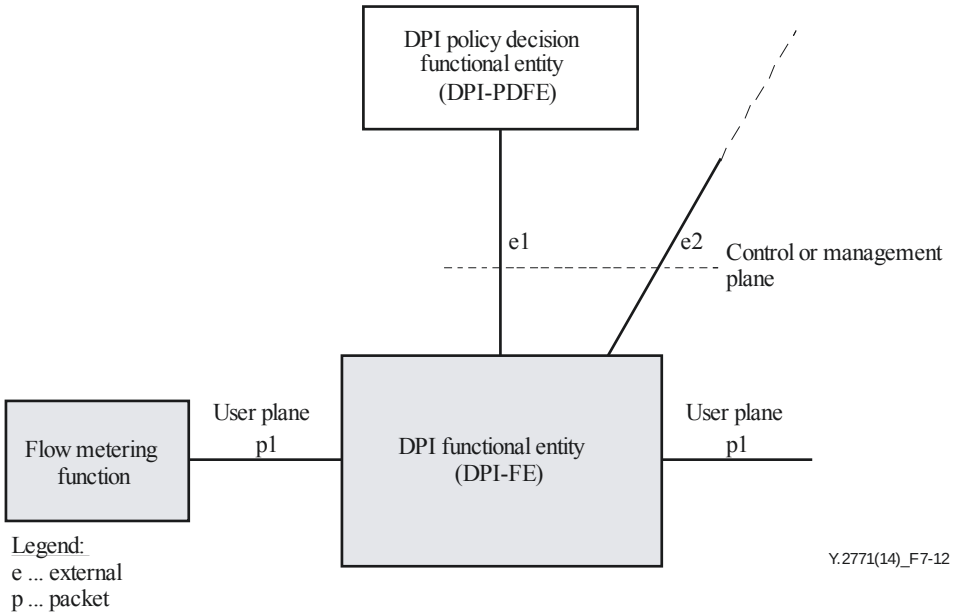


Figure 7-12 – Remote flow metering function

The DPI-FE and the remote flow metering function may be considered as a distributed architecture with two variants:

1) Decoupled architecture: both functional entities are only connected via p1, thus are fully decoupled in the control and management plane.
2) Coupled architecture: e.g., results of the flow metering function could be also reported via e2 of a DPI-FE. Such a functional sharing model would imply an additional, new interface (besides p1) between both functional entities.

Use cases for both variants are for further study.

7.1.9 Probabilistic DPI

7.1.9.1 Probabilistic DPI in general

The packet identification process may be inherently of a statistical nature (as opposed to deterministic packet identification), i.e., the identification result (such as a match criteria) is associated to a probability. This kind of packet inspection is called probabilistic DPI, also known as uncertain DPI.

Probabilistic DPI relates to DPI policy rules whose conditions are for instance:

– A signature set 'S' contains signatures 'S_1', 'S_2', …, 'S_N'. The signature set 'S' represents a compound policy condition according to a Boolean disjunctive normal form (DNF), i.e., signatures 'S_i'(i=1, 2…N) are combined as an OR-based list. The two DPI steps have the following characteristics:
step 1: the generation of the policy conditions is based on a probabilistic process; and
step 2: the execution of the policy conditions leads to deterministic match results, thus leading overall to probabilistic DPI.

The primary objective of probabilistic DPI is to determine quickly and efficiently whether a packet matches signature set 'S'. The concrete match information, i.e., which specific signature 'S_i' was identified, is secondary. Signature set 'S' represents in general an identification option for a particular application. The associated application tag, e.g., 'Packet with signature in Set S', might be the subject of reporting.

7.1.9.2 Bloom filter based probabilistic DPI

A bloom filter-based DPI is a well-known representative of probabilistic DPI, due to the inherent error rate ε_{DPI}, in terms of false positives $\varepsilon_{f\text{-}p}$ (see clause 8.2.3.3.1), of the underlying packet inspection approach.

An application-specific example realized as a bloom filter is considered:

– The DPI application is a detection of "application x traffic". The traffic of application x is characterized by a signature set S = {'application x v1', 'application x v2', …, 'application x vk'}, i.e., individual signatures concerning application-specific characteristics. There might be then an example DPI policy rule to detect whether a packet contains "application x", using the above signature set as compound DPI policy rule condition, and just discard the packet when a match happens without the need to know what the exact application x version is.

  The primary motivation of bloom filter based probabilistic DPI is a kind of trade-off between identification accuracy and identification resource consumption (e.g., in terms of CPU time or/and memory). Such an approach allows simplification of DPI processing significantly.

The probabilistic characteristic of a bloom filter based DPI is given by following process:

– Any arriving packet 'P' is compared against the bloom filter which represent the entire signature set 'S' in parallel; if a packet 'P' matches one or multiple signatures of set 'S', then the result would be a hit with an estimated information certainty of probability $P_{Hit,BloomFilter,'S'}$, according to following equation:

$$P_{Hit,BloomFilter,'S'} = 1 - \varepsilon_{f-p} = 1 - \left(1 - e^{-kN/m}\right)^k \quad (7\text{-}1)$$

with parameter values:

m = size of bloom filter in bits

N = number of signatures in set S

k = number of hash functions used for the creation of the bloom filter.

The notion of "probabilistic" is related to the "identification accuracy" of a DPI-FE concerning the identification of a packet, flow, application etc., and tightly coupled with performance metrics of the area of error rates (see clause 8.2). For probabilistic DPI realized by bloom filters, see Appendix I; the performance indicator "(DPI) false-positive error rate" $\varepsilon_{f\text{-}p}$ and performance indicator "(DPI) false-negative error rate" $\varepsilon_{f\text{-}n}$ is equal to zero.

7.2 Information and data models

The staged development process of communication services [b-ITU-T I.130] differentiates the abstraction levels of "information" and "data" (such as protocol data units). Information models are used at a very high level in order to describe for instance the information flow between network entities (see e.g., [b-ITU-T I.130], [b-ITU-T X.1036]). Data models are at a lower level in order to describe e.g., an information element from a syntactical viewpoint (see e.g., [b-ITU-T J.380.1]).

7.2.1 Information model (example framework)

Any kind of detailed specification for modelling DPI policy rule related information and information flows are outside the scope of a "framework" type of Recommendation. However, DPI policy rule information modelling [b-IETF RFC 3060] may be considered as an example as a baseline. Table 7-1 indicates some example information elements in order to provide some guidelines (how a concrete model could be developed):

Table 7-1 – Example information model, based on [b-IETF RFC 3060]

Information element (this Recommendation and [ITU-T Y.2770])	Generic policy core information model according to [b-IETF RFC 3060]
DPI policy rule	Could be based on class "PolicyRule"
DPI policy rule (compound) condition	Could be based on abstract class "PolicyCondition"
DPI policy rule (single) condition	Could be based on abstract class "PolicyCondition"
DPI policy rule action	Could be based on abstract class "PolicyAction"
And so on, such as grouping and prioritizing DPI policy rules, and characteristics such as validity period of rules, etc.	…

7.2.2 Data model (example framework)

Any kind of detailed specification for modelling DPI policy rule related data objects are outside the scope of a "framework" type of Recommendation (because then the area of protocol syntax development would be opened).

However, DPI policy rule data modelling [b-IETF RFC 4011] may serve as an example of a baseline. Table 7-2 indicates some example data objects in order to provide some guidelines (how a concrete model could be developed):

Table 7-2 – Example data model, based on [b-IETF RFC 4011]

Information element (this Recommendation and [ITU-T Y.2770])	Generic data object, using the policy based management MIB according to [b-IETF RFC 4011] as an example
DPI-PIB	Could be based on object "pmPolicyTable", which is again linked with object "pmPolicyCodeTable" (Note 1)
DPI policy rule, i.e., entry of a DPI-PIB	Could be based on object "pmPolicyEntry"
DPI policy rule condition	Could be based on object "pmPolicyCodeEntry" (Note 2)
DPI policy rule action	Could be based on object "pmPolicyCodeEntry" (Note 2)
Etc.	Etc.
NOTE 1 – The abstraction and separation of "rule description" and "rule code" in two associated tables allows the definition of an efficient "DPI-PIB". NOTE 2 – That is, rule condition and rule action could finally use the same data object model (in this example).	

Network plane perspective: it may be noted that a generic data object could be visible as a:

- managed object from the DPI management plane perspective (see interface e2 in clause 8 of [ITU-T Y.2770]); or/and as
- controlled object from the DPI control plane perspective (see interface e1 in clause 8 of [ITU-T Y.2770]).

For example, a particular DPI policy rule could be signalled (via e1) by the DPI PD-FE, or provisioned (via e2) by a DPI management system, but leading to the same data object "DPI policy rule" entry in the DPI-PIB.

7.3 Traffic models

7.3.1 Introduction

The purpose of this clause is the description of some interesting characteristics of DPI entities (definition see clause 3.2.7 in [ITU-T Y.2770]) from the perspective of traffic theory. The derived aspects may assist in the subsequent definition of e.g., functional, performance and availability requirements, the indication of architectural aspects, or performance evaluations.

The described traffic models are just examples and not necessarily representative for a particular DPI component (such as a DPI-PE, DPI-FE, DPI engine, DPI signature library, etc.).

7.3.2 Basic traffic models for packet path processing

Traffic handling is at the granularity of packets, the main functions of a DPI-FE, and it is mainly realized by the embedded DPI engine (for the definition see clause 3.2.6 in [ITU-T Y.2770]).

A DPI engine is the core component of a DPI-FE and plays an important role in the DPI-FE. DPI traffic is processed by the DPI engine. When a DPI engine would be realized as a physical component, parallel processing may help to improve the performance of the DPI engine. There might be consequently more than one processing unit within the physical component corresponding to a DPI engine.

7.3.2.1 Basic traffic model of a DPI-FE with scope on the DPI engine

Figure 7-13 uses the example functional model of a DPI-FE according to Figure 6-1 of [ITU-T Y.2770] in order to derive a typical traffic model. The traffic model focuses on the packet path only, and thus provides a model for a DPI engine.

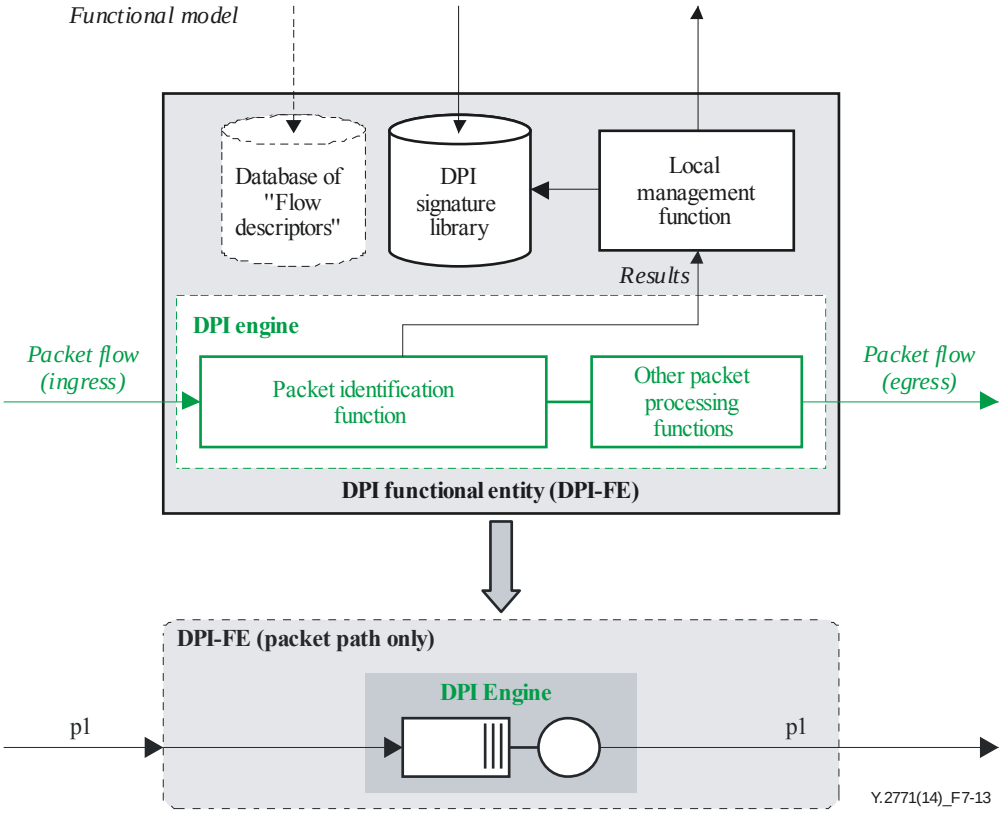


Figure 7-13 – Basic traffic model of a DPI-FE with scope on a DPI engine

The model is characterized by a single server and a finite waiting queue. The server processes therefore all packet path functions. The 1-stage server model represents a traffic model for an unidirectional packet flow.

7.3.2.2 DPI engine: extension to multi-stage packet path processing

7.3.2.2.1 DPI engine based on dual-stage server

Figure 7-14 provides an example of a dual stage traffic model of a DPI engine.

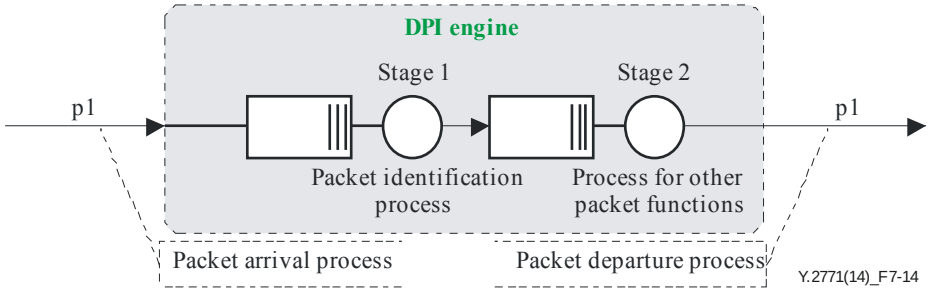


Figure 7-14 – Traffic model DPI engine: extension to dual-stage packet path

The first server is responsible for the processing of DPI rule conditions in this example.

7.3.2.2.2 DPI engine based on a 3-stage server

- A DPI engine could be internally realized as a distributed system, e.g., as a series of concatenated processing elements. For instance, the example functional model according to Figure 7-3 represents three processing stages, called "DPI scanning", "DPI analysing" and "DPI action execution", abbreviated as DPI-ScF, DPI-AnF and DPI-AcEF, respectively.

Figure 7-15 provides an example of a correspondent traffic model.

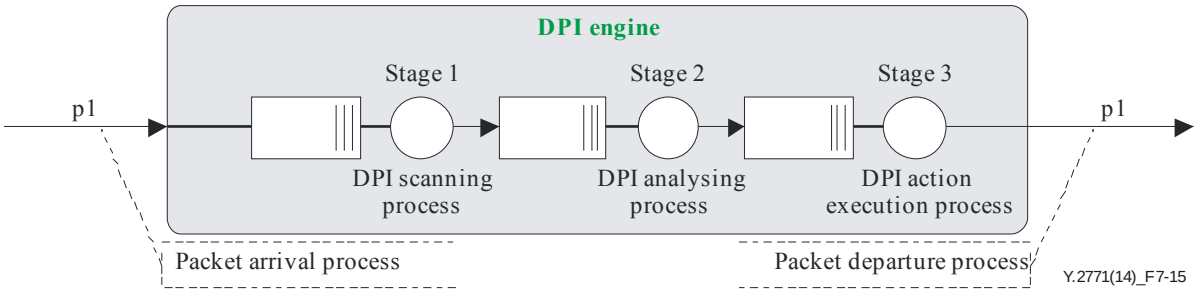


Figure 7-15 – Traffic model DPI engine: extension to 3-stage packet path

7.3.3 Extended traffic models for DPI engines

7.3.3.1 Single external interface and internal parallelism

Example: there might be an out-of-path DPI entity (see clause 6.1) which is connected via a single network route to the packet network. The purpose of such a DPI entity might be the off-line processing of a large number of selected packet flows, i.e., a high processing capacity may be required. Parallelism might be an option for achieving high processing capacity. Figure 7-16 provides an example traffic model, at the level of multiple parallel DPI engines.

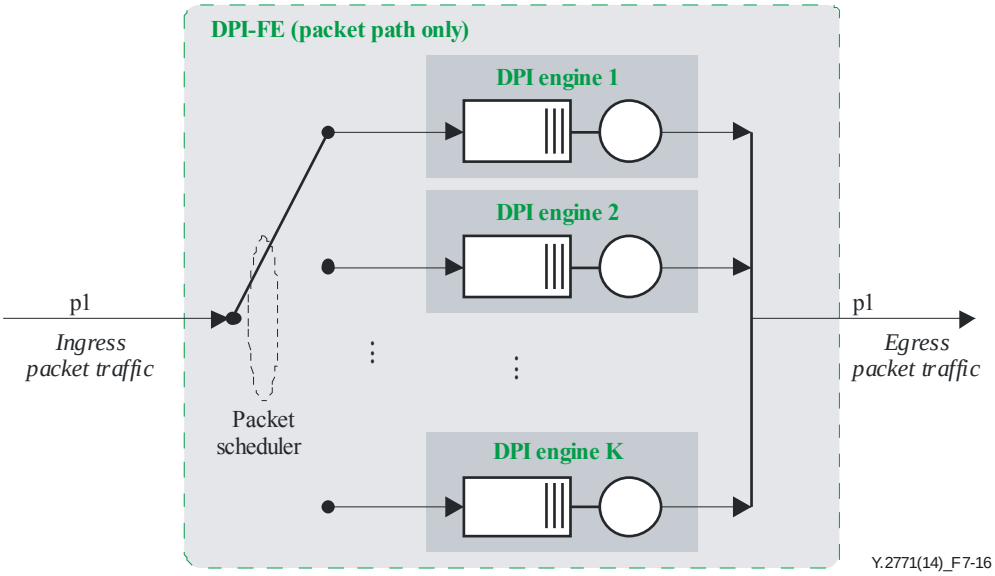


Figure 7-16 – DPI engines – Single external interface and internal parallelism

The traffic model implies a packet scheduler function for assignment of an incoming packet to a DPI engine (server). Such a function is outside the scope of this Recommendation.

NOTE – For instance, the packet scheduler could be:

a) simply a load balancing algorithm (i.e., scheduling based only on the estimated load state of the DPI engine servers), which actually only makes sense for a stateless DPI;

b) based on flow descriptor information (in order to address a statefull DPI), but then there would be at least a 2-stage server model from a traffic modelling viewpoint; or

c) any other kind of scheduling method.

7.3.3.2 Multiple external interfaces and internal parallelism

An in-path DPI entity located at the network core level (see clause 6.1) typically provides multiple physical p1 interfaces. Multiple DPI engines may serve in parallel all ingress packet flows (see

Figure 7-17). There is usually the requirement that all DPI engines (e.g., *K*) should be connected to *all* ingress packet interfaces p1 (e.g., *N*). Thus, a *N-to-K* packet switching fabric function would be required for that purpose. Such a function is outside the scope of this Recommendation.

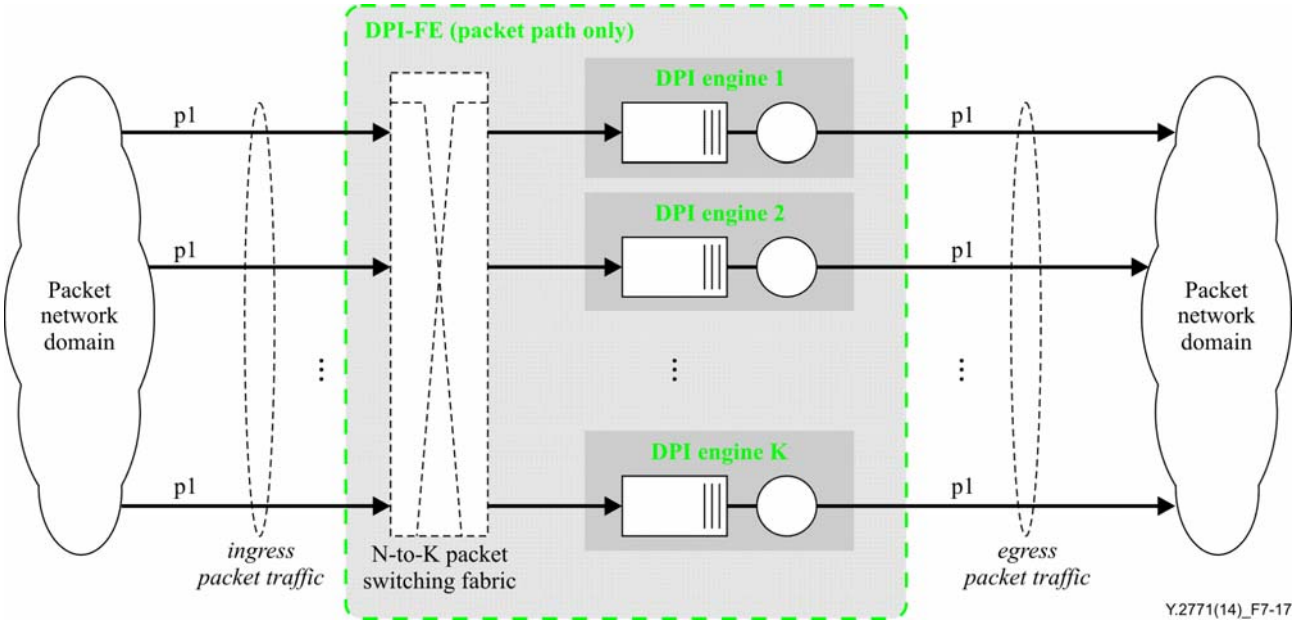


Figure 7-17 – Multiple external interfaces and internal parallelism

7.3.3.3 Parallel DPI engines based on 3-stage server models

Figures 7-18 and 7-19 illustrating an extended model, based on the combination of 3-stage DPI engine models (clause 7.3.2.2.2) and parallelism at the level of DPI engines (clause 7.3.3.1).

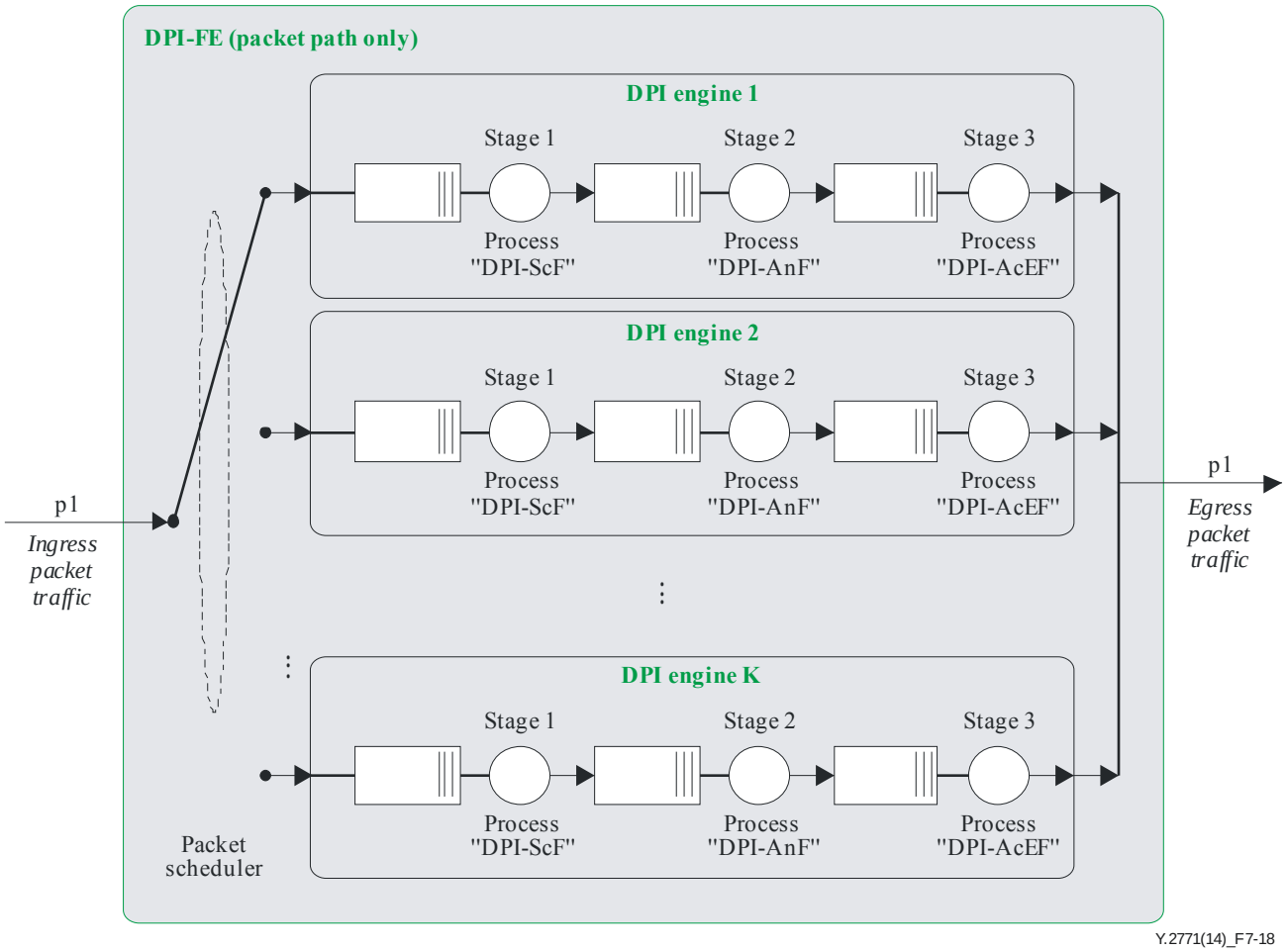


Figure 7-18 – Parallel DPI engines based on 3-stage server models (single external interface)

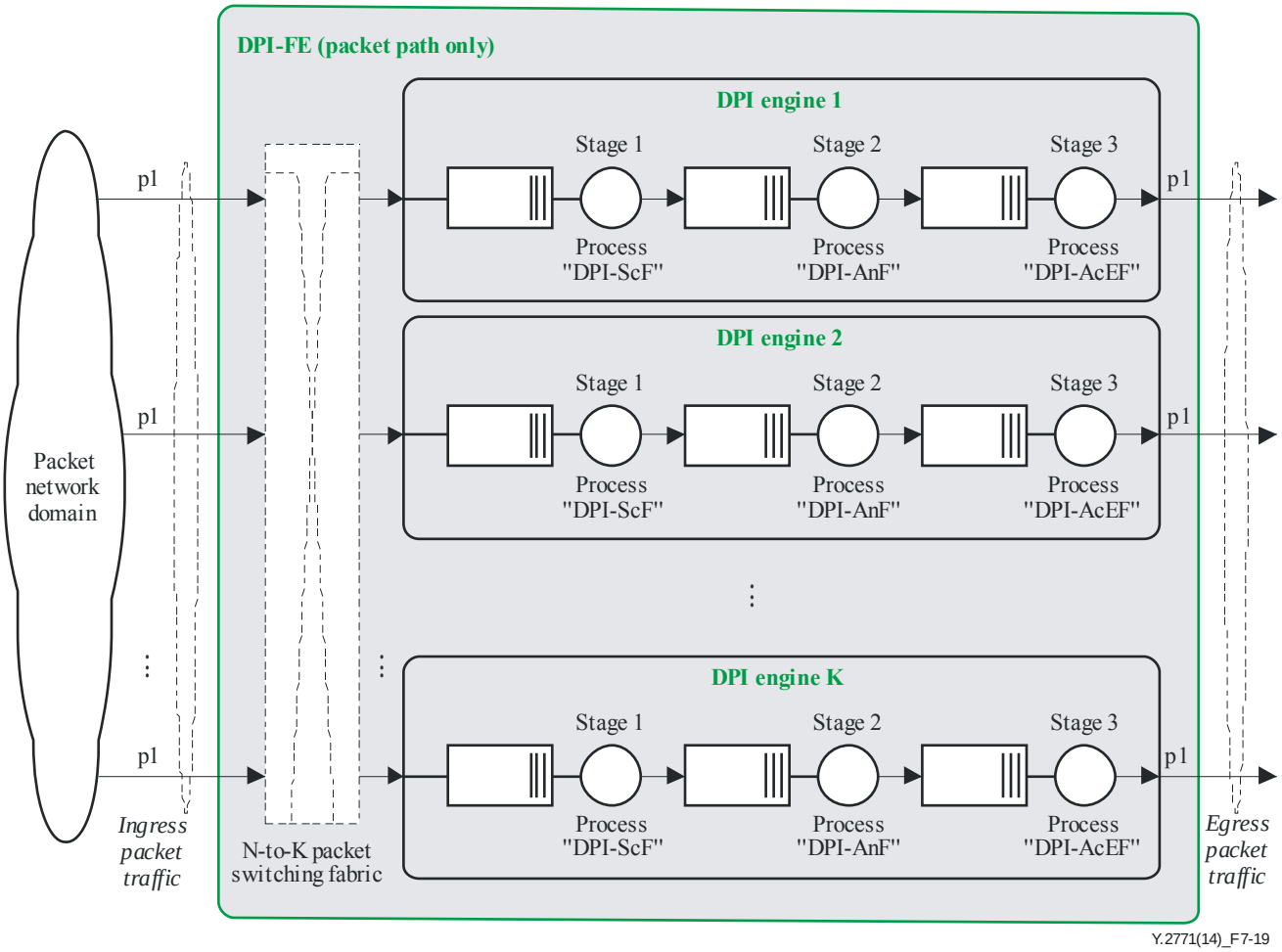


Figure 7-19 – Parallel DPI engines based on 3-stage server models (multiple external interfaces)

The traffic models are characterized by parallel DPI engines running fully concurrently, i.e., there are no dependencies between DPI engines. The maximum performance of such a DPI-PE architecture would be achieved when all servers would be "optimally loaded" (i.e., no server is overloaded or underloaded), which implies a homogeneous load distribution in both dimensions of the traffic model. The engineering of such architecture is fairly challenging, perhaps possible only for some specific traffic distributions concerning offered packet load.

Such a background may lead to different architectures, as e.g., considered in the subsequent clause.

7.3.3.4 Single DPI engine based on three stages and internal parallelism

Figures 7-20 and 7-21 illustrate an example of a single DPI engine, based on three processing stages and parallelism per stage. The level of parallelism may differ per stage, i.e., there might be a different number of servers per stage (i.e., different values of K1, K2 and K3).

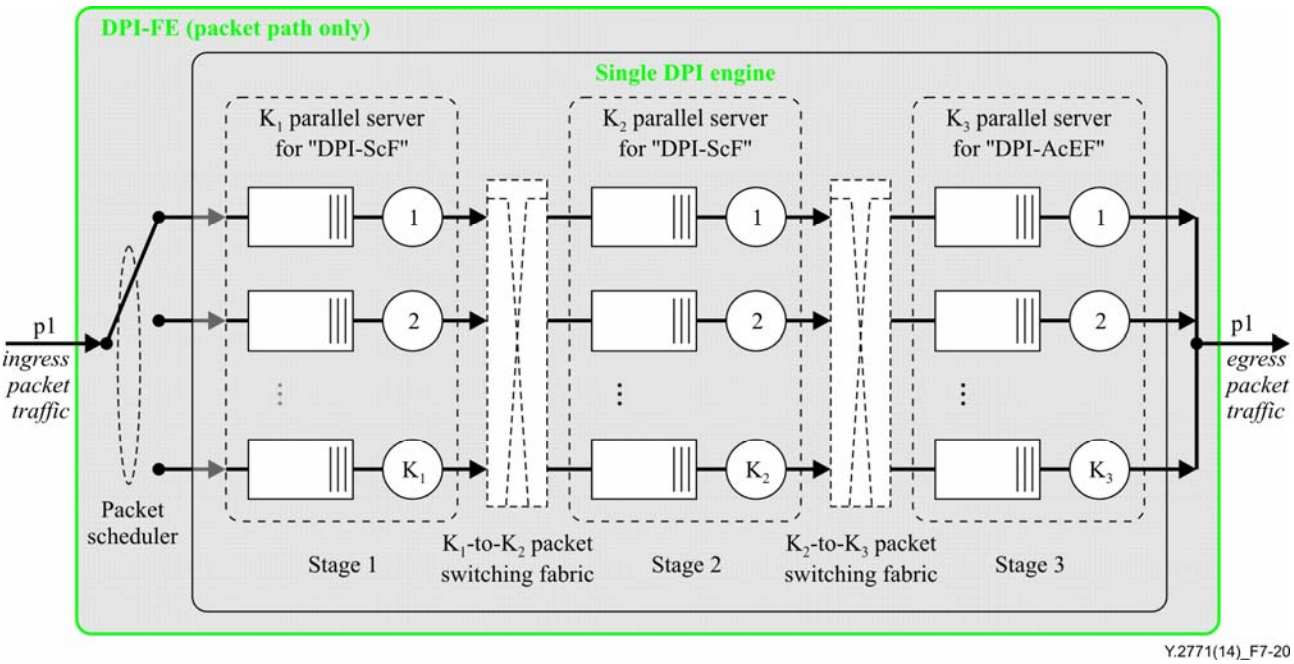


Figure 7-20 – Single DPI engine based on three stages and internal parallelism (single external interface)

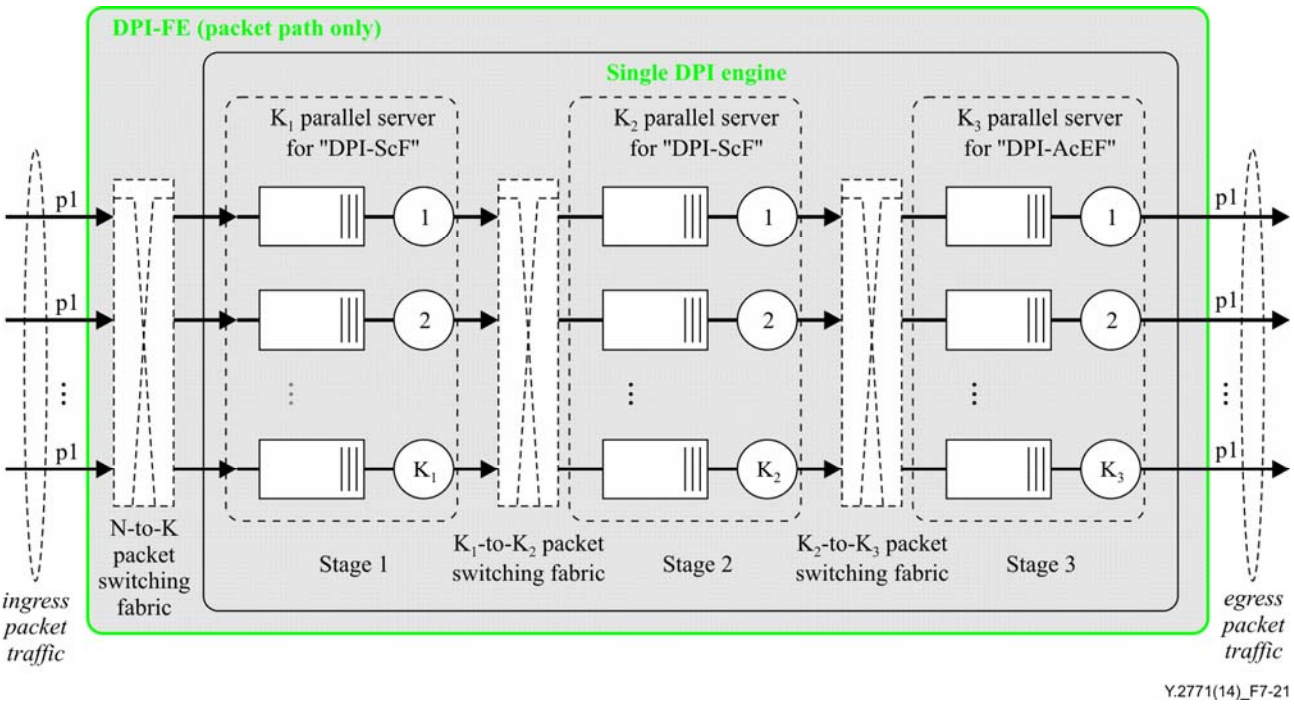


Figure 7-21 – Single DPI engine based on three stages and internal parallelism (multiple external interfaces)

When the DPI engine has to support a statefull DPI, then *all* packets of the *same flow* may need to be routed across the same path of servers due to local "state information". This aspect is outside the scope of the above traffic model.

7.4 Identification of possible DPI-FE subcomponents

A DPI-FE may be partitioned in functional subcomponents, as already illustrated by the example functional models in the previous clauses. Table 7-3 provides an overview of typical functional

subcomponents within a DPI-FE. The components are partially referred to by subsequent clauses (e.g., in the case of the discussion of performance aspects, possible functional or operational requirements, etc.).

Table 7-3 – Typical DPI-FE subcomponents

Component	Description
DPI policy enforcement function (DPI-PEF):	A functional element concerning the enforcement of DPI policy rules, which contains at least a DPI policy information base, a DPI packet identification function and a DPI action execution function
Further detailing the DPI-PEF: 1) Packet processing path	
1.1) DPI packet identification function (DPI-PIF) Example functional decomposition:	Functional element responsible for the processing of DPI policy conditions against incoming packet traffic
1.1.1) DPI scan function (DPI-ScF):	A functional element (as part of the DPI-PIF) to perform initial comparison functions, which are determined by DPI policy rule conditions
1.1.2) DPI analyser function (DPI-AnF):	A functional element (part of the DPI-PIF) to perform subsequent comparison functions, which are also determined by DPI policy rule conditions (e.g., related to packet header elements or the contents (in packet payloads)
1.2) DPI action execution function (DPI-AcEF):	A functional element to perform operations on considered packets, according to identified DPI policy rule actions
2) DPI policy information base (DPI-PIB; Note 1) function:	A functional element representing a database, which contains a set of one or multiple DPI policy rule entries (see below)
a) DPI policy rule entry:	The entry in a table which contains a DPI policy rule (Note 2)
i) DPI policy rule condition (briefly 'rule condition'):	An expression (typically of type Boolean). A condition is also known as a match criterion (e.g., due to condition types representing a partial match, full match, prefix match, longest prefix match, etc.)
ii) DPI policy rule action (briefly 'rule action'):	An action performed after the evaluation of all rule-specific policy conditions and the conclusion for executing that action
NOTE 1 – Also known as *rules table, policy signature library or just signature library*. NOTE 2 – There might be one or multiple rules enforced. Such rules may be statically predefined (via configuration management of the packet node, called DPI policy management), or signalled (via a policy control interface) or dynamically, locally generated (via a local PDF). DPI policy rules are used to compare protocol control information (PCI, i.e., packet header elements) or the payload/contents of the packet flows with a set of conditions to determine if the string matching is successful or not.	

7.5 Fault tolerance models

Reliability and availability of a network node (e.g., DPI node) are very important to the network in which the network node is deployed. When the network node is out of service (see e.g., the operational state model in [ITU-T X.731]), this could result in a network disaster, where it could be possible that all network users are forced offline. This would result in a massive loss of valuable

information. So it is critical to pursue high reliability and availability of a network node. As a kind of network node, a DPI node should also support high reliability and availability.

By using the fault tolerance method, the DPI "1+N" redundancy group aims at improving reliability and availability of the network deployed with DPI nodes.

Reliability of the DPI "1+N" redundancy group can be calculated by the follow parameters:

1) MTBF: mean time between failures is the average time between failure of a DPI "1+N" redundancy group.

2) MTTR: mean time to repair is the time taken to make the failed DPI "1+N" redundancy group work normally again.

Availability of a DPI "1+N" redundancy group can be calculated by the follow formulas (see [ITU-T G.602]):

3) Availability = uptime / (downtime + uptime) or

4) Availability = MTBF / (MTBF + MTTR).

5) Within a DPI "1+N" redundancy group, the number of redundancy functional components is dependent on a special realization, and it is outside the scope of this Recommendation. Functional components within a redundancy group work in active/backup mode, and only one functional component works as an active functional component, while the other functional components work as backup functional components. When the active functional components are out of service, one and only one of the backup functional components will become the new active functional component and the former active functional component will change to a backup functional component.

6) The interface between the active functional component and a backup functional component, which is used for a switchover between active and backup functional components is DPI-independent and implementation specific, thus outside the scope of this Recommendation.

7) Multiple fault tolerance models are presented, illustrating the DPI node level fault tolerance model (clause 7.5.1), DPI PEF level fault tolerance model (clause 7.5.2), DPI PIB level fault tolerance model (clause 7.5.3) and DPI engine level fault tolerance model (clause 7.5.4).

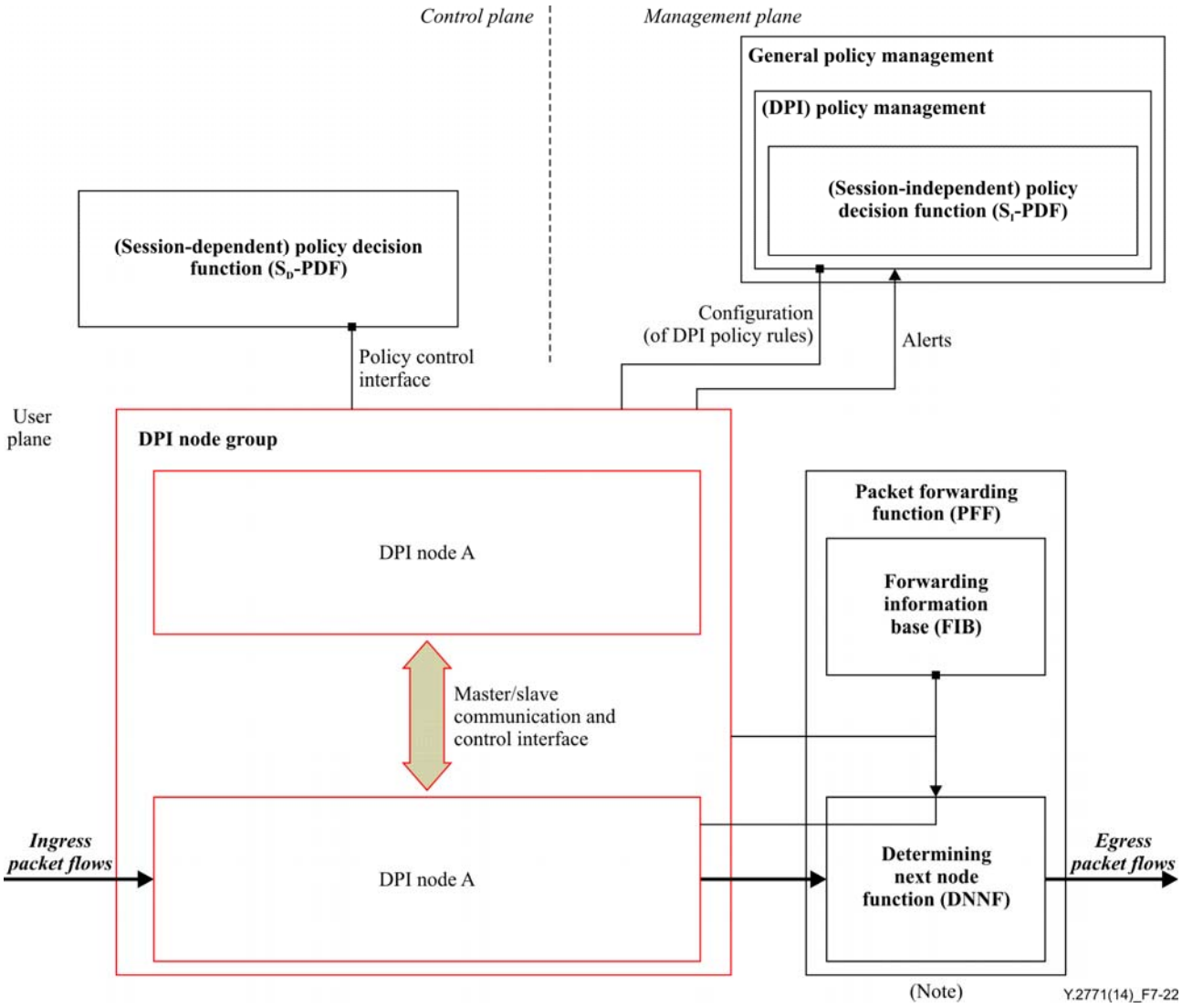
8) All fault tolerance models are based on a DPI "1+N" redundancy group (in other words, redundancy of the functional components, e.g., DPI nodes in Figure 7-22).

9) The active functional components and the backup functional components should keep totally identical information such as PIB through a data synchronization method. The data synchronization method depends on a special realization and it is outside the scope of this recommendation.

7.5.1 DPI node level fault tolerance model

Figure 7-22 depicts a DPI model with DPI node level reliability guaranteeing that two or more DPI nodes are deployed together to form a DPI node group (a DPI "1+N" redundancy group of which the DPI nodes are the functional components), and one DPI node works as the active DPI node while the others work as backup DPI nodes. Moreover, the active and backup DPI nodes have to duplicate internal information as required for normal operation. Once the active DPI node is out of service, one of the backup DPI nodes will automatically become the active DPI node.

Although only two DPI nodes are depicted in Figure 7-22, the fault tolerance model is similar when there are more than two DPI nodes (due to the "1+N" redundancy concept).



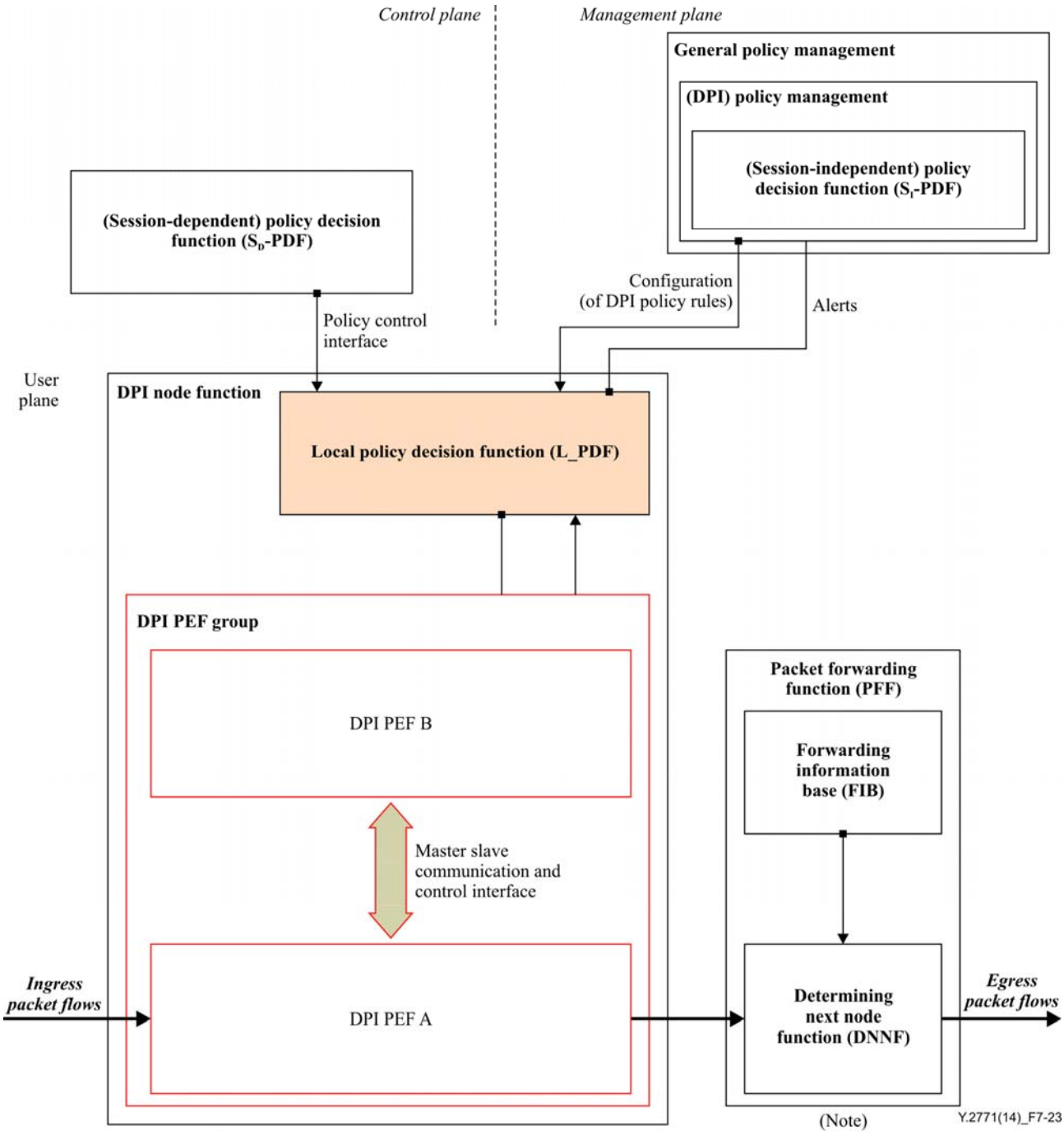
NOTE – The PFF is out of scope of this Recommendation.

Figure 7-22 – DPI model with DPI node level reliability guaranteeing

7.5.2 DPI PEF level fault tolerance model

Figure 7-23 depicts a DPI model with DPI PEF level reliability support, two or more DPI PEF components (in other words, a DPI "1+N" redundancy group of which the DPI PEF components are the functional components) are designed within a DPI node, and one DPI PEF component works as the active component while the other DPI PEF components operate as backup components. The switchover procedures in failure situations are similar to those at node level reliability support (see clause 7.5.1).

Although only two PEF components are depicted in Figure 7-23, the fault tolerance model is similar when there are more than two PEF components.



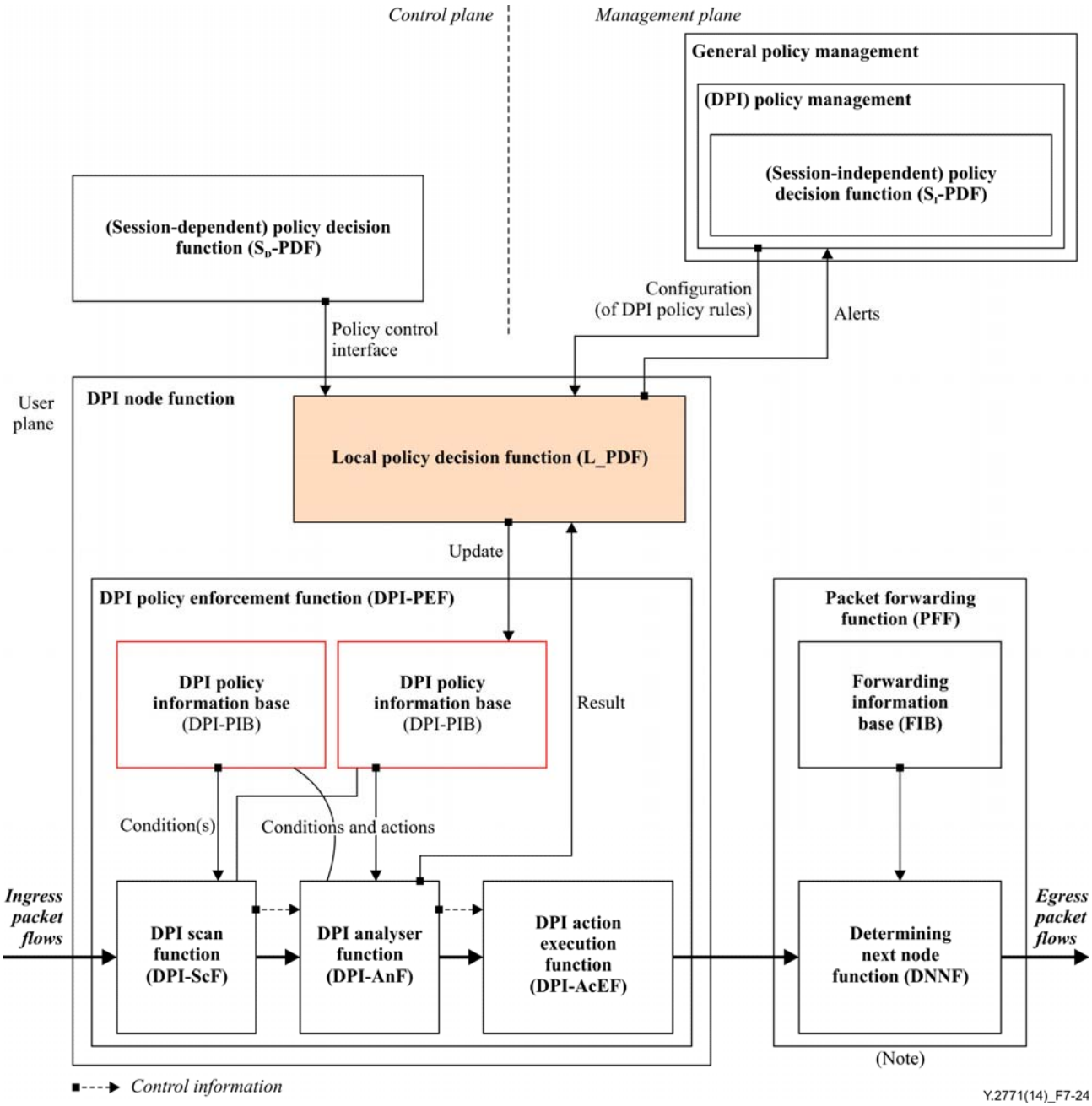
NOTE – The PFF is out of scope of this Recommendation.

Figure 7-23 – DPI model with DPI PEF level reliability support

7.5.3 DPI-PIB level fault tolerance model

Figure 7-24 depicts a DPI model with DPI PIB level reliability support, two or more copies of DPI PIB (in other words, two or more copies of the DPI PIB constitute a DPI "1+N" redundancy group) are located in a DPI node, and all DPI PIBs are synchronized, and thus contain identical information. One DPI-PIB is assigned the active role, and the other PIBs realize backup roles. Once the active PIB is out of service, one of the backup PIBs will automatically become the active one.

Although only two DPI PIBs are depicted in Figure 7-24, the fault tolerance model is similar when there are more than two DPI PIBs.



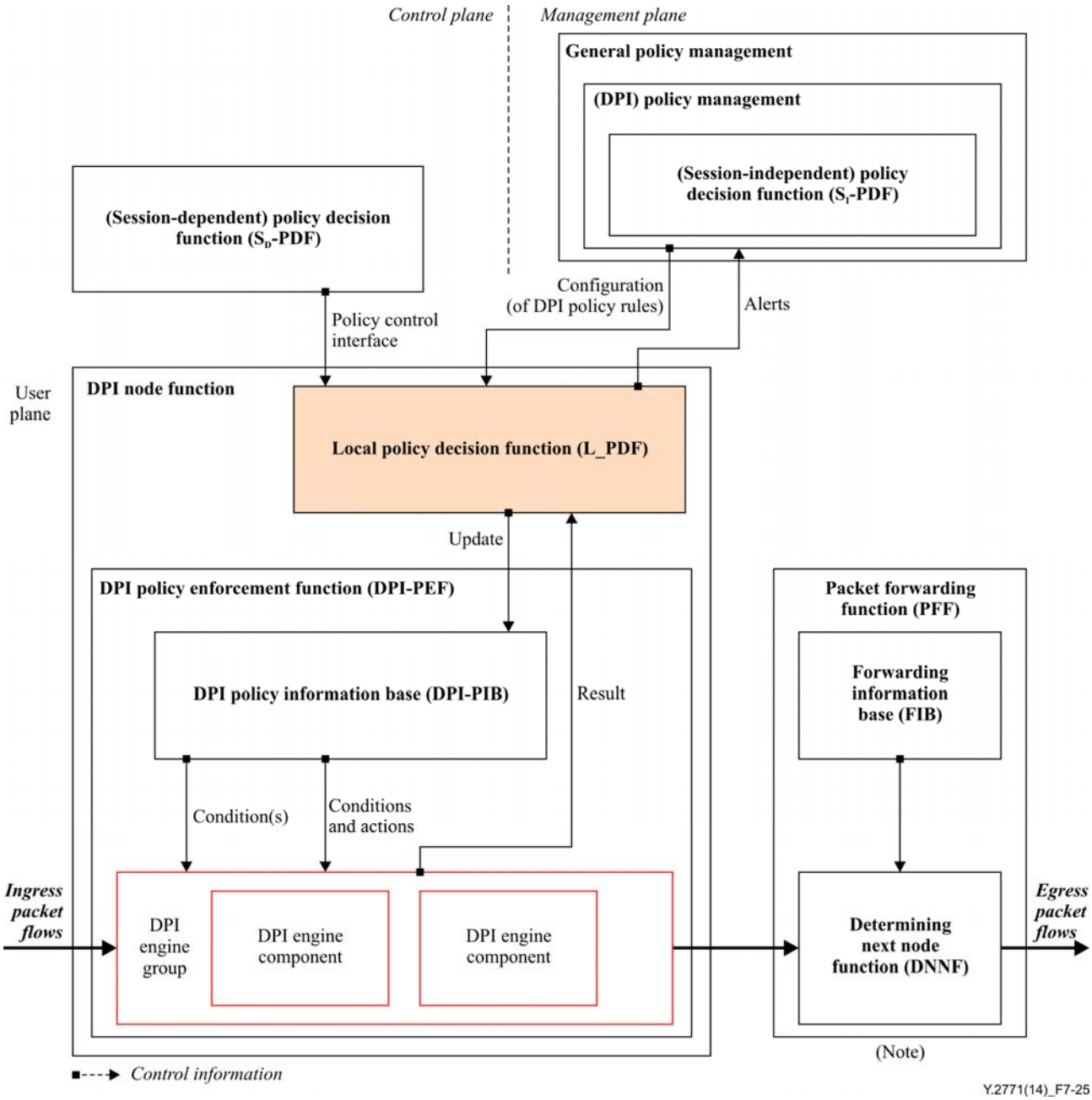
NOTE – The PFF is out of scope of this Recommendation.

Figure 7-24 – DPI model with DPI PIB level reliability guaranteeing

7.5.4 DPI engine level fault tolerance model

Figure 7-25 depicts a DPI model with DPI engine level reliability support. The fault tolerance principles are the same as in previous models at higher levels.

Although only two DPI engine components are depicted in Figure 7-25, the fault tolerance model is similar when there are more than two DPI engine components.



NOTE – The PFF is out of scope of this Recommendation.

Figure 7-25 – DPI model with DPI engine level reliability guaranteeing

8 Performance framework

8.1 Purpose and scope of performance considerations

This clause describes a framework and proceeding for identifying and developing performance metrics related to DPI. These metrics can be used to characterize the behaviour of DPI entities.

The performance framework covers basically the areas of:

1) performance metrics:

   The capacity, availability and performance of a DPI entity may be characterized by performance metrics. The prime purpose is:

   a) to clarify whether existing, non-DPI performance metrics could be reused;

b) to recognize DPI-specific performance metrics, which are already introduced by other organizations working on DPI;

c) to identify new DPI-specific performance metrics, which implies the definition of such a metric; and

d) to classify the set of metrics into key performance indicators (KPIs) and other metrics.

2) performance requirements:

Such kinds of DPI requirements are associated to particular performance metrics. Implementation dependent performance requirements are outside the scope of this Recommendation. Thus, it is possible to derive qualitative or relative performance requirements. The specification of additional quantitative or absolute performance requirements is only possible for some limited areas (e.g., if the maximum node transfer delay budget would be subject to an overall end-to-end network consideration…).

3) performance benchmarks:

Benchmarking is a fairly challenging area with respect to the identification and specification of well-recognized and meaningful benchmark scenarios. The definition of DPI performance benchmarks is basically outside the scope of this Recommendation. However, this Recommendation could provide information and guidance on aspects of consideration when trying to specify a performance benchmark for DPI entities.

The definition of new performance metric types (also known as performance indicator) should basically follow the guidelines of [b-IETF RFC 6390], and therefore principally incorporate at least:

- metric name and metric description;
- method of measurement or calculation;
- measurement unit; and
- a performance metric definition shall not be tied to a statistical parameter like min, max, mean, PDF, variance, etc. Such aspects are rather subject to a requirements specification.

8.2 Performance metrics

8.2.1 Overview – Performance indicators for DPI nodes

Performance requirements are related to performance metrics. The crucial metric types are called key performance indicators (KPIs), which represent a subset of the overall set of metrics.

8.2.1.1 Guidelines for classifying DPI-related performance metrics as KPIs

This Recommendation uses the following KPI definition (NOTE 1 – derived from [ETSI TS 132 410]):

- **Key performance indicators (KPIs) in general**:

  Are the primary metrics to evaluate process performance as indicators of the core function of the network element.

- **Key performance indicators for DPI entities** (KPI_{DPI}):

  A KPI_{DPI} – as in the scope of this Recommendation – does therefore characterize the performance of the DPI engine (see clause 3.2.6 of [ITU-T Y.2770]; also known as DPI packet processing path).

NOTE 2 – The notion of performance indicator (PI) is synonym to performance metric in this Recommendation.

In order to classify DPI performance metrics as KPIs or non-KPIs, the following criteria should be considered. A KPI for DPI entities should fulfil the following conditions:

1) Performance metric should be tied to the packet processing path itself, where DPI policy rules are enforced on packet objects (hence, not any other DPI functions outside the DPI packet path); and

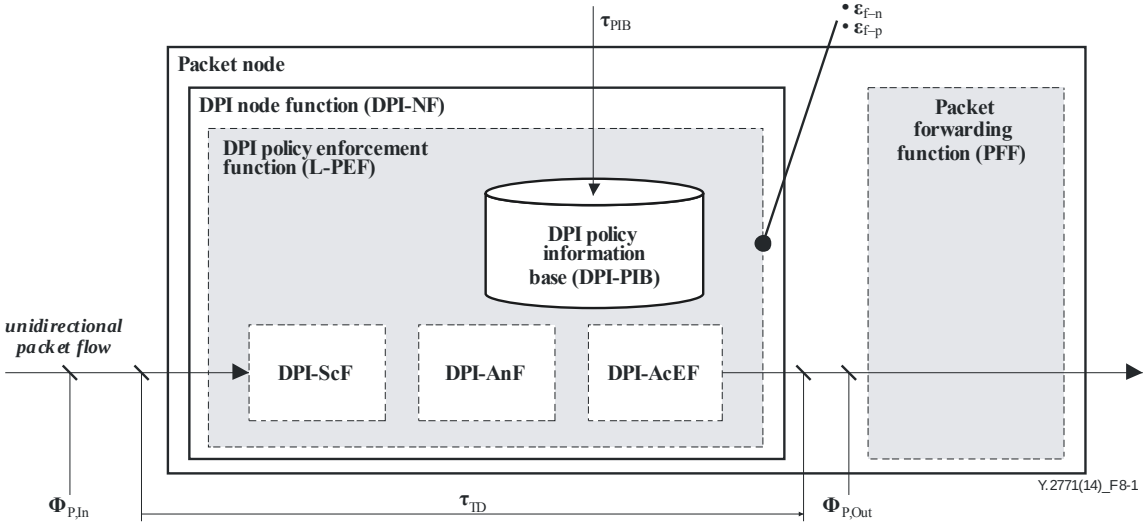
2) Performance metric should be DPI use case independent (hence, not application-specific to particular DPI services); and

3) Performance metric should be independent of specific protocols below and above the IP layer (hence, e.g., not specific to a particular IP transport protocol (such as TCP), IP application protocol, etc.); and

4) Performance metric should be independent of physical realizations of DPI entities (hence, not related to implementation specific aspects such as energy consumption, power dissipation, state-of-the-art processing components, etc.).

8.2.1.2 Typical key performance indicators for DPI nodes

Figure 8-1 illustrates some well-known example KPIs for DPI nodes (the list of KPI_{DPI} is not exhaustive). The KPI_{DPI} types are listed below the diagram. Correspondent performance requirements (if any) are outlined in separate clauses.



NOTE 1 – The packet node and PFF are shown with respect to an unambiguous specification of performance metrics, but both entities as such are out of scope of this Recommendation.
NOTE 2 – The PFF is only present for the in-path DPI mode.

Figure 8-1 – Overview – Key performance indicators for DPI nodes

The major performance metrics are typically:

- KPI "(DPI) **node-internal transfer delay**" τ_{TD} [µs]: see clause 8.2.3.1.
- KPI "(DPI) **packet processing rate**" $\phi_{P,In}[s^{-1}]$: see clause 8.2.3.2.
- KPI "(DPI) **error rate**" ε_{DPI}: see clause 8.2.3.3.
  - KPI "(DPI) false-positive error rate" $\varepsilon_{f\text{-}p}$.
  - KPI "(DPI) false-negative error rate" $\varepsilon_{f\text{-}n}$.
- KPI "(DPI) **rate of successfully identified packets**" $\phi_{P,Identified}[s^{-1}]$: see clause 8.2.3.4.

8.2.2 Formal template for performance metric definitions

Performance metric definitions in this Recommendation use the template according to Table 8-1, which itself is derived from the IETF template according to clause 5.4.4 of [b-IETF RFC 6390] "Performance Metric Definition Template".

Table 8-1 – Formal template for performance metric definitions

Metric name:	N	
Symbol:	I	
Metric description:	N	
Method of measurement or calculation:	N	
Units of measurement:	N	
Measurement point(s) with potential measurement domain:	N	
Measurement timing:	N	
Implementation:	I	
Verification:	I	
Use and applications:	I	For example, "realtime DPI", "non-realtime DPI"
Reporting model:	I	
Type "KPI": yes/no?	I	That is, "KPI", "non-KPI" or "unclassified"
NOTE – Normative (N) and informative (I) description elements.		

The template is used in order to ensure a certain minimum of specification quality for the introduced metrics by this Recommendation. However, primarily the *normative* description elements are provided due to the "framework character" of this Recommendation. Empty (*informative*) description elements are an indicator that the usage of such a metric in a real performance specification would firstly need further specification work in order to get complete, applicable metrics. For instance, item "implementation" description is outside the scope of a "framework" Recommendation, or a metric definition without item "verification" information is useless (because required e.g., for the calibration of the measurement function).

8.2.3 General performance metric definitions for DPI entities

8.2.3.1 DPI metric "node-internal transfer delay"

DPI related policy rules are applied to each individual packet of a particular packet flow. Such a kind of policy enforcement introduces basically an additional service and waiting time in the packet forwarding path of a packet node (e.g., an IP hop) with "DPI engine" support (i.e., a policy enforcement point (PEP) providing DPI). The performance metric node-internal transfer delay represents the packet transfer delay by the network element itself.

Table 8-2 provides the metric definition.

Table 8-2 – DPI metric "node-internal transfer delay"

Metric name:	N	Node-internal transfer delay
Symbol:	I	τ_{TD}
Metric description:	N	The accumulated waiting and service times of a packet through a DPI node
Method of measurement or calculation:	N	This value is calculated by measuring the entry and exit times ($T_{in,i}$ and $T_{out,i}$) of individual packets at the packet interfaces of a physical or logical representation of a DPI node function. Precondition: the measurement entity must be able to identify individual packets. Warning: this metric is typically load-dependent because the node-internal *transfer delay* is composed of node-internal *service* and *waiting times*. Load, or more precisely the DPI offered load $A_{DPI\text{-}NF}$, is given by the incoming packet rate $\phi_{P,In}$ and the mean service time per packet $T_{H,Packet}$ according to: $\boxed{A_{DPI-NF} = \phi_{P,In} \cdot T_{H,Packet}}$. Principal load dependency (see also clause 8.3): $\boxed{\tau_{TD} = f(A_{DPI-NF})}$
Units of measurement:	N	ns
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	This metric can be used over a wide range of time intervals
Implementation:	I	–
Verification:	I	–
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	"KPI"
NOTE – N: Normative description element; I: Informative description element.		

8.2.3.1.1 Further discussion

a) DPI versus non-DPI nodes:

Example of IP node: the transfer delay of a DPI node may be fundamentally greater than the transfer delay of a legacy IP node (i.e., an IP hop or router according to [b-IETF RFC 1812]) due to the additional service function on top of native IP forwarding functionality.

b) Typical relationships:

The node-internal transfer delayτ_{TD} could be dependent (because it is implementation specific) also on the following parameters:

– the number of DPI policy rules N_{db} (e.g., increased service time when multiple DPI policy rules are processed in series);

– the packet size *Sp*[bit] (e.g., increased search or comparison time as part of DPI policy condition verification), the packet size *Sp* could be related to the L2 frame size value given by [b-IETF RFC 2544]; and

– the number of DPI engines N_{DPIeng} (e.g., consideration of internal parallelism; see clause 7.3.3).

Thus, in this example, τ_{TD} would be a function of parameters N_{db}, Sp and N_{DPIeng}. i.e., $\tau_{TD} = f(N_{db}, S_P, N_{DPIeng})$.

The three parameters influence consequently the mean service time per packet $T_{H,Packet}$ (as introduced in Table 8-3): the first two parameters are increasing factors, whereas the third parameter leads to a reduction in the mean service time.

c) Qualitative performance requirement:

The transfer delay (inclusive of the additional service time due to DPI processing) should not exceed any end-to-end real-time requirements related to the overall communication service.

NOTE 1 – Such a packet forwarding capability is also colloquially known as "wire speed processing".

NOTE 2 – Such a performance objective may limit the number of enforced policy rules per packet (just due to the limited budget of service time).

8.2.3.2 DPI metric "packet processing rate"

Table 8-3 provides the metric definition.

Table 8-3 – DPI metric "packet processing rate"

Metric name:	N	Packet processing rate
Symbol:	I	$\phi_{P,In}$
Metric description:	N	The rate of packets, processed by the DPI-PEF. This is the ingress packet rate because DPI policy rules are performed for every incoming packet. The egress rate is equal or smaller than the ingress rate (due to possible packet discard actions), $\phi_{P,In} \leq \phi_{P,Out}$
Method of measurement or calculation:	N	Counting all packets observed at ingress interface p1 over a time period. The value is then calculated by dividing the observed number by the time period
Units of measurement:	N	s^{-1}
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	This metric can principally be used over a wide range of time intervals. Typically time scale is at the level of seconds
Implementation:	I	–
Verification:	I	–
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	"KPI"
NOTE – N: Normative description element; I: Informative description element.		

The DPI packet processing rate ϕ_P, is dependent on many parameters e.g., the combination of:

– the number of DPI policy rules, or DPI PIB size, *Ndb*;

– the packet size, *Sp*. The packet size *Sp* could be related to the L2 frame size values given by [b-IETF RFC 2544]; and

– possible other parameters.

Example: When $(\phi_P, N_{db}, Sp) = (200, 1000, 64)$, the processing rate is at least 200 packets/sec when the number of policy rules is less than 1000 and the packet size is 64.

The qualitative behaviour is described in clause 8.3. The upgrade of the DPI-PIB by adding new, modifying existent or deleting DPI policy rules, should not impact the nominal DPI packet processing rate ϕ_P (with ϕ_P equal to $\phi_{P,In}$).

8.2.3.3 DPI metric "error rate"

The sum of false-negative and false-positive results relates to the error rate of a DPI node. These performance metrics are only related to statistical decisions (if any at all) of a DPI node. The DPI-PEF will provide a deterministic behaviour for the majority of DPI policy rules, however, there might be DPI policy rules with statistical policy conditions or packet flows with statistical traffic information which may lead to incorrect DPI-PEF decisions.

Table 8-4 provides the metric definition.

Table 8-4 – DPI metric "error rate"

Metric name:	N	Error rate
Symbol:	I	ε_{DPI}
Metric description:	N	The sum of false-negative (see clause8.2.3.3.2) and false-positive (see clause 8.2.3.3.1) results of the DPI node
Method of measurement or calculation:	N	Direct measurement: not possible (Note 2). Indirect measurement (calculation): $\varepsilon_{DPI} = \varepsilon_{f-n} + \varepsilon_{f-p}$
Units of measurement:	N	–
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	The measurement interval is dependent on the time scale from the perspective of the served user instance (Note 3)
Implementation:	I	–
Verification:	I	–
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	"KPI"
NOTE 1 – N: Normative description element; I: Informative description element. NOTE 2 – This performance metric is a so-called *composed* metric, i.e., it may not be measured directly, but can be composed from *base* metrics that have been measured (see clause 5.3.1 of [b-IETF RFC 6390]). NOTE 3 – The served user instance in general represents a remote entity ("the user"), interested in the measurements. Examples: performance management system, DPI PD-FE.		

8.2.3.3.1 DPI metric "false-positive error rate"

Table 8-5 provides the metric definition.

Table 8-5 – DPI metric "false-positive error rate"

Metric name:	N	False-positive error rate
Symbol:	I	$\varepsilon_{f\text{-}p}$
Metric description:	N	The proportion of negative instances that were erroneously reported as being positive
Method of measurement or calculation:	N	Measurements of this metric are inherently challenging, hence, only indications may be given by this Recommendation: Typically a well-known pattern of a sufficiently large series of packets would be sent to the DPI entity. The *expected* result (as given by the applied DPI policy rules) is compared against the *measured* results from the DPI process. The measurement may be done in intrusive or non-intrusive test types
Units of measurement:	N	–
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	The measurement interval is dependent on the time scale from the perspective of the served user instance
Implementation:	I	–
Verification:	I	–
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	Yes
NOTE – N: Normative description element; I: Informative description element.		

Example 1:

A DPI policy condition C_i implies the identification of "application type *X*" and the packet identification function (DPI-PIF) concluded "application type *X*" for a packet of "application type *Y*", which is a false-positive result.

Example 2:

The calculation of this metric is possible for bloom filter-based probabilistic DPI (see Appendix I) with the following parameter values:

- m = Size of bloom filter in bits
- n = Number of signatures in set S
- k = Number of hash functions used for the generation of the bloom filter.

The false positive rate, $\varepsilon_{f\text{-}p}$, is given by equation:

$$\varepsilon_{f-p} = \left(1 - e^{-kn/m}\right)^k$$

The calculated and expected result can be then verified by measurements.

8.2.3.3.2 DPI metric "false-negative error rate"

Table 8-6 provides the metric definition.

Table 8-6 – DPI metric "false-negative error rate"

Metric name:	N	False-negative error rate
Symbol:	I	$\varepsilon_{f\text{-}n}$
Metric description:	N	The proportion of positive instances that were erroneously reported as negative
Method of measurement or calculation:	N	See correspondent entry in Table 8-5
Units of measurement:	N	-
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	The measurement interval is dependent on the time scale from the perspective of the served user instance
Implementation:	I	-
Verification:	I	-
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	Yes
NOTE – N: Normative description element; I: Informative description element.		

Example:

> A DPI policy condition C_i implies the identification of "application type *X*" and the DPI packet identification function (DPI-PIF) does not identify an "application type *X*" packet as "application type *X*", which is a false-negative result.

8.2.3.3.3 Relation to run-time errors

The DPI engine as run-time environment for the execution of DPI policy rules is inherently not error free. However, the run-time error rate and DPI metric "error rate" represent different performance indicators.

Background information:

For example, the run-time exception event according to clause 4.1 of [b-IETF RFC 4011] provides information about the concept of run-time errors:

> "[…] Run-Time Exception (RTE) – A run-time exception is a fatal error caused in language or function processing. If, during the invocation of a script, a run-time exception occurs, execution of that script is immediately terminated. If a policyCondition experiences a run-time exception while processing an element, the element is not matched by the condition and the associated action will not be run on that element. […]"

8.2.3.4 DPI metric "rate of successfully identified packets"

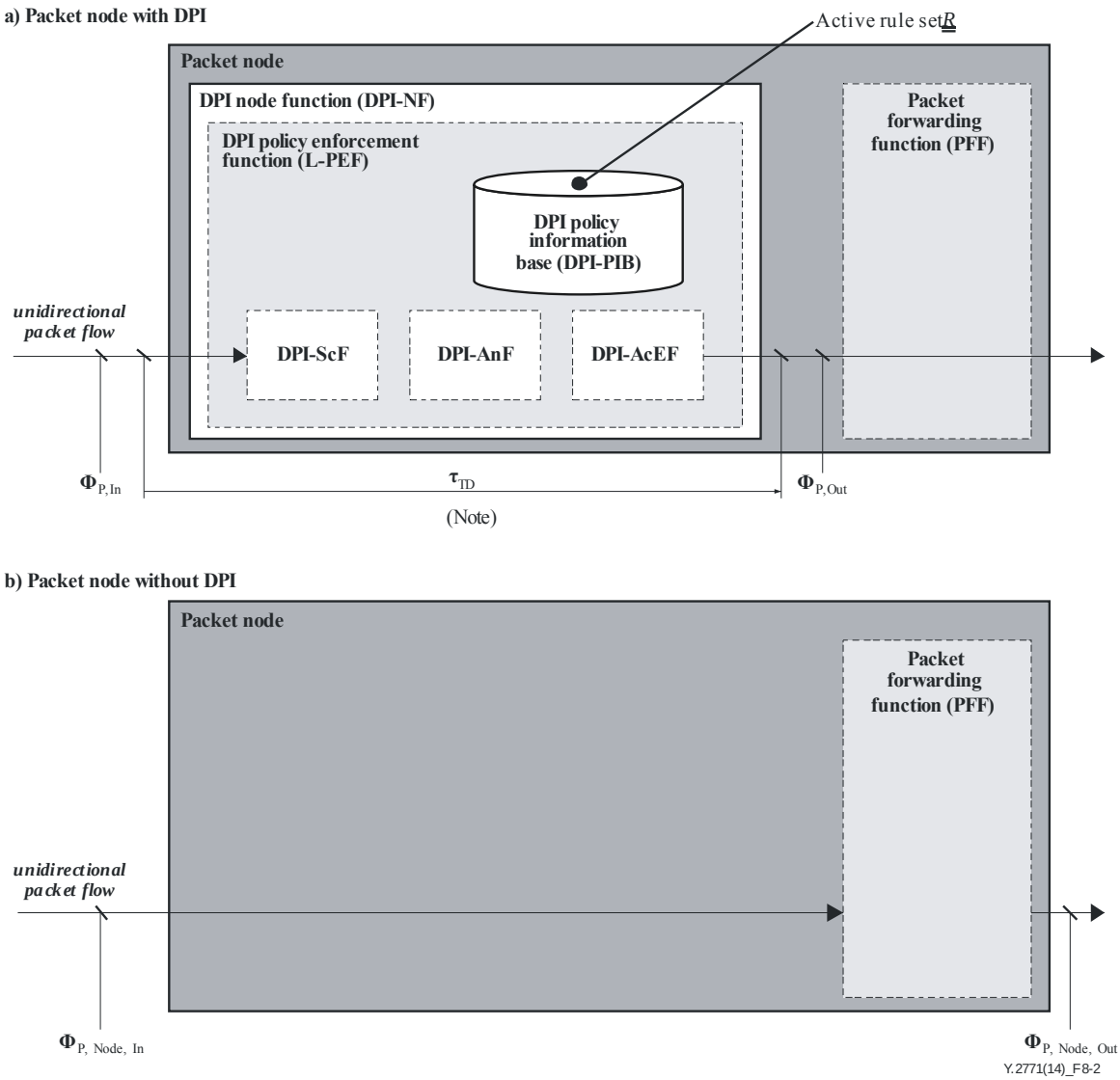
Table 8-7 provides the metric definition.

Table 8-7 – DPI metric "rate of successfully identified packets"

Metric name:	N	Rate of successfully identified packets
Symbol:	I	$\phi_{P,Identified}$
Metric description:	N	An incoming packet is "successfully identified" (by the packet identification function) when the DPI policy rule conditions (from at least one DPI policy rule) "match" the inspected packet. The type of "match" (such as full, partial, deterministic, with probability …, etc.) is not further qualified. The "rate" relates to the number of successfully identified packets per time unit
Method of measurement or calculation:	N	1. Direct measurement: For instance: enforcement of a known DPI policy rule and the generation of a packet flow with known characteristics (i.e., the ratio of traffic which should match (or not) is known in advance). The measured value is then compared against the nominal value 2. Indirect measurement (calculation): $\phi_{P,Identified} = \phi_{P,In} \cdot (1 - \varepsilon_{DPI})$
Units of measurement:	N	s^{-1}
Measurement point(s) with potential measurement domain:	N	See Figure 8-1 (traffic model)
Measurement timing:	N	The measurement interval is dependent on the time scale from the perspective of the served user instance
Implementation:	I	-
Verification:	I	See above method "direct measurement"
Use and applications:	I	"realtime DPI"
Reporting model:	I	Typically as part of performance management
Type "KPI": yes/no?	I	"KPI"
NOTE – N: Normative description element; I: Informative description element.		

8.3 Performance of policy enforcement points, estimation of qualitative performance behaviour

The purpose of this clause is to provide complementary information concerning qualitative performance estimations for protocol layer dependent policy enforcement. Figure 8-2 shows a packet node (a) with a DPI node function and (b) without any enforced DPI. The considered key performance indicator here is the packet node throughput, $\phi_{P,Node,Out}$.



NOTE – The packet node and PFF are shown with respect to an unambiguous specification of performance metrics, but both entities as such are out of scope of this Recommendation.

Figure 8-2 – Policy enforcement performance – Packet node throughput $\phi_{P,Node,Out}$ as a function of the set of enforced policy rules $\underline{\underline{R}}$ per packet

Figure 8-3 illustrates some principal throughput curves. A specific policy enforcement function (y-axis) is characterized by the number of policy rules $\underline{\underline{R}}$ per packet as well as aspects of rules interaction. Performing a particular policy rule consumes a certain amount of packet path resources in terms of processing time, packet memory, TCAM/CAM memory, policy database, etc.

Simplified rule of thumb: the more rules per packet, the more resources required for policy enforcement.

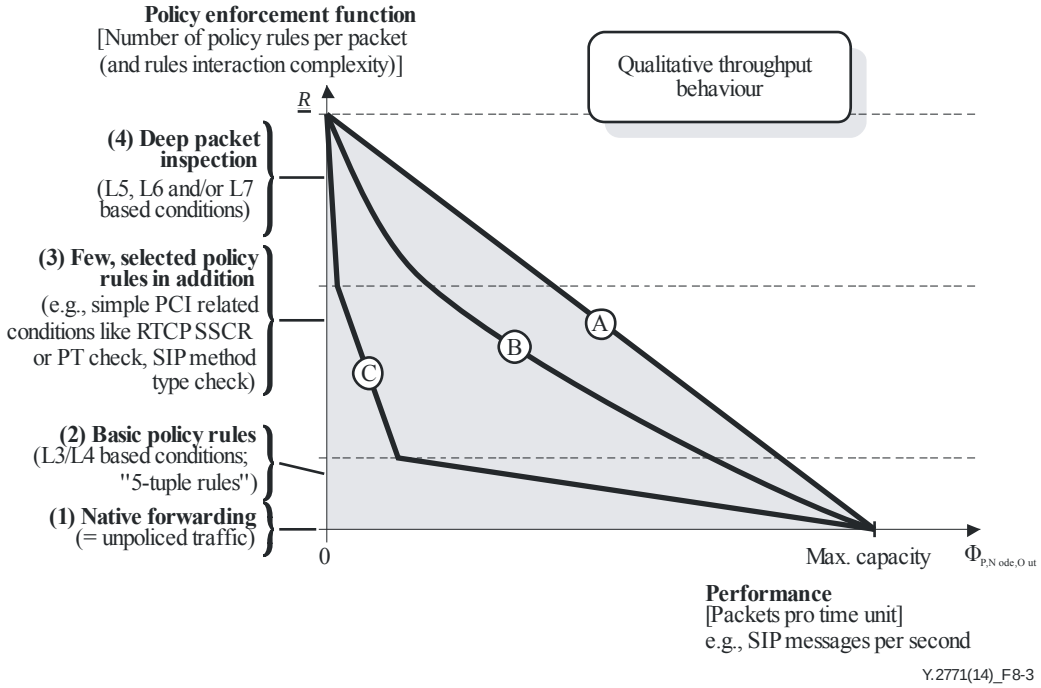


Figure 8-3 – Policy enforcement performance – Qualitative throughput behaviour

An ideal implementation may achieve a "linear" behaviour like *A*. A more realistic and cost effective model would rather follow curve *C*.

The technical (and commercial) challenge with behaviour *C* is the fairly non-linear relationship, see Figure 8-4, which makes it not trivial to engineer a nominal load point or/and to achieve the required trade-off for limiting the set of enforced policy rules.

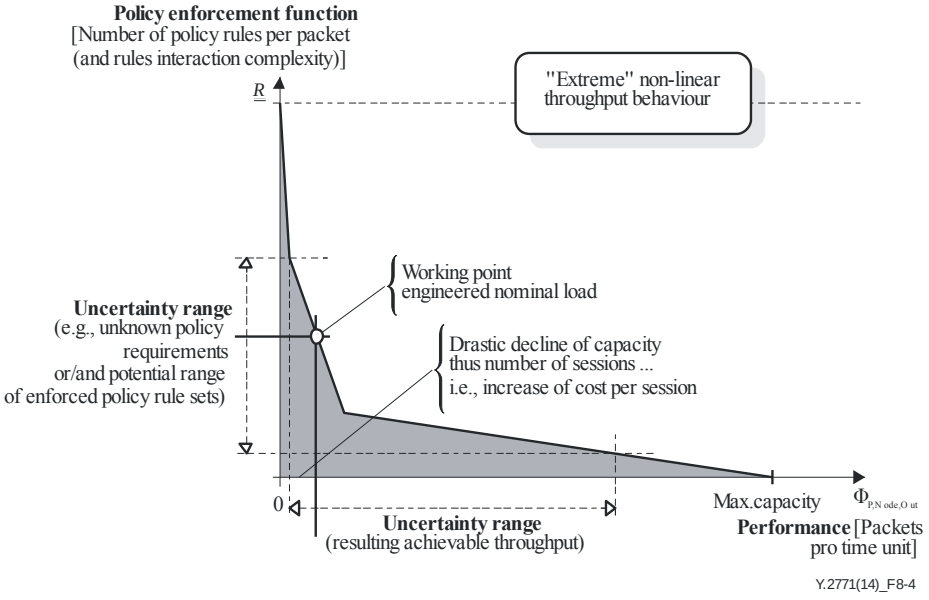


Figure 8-4 – Policy enforcement performance – Example of use case *C* as a worst-case scenario

9 Categorization of DPI functional entities

A DPI entity is typically not supporting the entire set of DPI requirements according to [ITU-T Y.2770], rather just a subset given by the aimed use cases. There might be consequently different types of DPI-FEs identified.

9.1 Categorization principles

Every identifiable capability of a DPI entity could be basically associated with DPI requirement(s) as specified by [ITU-T Y.2770]. There are at high level:

- capabilities in terms of conditions processing;
- capabilities in terms of actions processing; and
- possibly other capabilities.

Typical criteria for the identification of particular DPI-FE types are deployment scenarios (DPI use case), complexity of processing logic, cost factors, etc.

9.2 Capabilities in terms of conditions processing

Capabilities of conditions processing can be divided into two classes: L4 PI (L4 payload inspection enabled) and non-L4 PI (payload inspection disabled).

9.3 Capabilities in terms of actions processing

See clause 6.3.3.1 in [ITU-T Y.2770] concerning hierarchical levels of actions and examples.

9.4 DPI-FE types

Table 9-1 uses the classification principles from clauses 9.2 and 9.3 and provides an overview with regard to three relevant types:

Table 9-1 – Categories of DPI-FE types

Type of DPI-FE			*Capabilities in terms of **actions** processing*	
			Support of DPI-AcEF:	
			No	Yes
*Capabilities in terms of **conditions** processing*	Support of L4 Payload Inspection:	No	Type 1	
		Yes	Type 2	Type 3

According to the DPI functional capabilities of a DPI-FE, the DPI-FE can be categorized as follows (Table 9-2):

Table 9-2 – The three types in detail

Type	Rule processing
1	A FE without the capability of L4 payload inspection ($L_4PI = L_{4+}HI \cup L_7PI$), i.e., a type of non-DPI-FE (e.g., a SPI-FE)
2	DPI-FE without the capability of action execution (DPI-AcFE), but with L4 payload inspection ($L_4PI = L_{4+}HI \cup L_7PI$)
3	DPI-FE with the capability of action execution (DPI-AcFE) plus L4 payload inspection ($L_4PI = L_{4+}HI \cup L_7PI$)

The type of a DPI FE could be the result of factors such as:

1) amount of available resources: capabilities of a specific DPI physical entity (DPI-PE) (like hardware (HW) or software (SW) components) or
2) amount of allocated/enabled resources for DPI processing: via configuration management (e.g., via policy management entities (see clause 7) by provisioning a dedicated capability set).

Note that a particular type *n* DIP FE can be configured as a type *m* (n > m) by its management entity (due to the fact that e.g., the "type 3 DPI capability set" is a superset with regard to the other types).

The DPI FE should be able to report its type to related functional entities (e.g., RACF).

According to the DPI functional capabilities of a DPI-FE, the type 3 DPI-FE may be further subdivided as follows (Table 9-3):

Table 9-3 – Sub-variants of type 3

Type	Rule processing
3.1	DPI-FE with the capability of information gathering and reporting
3.2	Type 3.1 plus the ability of traffic control, but without the capability of modifying packet contents
3.3	Type 3.2 plus the capability of modifying packet contents

10 Security considerations

Regulation, privacy and security application aspects of DPI are outside the scope of this Recommendation. Vendors, operators and service providers are required to take into account national regulatory and policy requirements when implementing this Recommendation.

According to [ITU-Y.2770], the DPI-FE and the information pertaining to DPI operations should be under protection against threats. The mechanisms specified in [ITU T Y.2704] address the security requirements of [ITU-T Y.2770].

Appendix I

Example functional architecture of probabilistic DPI based on the bloom filter

(This appendix does not form an integral part of this Recommendation.)

I.1 Introduction

Bloom filter is described in [b-Bloomfilter]:

Bloom filters use a randomized technique to test membership queries on a set of strings. Given a string, the Bloom filter computes *k* hash functions on it producing hash values ranging from 1 to *m* (see Figure I.1). It then sets *k* bits in a m-bit long vector at the addresses corresponding to the *k* hash values, where *k* is not greater than *m* (see also Equation 7-1). The same procedure is repeated for all the members of the set. This process is called "programming" of the filter. The query process is similar to programming, where a string whose membership is to be verified is input to the filter. The Bloom filter generates *k* hash values using the same hash functions it used to program the filter. The bits in the m-bit long vector at the locations corresponding to the *k* hash values are looked up. If at least one of these bits is found not set then the string is declared to be a non-member of the set. If all the bits are found to be set then the string is said to belong to the set with a certain probability. This uncertainty in the membership comes from the fact that those *k* bits in the m-bit vector can be set by any of the members. Thus finding a bit set does not necessarily imply that it was set by the particular string being queried. However, finding a bit not set certainly implies that the string does not belong to the set, since if it did then all the *k* bits would definitely have been set when the Bloom filter was programmed with that string. This explains the presence of false positives in this scheme, and the absence of any false negatives.

For example, in Figure I.1, the bloom filter BF (B[0..m-1]) is generated by 3 hash functions, h1, h2 and h3, on string x1 and x2, where in the bloom filter-based DPI strings x1 and x2 are the DPI signatures. String y1 and y2 are verified by 3 hash functions h1, h2 and h3 on string y1 and y2 against the bloom filter BF (given by bit vector B[0..m-1]), where in the bloom filter based DPI, strings y1 and y2 represent the inspected data structures, given e.g., by the payload of incoming packets.

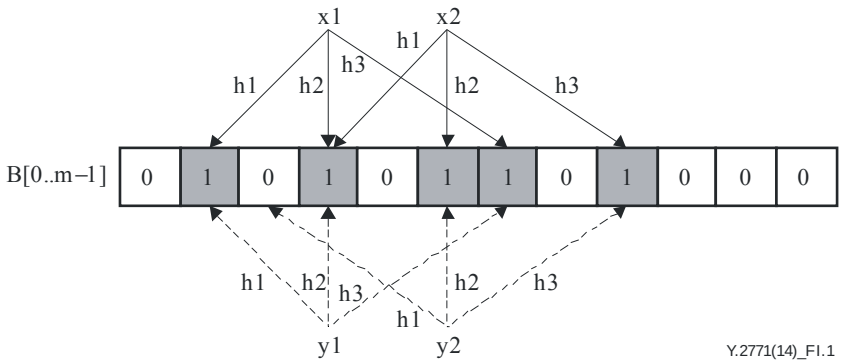


Figure I.1 – Programming and query of bloom filter (BF equal to bit vector B[0..m-1])

The false positive rate, $\varepsilon_{f\text{-}p}$, is expressed by Equation I-1:

$$\varepsilon_{f-p} = \left(1 - e^{-kn/m}\right)^k \tag{I-1}$$

where, *n* is the number of strings programmed into the bloom filter. The value of $\varepsilon_{f\text{-}p}$ can be reduced by choosing appropriate values of *m* and *k* for a given size of the member set, *n*.

I.2 Functional model of bloom filter based probabilistic DPI

The functional model of bloom filter based probabilistic DPI is illustrated in Figure I.2. The policy rule for probabilistic DPI may be:

If packet 'P' contains signatures from a DPI signature set *'S'* (as policy condition), where the signature set is given by *'S'* = {*'S1'*, *'S2'*,..., *'Sm'*}, then discard the packet (as policy action).

The bloom filter BF_S for signature set S is generated by the set of hash functions H_1, H_2, …, H_k. The policy rule is converted to:

If H_1('P'), H_2('P'),…, H_k('P') match BF_S, then discard.

Before the DPI analyser compares the arriving packet against this DPI policy rule condition, the DPI scanner needs to determine the offset and length in the arriving packet which is used to match against the DPI policy rule conditions.

There are two principle options:

1) For the protocol-stack aware DPI rule condition, the offset and signature length in the set 'S' is known, and the scanner reports the offset and length information to the DPI analyser directly.

2) For the protocol-agnostic DPI rule condition, the DPI scanner needs to scan and determine the offset and length. The DPI scanner reports that information to the DPI analyser.

The DPI analyser generates the hash result of packet P using H_1, H_2, …, H_k, matches the results with the BF_S and reports the match result to the DPI action execution function. The DPI action execution function reports the matching result ("True" or "False") and discards the packet when the match result is estimated to be True, or else passes the packet to the packet forwarding function. If the generated match result does not match the BF_s, the DPI scanner needs to scan from the 'offset+1' byte (offset = offset + 1), and the process will continue until the end of the packet.

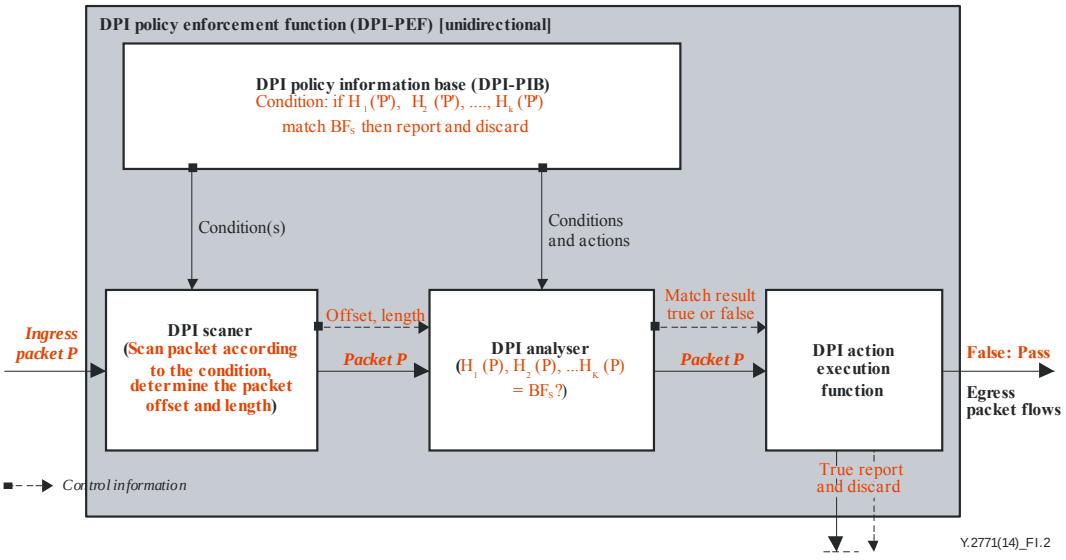


Figure I.2 – Functional model of bloom filter based probabilistic DPI

It may be noted that the match result of the DPI analyser is of type Boolean, i.e., True or False, hence not any probability value such as a "*positive match with probability* ***p*** (and p between 0% and 100%)". However, the entire DPI packet processing path stages as such are leading to probabilistic DPI results due to the inherent false positive rate $\varepsilon_{f\text{-}p}$, as part of the DPI policy condition.

Bibliography

[b-ITU-T H.248.53] Recommendation ITU-T H.248.53 (2009), *Gateway control protocol: Traffic management packages.*

[b-ITU-T I.130] Recommendation ITU-T I.130 (1988), *Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN.*

[b-ITU-T J.380.1] Recommendation ITU-T J.380.1 (2011), *Digital program insertion – Advertising systems interfaces – Advertising systems overview.*

[b-ITU-T X.1036] Recommendation ITU-T X.1036 (2007), *Framework for creation, storage, distribution and enforcement of policies for network security.*

[b-ITU-T Y.1221] Recommendation ITU-T Y.1221 (2010), *Traffic control and congestion control in IP-based networks.*

[b-ITU-T Y.2121] Recommendation ITU-T Y.2121 (2008), *Requirements for the support of flow-state-aware transport technology in NGN.*

[b-ITU-T Y-Sup.23] ITU-T Y-series Recommendations – Supplement 23 (2013), *ITU-T Y.2770-series - Supplement on DPI terminology.*

[b-IETF RFC 1812] IETF RFC 1812 (1995), *Requirements for IP Version 4 Routers*.

[b-IETF RFC 2544] IETF RFC 2544 (1999), *Benchmarking Methodology for Network Interconnect Devices.*

[b-IETF RFC 3060] IETF RFC 3060 (2001), *Policy Core Information Model – Version 1 Specification.*

[b-IETF RFC 3198] IETF RFC 3198 (2001), *Terminology for Policy-Based Management.*

[b-IETF RFC 4011] IETF RFC 4011 (2005), *Policy Based Management MIB.*

[b-IETF RFC 4292] IETF RFC 4292 (2006), *IP Forwarding Table MIB.*

[b-IETF RFC 6390] IETF RFC 6390 (2011), *Guidelines for Considering New Performance Metric Development.*

[b-Bloomfilter] Dharmapurikar, S. et al., (2003), *Implementation of a Deep Packet Inspection Circuit using Parallel Bloom Filters in Reconfigurable Hardware.* IEEE Proceedings of 11th Symposium on High Performance Interconnects. Stanford University, Wiley, John & Sons, Inc.

[b-CRTC] Canadian Radio-Television and Telecommunications Commission(2009), *ISP Traffic Management Technologies: The State of the Art*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects and next-generation networks**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2014