World Telecommunication Standardization Assembly (WTSA-12)
Dubai, 20 November - 29 November 2012

PLENARY MEETING

Document 30-E
July 2012
Original: English

ITU-T Study Group 13
Future networks including mobile and NGN

DRAFT NEW RECOMMENDATION ITU-T Y.2770 PROPOSED FOR APPROVAL AT THE WORLD TELECOMMUNICATION STANDARDIZATION CONFERENCE (WTSA-12)

Contact: TSB
Tel: +41 22 730 5126
Fax: +41 22 730 5853
Email: tsbsg13@itu.int

ADD SG13/30/1

DRAFT NEW RECOMMENDATION ITU-T Y.2770 (FORMERLY Y.DPIREQ)

Requirements for Deep Packet Inspection in Next Generation Networks

Summary

This Recommendation specifies the requirements for Deep Packet Inspection (DPI) in Next Generation Networks (NGN). This Recommendation primarily specifies the requirements for Deep Packet Inspection (DPI) entities in NGN, addressing, in particular, aspects such as application identification, flow identification, inspected traffic types, signature management, reporting to the network management system (NMS) and interaction with the policy decision functional entity. Although aimed at the NGN, the requirements may be applicable to other types of networks. This Recommendation also contains use cases and other complementary information as appendixes.

CONTENTS

DRAFT NEW RECOMMENDATION ITU-T Y.2770 (FORMERLY Y.DPIREQ)

Requirements for Deep Packet Inspection in Next Generation Networks

1 Scope

This Recommendation primarily specifies the requirements for Deep Packet Inspection (DPI) entities in NGN, addressing, in particular, aspects such as application identification, flow identification, inspected traffic types, signature management, reporting to the network management system (NMS) and interaction with the policy decision functional entity.

This Recommendation also identifies the requirements for DPI of traffic in non-native encoding formats (e.g., encrypted traffic, compressed data, and transcoded information).

Any DPI function may be generally described by the concept of policy rules (see clause 1.2). DPI application scenarios and complementary information such as example policy rules for packet identification, policy enforcement process, policy specification languages, DPI in layered protocol architectures, and definition of terminology are given in Appendixes.

Implementers and users of the described techniques shall comply with all applicable national and regional laws, regulations and policies.

The Recommendation does not address the specific impact of implementing a distributed DPI functionality. The requirements are primarily about functional aspects of DPI, but physical aspects are also covered. In the context of functional to physical mapping scenarios, only 1-to-1 mapping and N-to-1 mapping between a DPI-FE and a DPI-PE is in scope of this Recommendation. In other words, no requirements cover distributed DPI-PEs.

1.1 Applicability

The Recommendation is applicable to the scenarios identified in Figure 1-1:

		Packet-based network type	
		NGN	non-NGN
Packet bearer technology	IP	Applicable	Possibly applicable
	non-IP	Possibly applicable	Possibly applicable

Y.2770(12)_F1-1

Figure 1-1 – Applicability of this Recommendation

The notion of "non-IP" refers to protocol stacks for packet bearer types without any IP protocol layer ([IETF RFC 791] and [IETF RFC 2460]).

Though this recommendation mainly addresses the requirements of DPI for NGN, these requirements may be applicable to other types of networks. This further applicability is for further study.

1.2 Policy Rules

This Recommendation assumes a generic high-level format for all policy rules. This high level format applies to DPI rules as shown in Figure 1-2, as well as non-DPI (e.g. shallow packet

inspection, as mentioned in appendix III.3.1 which are not specifically described in this Recommendation). The format distinguishes three basic blocks of

i) **rules identifier/name** (with ranking/order indication due to possible multiple rules);

ii) **DPI signature/condition**s;

iii) **action**s.

There is a logical binding between action(s) and condition(s), see clause 3.1.2.

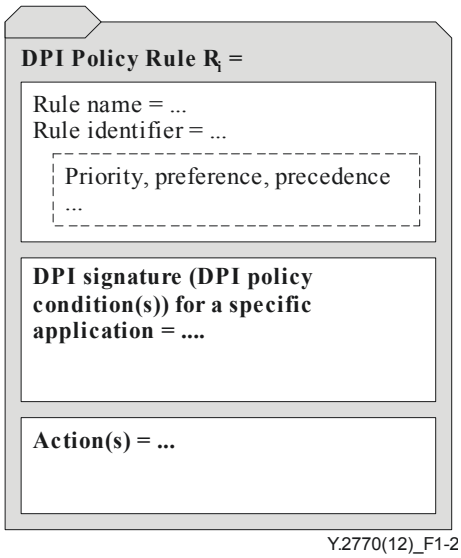


Figure 1-2 – Generic format of DPI Policy Rules

Note that the following aspects are in scope:

- The specification of requirements related to the DPI signature, (i.e., the DPI signatures used for application identification and flow identification);
- The specification of requirements related to the identification and naming of DPI policy rules; and
- The identification of possible scenarios involving policy actions as potential follow-up activities after the evaluation of DPI signatures.

In contrast, the following aspects are out of scope:

- The specifications of requirements related to actions concerning the modification of inspected packet(s);
- The specification of explicit bindings between actions and conditions (NOTE);
- The specification of DPI policy rules in full;
- The specification of a language for DPI signatures; and
- The specifications of concrete DPI policy conditions (such as behavioural or statistical functions).

NOTE – For instance, there might specification of the action of discarding a packet, and the condition of searching for a packet signature, but there will *not* be any specification that associates an individual action to an actual condition.

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.

[IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol*.

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

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

[ITU-T E.107] Recommendation ITU-T E.107 (2007), *Emergency Telecommunications Service (ETS) and interconnection framework for national implementations of ETS*.

[ITU-T X.200] Recommendation ITU-T X.200 (1994) | ISO/IEC 7498-1:1994, *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.1221] Recommendation ITU-T X.1221 (2010), *Traffic control and congestion control in IP based networks*.

[ITU-T Y.2111] Recommendation ITU-T Y.2111 (2008), *Resource and admission control functions in Next Generation Networks*.

[ITU-T Y.2205] Recommendation ITU-T Y. 2205 (2011), *Next Generation Networks - Emergency telecommunications - Technical considerations*.

[ITU-T Y.2701] Recommendation ITU-T Y.2701 (2007), *Security requirements for NGN release 1*.

[ITU-T Y.2704] Recommendation ITU-T Y.2704 (2007), *Security mechanisms and procedures for NGN*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 filter [b-IETF RFC 3198]: A set of terms and/or criteria used for the purpose of separating or categorizing. This is accomplished via single- or multi-field matching of traffic header and/or payload data. "Filters" are often manipulated and used in network operation and policy. For example, packet filters specify the criteria for matching a pattern (for example, IP or 802 criteria) to distinguish separable classes of traffic.

NOTE – In this Recommendation, the term “traffic header” is equivalent to “packet header”.

3.1.2 filter/policy rule [b-IETF RFC 3198]: A basic building block of a policy-based system. It is the binding of a set of actions to a set of conditions, where the conditions are evaluated to determine whether the actions are performed.

NOTE – In this Recommendation, a filter rule is a specific policy rule with the purpose of separating traffic, e.g., in the main categories of "accepted" and "not-accepted".

3.1.3 **flow** [IETF RFC 5101]: A set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties. Each property is defined as the result of applying a function to the values of:

1) one or more packet header fields (e.g., destination IP address), transport header fields (e.g., destination port number), or application header fields (e.g., RTP header fields [b-IETF RFC 3550]).
2) one or more characteristics of the packet itself (e.g., number of MPLS labels, etc.).
3) one or more of fields derived from packet treatment (e.g., next hop IP address, the output interface).

A packet is defined as belonging to a flow if it completely satisfies all the defined properties of the flow.

This definition covers the range from a flow containing all packets observed at a network interface to a flow consisting of just a single packet between two applications. It includes packets selected by a sampling mechanism.

NOTE – The above numbered listed items indicate flow properties in the categories of (1) "Protocol Control Information (PCI) of packets", (2) "Protocol Data Unit (PDU) properties of packets" and (3) "Local packet forwarding information".

3.1.4 **policy** [b-IETF RFC 3198]: A set of rules to administer, manage, and control access to network resources.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 **application:** A designation of one of the following

- *An application protocol type* (e.g., IP application protocols H.264 video, or 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), as a common characteristic of all above listed levels of applications.

3.2.2 **application-descriptor (also known as application-level conditions):** A set of rule conditions that identifies the application (according to clause 3.2.1).

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

3.2.3 **application tag:** A unique name for an application which is used to indicate the application semantics and is typically used for reporting scenarios.

Figure 3-1 outlines the relationship between application tag and application descriptor.

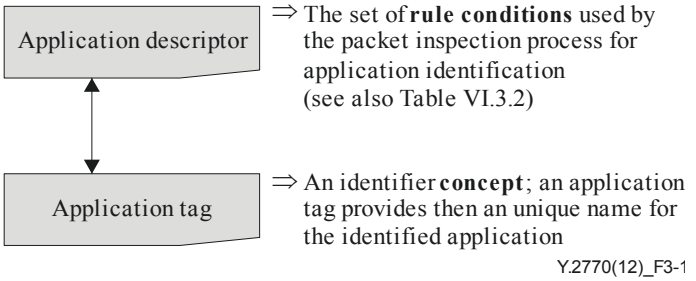


Figure 3-1 – Relationship between Application Tag and Application Descriptor

3.2.4 bidirectional DPI: DPI that involves policy conditions concerning both traffic directions.

NOTE – See also the formal description of the Application Descriptor in clause VI.3.2: the policy conditions relates to the simple conditions. There is at least one simple condition per traffic direction in the case of bidirectional DPI.

3.2.5 deep packet inspection (DPI): 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 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.

3.2.6 DPI engine: 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).

3.2.7 DPI entity: The DPI entity is either the DPI functional entity or the DPI physical entity.

3.2.8 DPI functional entity (DPI-FE): A functional entity that performs deep packet inspection.

3.2.9 DPI physical entity (DPI-PE): The implemented instance of a DPI functional entity.

3.2.10 DPI policy: A policy as defined, for example in [b-IETF RFC 3198] (see clause 3.1.4), enforced in a DPI entity.

3.2.11 DPI policy condition (also known as DPI signature): A representation of the necessary state and/or prerequisites that identifies 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 (see also Tables VI.3.1 and VI.3.2).

3.2.12 DPI policy decision functional entity (DPI-PDFE): 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.

3.2.13 DPI policy rule: The policy rule pertinent to DPI (See also clause 3.1.2). In this Recommendation, a DPI policy rule is referred to simply as a rule.

3.2.14 DPI signature: A synonym to DPI policy condition(s) (see clause 3.2.11).

3.2.15 DPI signature library: A database consisting of a set of DPI signatures. It is also called DPI protocol library because the signatures may be typically used for protocol identification.

3.2.16 flow descriptor (also known as flow level conditions): Set of rule conditions that is used to identify a specific type of flow (according clause 3.1.3) 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.

NOTE 2 – For further normative discussion of the flow descriptor as used in this Recommendation, see Annex A.

3.2.17 IPFIX flow identifier (IPFIX flow ID): The set of values for the IPFIX flow keys, which is used in conjunction with the flow descriptor to identify a specific flow.

3.2.18 IPFIX flow key: Each of the information elements of the flow descriptor that is used in IPFIX-based flow identification processes (according to [IETF RFC 5101]).

NOTE – The IPFIX flow key definition is semantically consistent with the flow key definition specified in IPFIX [IETF RFC5101]. The only difference between the two terms is that the definition in this document is scoped to the flow descriptor.

3.2.19 L3,4 Header Inspection ($L_{3,4}$HI)**:** Processing of policy rule(s) with policy conditions involving only the protocol control information (PCI) elements of the network layer or/and transport layer.

3.2.20 L4+ Header Inspection (L_{4+}HI)**:** Processing of policy rule(s) with policy conditions involving only the PCI elements above the transport layer.

3.2.21 L4 Payload Inspection (L_4PI)**:** Processing of policy rule(s) with policy conditions involving only the transport payload which may be the "application data" for particular application protocols (e.g., SIP).

NOTE – L_4PI relates to the union of L_{4+}HI and L_7PI policy conditions.

3.2.22 L7 Payload Inspection (L_7PI)**:** Processing of policy rule(s) with policy conditions based on the application data.

3.2.23 payload: The data unit following the header elements in a packet, and excluding optional elements at the end of a packet (e.g., padding, trailer, checksum elements).

NOTE 1 – Thus, the notion of payload is synonym to Service Data Unit (SDU) in the OSI-BRM [ITU-T X.200], packet is synonymous to Protocol Data Unit (PDU), and Protocol Control Information (PCI) covers all packet header and trailer elements. In summary, "PDU = PCI + SDU".

NOTE 2 – The notion of payload is specific to a particular protocol layer (i.e., Lx-Payload refers to the payload at protocol layer x). Ditto for Lx-SDU, Lx-PDU and Lx-PCI.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

ABNF Augmented Backus–Naur Form
AD Application Descriptor
AH Authentication Header
ASIC Application-specific Integrated Circuit
AVP Attribute Value Pair
BRM Basic Reference Model
CLI Command Line Interface
CPE Customer Premises Equipment
CPU Central Processor Unit
DA Destination Address
DCCP Datagram Congestion Control Protocol
DPI Deep Packet Inspection
DPI-FE DPI Functional Entity
DPI-PDFE DPI Policy Decision Functional Entity
DPI-PE DPI Physical Entity
DPI-PIB DPI Policy Information Base
DS Differentiated Services
ECN Explicit Congestion Notification
ERM Extended Reference Model
ESP Encapsulating Security Payload
ET Emergency Telecommunications
FD Flow Descriptor
F_D Flow dependent
F_I Flow independent
FPGA Field Programmable Gate Array
FTP File Transfer Protocol
FPA Full Payload area Analysis
FSL Filter Specification Language
GPRS General Packet Radio Service
GRE Generic Routing Encapsulation
HTTP Hypertext Transfer Protocol
IANA Internet Assigned Numbers Authority

IDS	Intrusion Detection System
IS	In-Service
IE	Information Elements
IP	Internet Protocol
IPFIX	IP Flow Information Export
L-PDF	Local PDF
NMS	Network Management System
MIME	Multipurpose Internet Mail Extensions
MMI	Man-Machine Interface
MP3	MPEG-1 or MPEG-2 Audio Layer III
MPEG	Moving Picture Experts Group
MPI	Medium depth Packet Inspection
MPLS	Multi Protocol Label Switching
MSRP	Message Session Relay Protocol
NAT	Network Address Translation
NGN	Next Generation Network
OoS	Out-of-Service
P2P	Peer to Peer
PCC	Policy and Charging Control
PCI	Protocol Control Information
PD	Packet Descriptor
PDF	Policy Decision Function
PEL	Policy Expression Language
PDU	Protocol Data Unit
PFF	Packet Forwarding Function
PI	Packet identification
PIB	Policy Information Base
PPA	Payload area Analysis
PSAMP	Packet Sampling
PSL	Policy Specification Language
QoE	Quality of Experience
QoS	Quality of Service
RACF	Resource and Admission Control Functions
RACS	Resource and Admission Control Subsystem
R-PDF	Remote PDF (i.e., PDF remotely located from DPI node perspective)
RTP	Real-time Transport Protocol
SA	Source Address (IP) Security Association (IPsec)
SCTP	Stream Control Transmission Protocol

SD	Session Descriptor
SDU	Service Data Unit
SigComp	Signalling Compression
SIP	Session Initiation Protocol
SLA	Service Level Agreement
SMTP	Simple Mail Transfer Protocol
SP	Service Provider
SPI	Shallow Packet Inspection (DPI) Security Parameter Index (IPsec)
TC	Traffic Class (IPv6)
TCP	Transmission Control Protocol
THIG	Topology Hiding
TISPAN	Telecommunication and Internet Converged Services and Protocols for Advanced Networking
ToS	Type of Service (IPv4)
TRM	Tunnelled Reference Model
UDP	User Datagram Protocol
VPN	Virtual Private Network
ZIP	Denotes a file format with compressed data (e.g., according [b-IETF RFC 1950])

5 Conventions

This document provides a list of items, labelled as ***R-x/y***, where *x* refers to the clause number and *y* a number within that clause. Such items use the following keywords with meanings as prescribed below:

The keywords "**is required to**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "**is prohibited from**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "**is recommended**" indicate a requirement which is recommended but which is not absolutely required. Thus this requirement need not be present to claim conformance.

The keywords "**can optionally**" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

In the body of this document and its annexes, the words shall, shall not, should, and may sometimes appear, in which case they are to be interpreted, respectively, as is required to, is prohibited from, is recommended, and can optionally. The appearance of such phrases or keywords in an appendix or in material explicitly marked as informative are to be interpreted as having no normative intent.

6 DPI functional entity requirements

6.1 Flow and application identification

R-6.1/1: The DPI Functional Entity is required to perform application identification.

R-6.1/2: The DPI Functional Entity is required to support various kinds of DPI policy rules.

R-6.1/3: The DPI-FE is required to identify an application by inspecting the application payload.

R-6.1/4: The DPI application level conditions (and optional flow level conditions) is required to allow application identification based on unidirectional traffic (unidirectional DPI) for all unidirectional applications and for bidirectional applications under the condition that one traffic direction allows an unambiguous identification.

R-6.1/5: The DPI application level conditions and (optional flow level conditions) can optionally allow application identification based on bidirectional traffic (bidirectional DPI).

R-6.1/6: The information element(s) used in the flow level conditions are recommended to comply with [b-IETF RFC 5102]**,** as registered with IANA [b-IETF IANA IPFIX].In such a case IEs are recommended to include IPFIX information elements related to the link (L2), network (L3) and transport (L4) protocol layers, following the basic IETF layered protocol architecture.

NOTE – The IANA registry for IPFIX information elements can optionally be augmented to include additional elements (by the IETF). The present IANA registry (as of the end of year 2011) is missing information elements for L4 protocols other than UDP and TCP (e.g., for SCTP and DCCP).

R-6.1/7: The information element(s) can optionally be other L2, L3 or L4 related information elements outside the IPFIX registry (called enterprise specific information elements in the IPFIX protocol [IETF RFC5101]).

DPI signature management

This clause defines requirements concerning operations on the DPI Signature Library. Such operations may be locally initiated by the DPI-FE, or by a remote network entity (see Figure 6-1). All possible types of remote network entities may be abstracted as the DPI policy decision functional entity that decides the signature-based rules to be enforced in the DPI-FE.

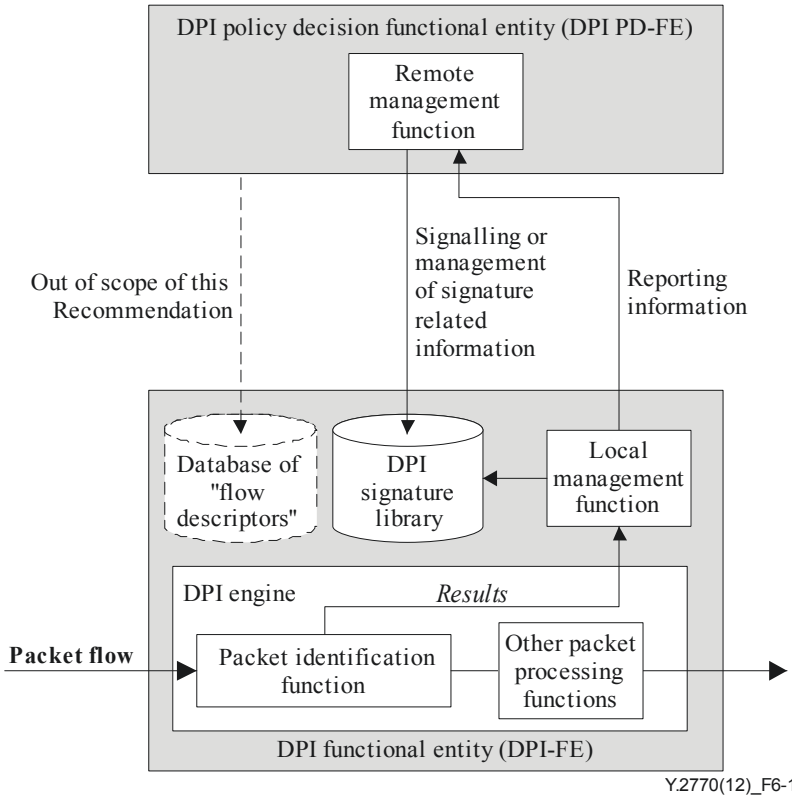


Figure 6-1 – DPI signature management in scope of an example DPI functional entity architecture (see also Figure 8-2 with regards to the internal interfaces)

The DPI Policy Decision functional entity would be associated with the RACF (in case of an NGN with a RACF), but its specification is out of scope for this Recommendation. It is included in Figure 6.1 because it contains the remote management functions for the DPI-FE.

6.2.1 General signature requirements

R-6.2.1/1: DPI signatures are required to be stored in the *DPI signature library* which is a sub-entity of the DPI-FE.

NOTE – The rationale behind a local DPI signature library is the fact that the packet identification function requires immediate access on the database content.

The DPI signature may be used for

- approximate identification (e.g., behavioural, heuristics, etc.) and
- exact identification (e.g., exact matching rules).

The language (formal or behavioural) used for specifying DPI policy rules in this library, as well as the matching rules themselves, are outside the scope of this Recommendation. This Recommendation only specifies that the DPI signature library exist, what DPI signature (s) is/are, and the library management functions.

R-6.2.1/2: The DPI signatures library is required to be securely maintained and not visible to unauthorized users.

6.2.2 Management of DPI signature library

This clause defines requirements for management of DPI signatures library.

6.2.2.1 Adding new signatures

R-6.2.2.1/1: It is required to be able to add new DPI signatures to the DPI signature library.

6.2.2.2 Operations on existing signatures

R-6.2.2.2/1: It is required to be able to modify (update) existing signatures in the DPI signature library.

R-6.2.2.2/2: It is required to be able to enable and disable specific DPI signatures in the DPI signature library.

R-6.2.2.2/3: It is required to be able to delete (remove) specific DPI signatures in the DPI signature library.

6.2.2.3 The rule format exchanged through external interface

R-6.2.2.3/1: The DPI signature for application identification exchanged through external interfaces (i.e., *e1* and *e2* in Figure 8-1) can optionally follow any rule format (see also clause 1.2).

6.2.3 Location of management function

R-6.2.3/1: The DPI signature management actions specified in clause 6.2.2 are required to be performed locally from the DPI functional entity or remotely or both (see Figure 6-1).

6.2.4 Initiation of management actions

R-6.2.4/1: It is required to support the push mode regarding DPI signature operations, when the operations are remotely initiated (e.g., by the DPI-PDFE in Figure 6-1).

R-6.2.4/2: It is required to support the pull mode regarding the DPI signature operations, when the operations are locally initiated by the DPI-FE. The notion of pull means that the DPI-FE local management function requests the DPI-PDFE to perform a management action on a new or existing signature.

How a DPI-FE could initiate a request is out of scope of this Recommendation.

6.3 Traffic inspection aspects

This clause addresses the aspects concerning the types of traffic subject to DPI.

6.3.1 Flow identification aspects

R-6.3.1/1: The DPI Functional Entity is recommended to support the identification of applications, without flow level inspection (see also Figure VII-7).

R-6.3.1/2: Any DPI scenario can optionally be initially flow-independent, i.e. the provided DPI policy rule to the DPI-FE would not contain a flow descriptor. However, the rule could request to collect interested flow information.

R-6.3.1/3: Such a request is required to provide an IPFIX flow key plus the optional completion of lacking flow information.

R-6.3.1/4: DPI functional entity can optionally require a complete recognition of IPFIX flow identifier based on a given IPFIX flow key and the inspection of multiple subsequent packets.

R-6.3.1/5: The reporting action of a complete or incomplete IPFIX flow identifier by the DPI-FE to a remote network entity can optionally be conditional (e.g., event-driven, timer-controlled, etc.).

6.3.2 Protocol-stack aware and protocol-stack agnostic DPI aspects

The DPI identification function (within a DPI-FE) is responsible for application identification and concerns the compare and search operations, based on the DPI signature, against an incoming packet (PDU). There are two options: the DPI-FE is either aware of the internal PDU structure (i.e., "*protocol stack aware DPI-FE*") or unaware of the structure ("*protocol stack agnostic DPI-FE*").

Both options may provide the same identification result and be functionally equivalent. The main difference is that the protocol stack aware identification logic may be more efficient.

It is useful to distinguish the following two types of analysis regarding operational efficiency (i.e., application identification and optional flow identification):

a) Predetermined Payload area Analysis (PPA): When packets (flow) correspond to a known application with a clearly defined payload structure, the DPI-FE may inspect the fixed predetermined location of the payload (i.e., the protocol-stack aware packet inspection mode).

b) Full Payload area Analysis (FPA): When packets (flow) do not correspond to a known application or the structure of the application payload is not clearly defined or known, the DPI-FE inspects the "entire payload area" (i.e., the protocol-stack agnostic packet inspection mode).

Both PPA and FPA can be applied to the same traffic flow.

R-6.3.2/1: The DPI-FE is recommended to support protocol stack aware application identification.

R-6.3.2/2: The DPI-FE is recommended to support protocol stack agnostic application identification.

R-6.3.2/3: The DPI-FE is required to identify applications on top of IPv4 and IPv6 protocol stack and can optionally on top of other underlying protocol stack.

R-6.3.2/4: The DPI-FE is recommended to identify applications in nested traffic, such as encapsulated or tunnelled traffic.

6.3.3 DPI policy rule actions aspects

6.3.3.1 Background

DPI policy actions may be performed on different hierarchical levels, e.g., DPI-FE, local and remote PDFs, and may include for instance the following:

1) Packet path level actions (by the DPI-FE):
   - a) Accept the packet and forward it to the packet forwarding function(PFF)(a conditional action for *In-Path DPI* mode only);
   - b) Discard the packet (silently or otherwise);
   - c) Redirect the packet to other output interfaces;
   - d) Replicate/mirror the packet to other output interfaces;
   - e) Traffic classification, local measurements, and reporting of measurement data;
   - f) Prioritization, blocking, shaping and scheduling methods of individual packets.

2) Node level actions (by involvement of the *local policy decision function* (L-PDF)):
   - a) Dynamic building of new DPI policy rules and/or modification of existing rules (stored in the DPI policy information base (DPI-PIB));

b) Generation of logging/tracing data and reporting to policy management (see clause 2.11.2 in [b-IETF RFC 3871]);

c) Detecting and reporting of unidentifiable applications

d) Notification of intrusion detection systems (e.g., by reporting traffic samples, suspicious packets);

3) Network level actions (via the *remote policy decision function* (R-PDF)):

a) Resource management, admission control and high-level filtering (at the level of network subsystems (such as specified in ITU-T RACF [ITU-T Y.2111], ETSI TISPAN RACS [b-ETSI ES 282 003] and 3GPP PCC [b-ETSI TS 123 203]);

b) Content charging based on subscribers' application types (e.g., IETF RADIUS or Diameter).

Figure 6.2 further explains the above structuring principle through a detailed generic policy rule format (versus the one introduced in clause 1.2):

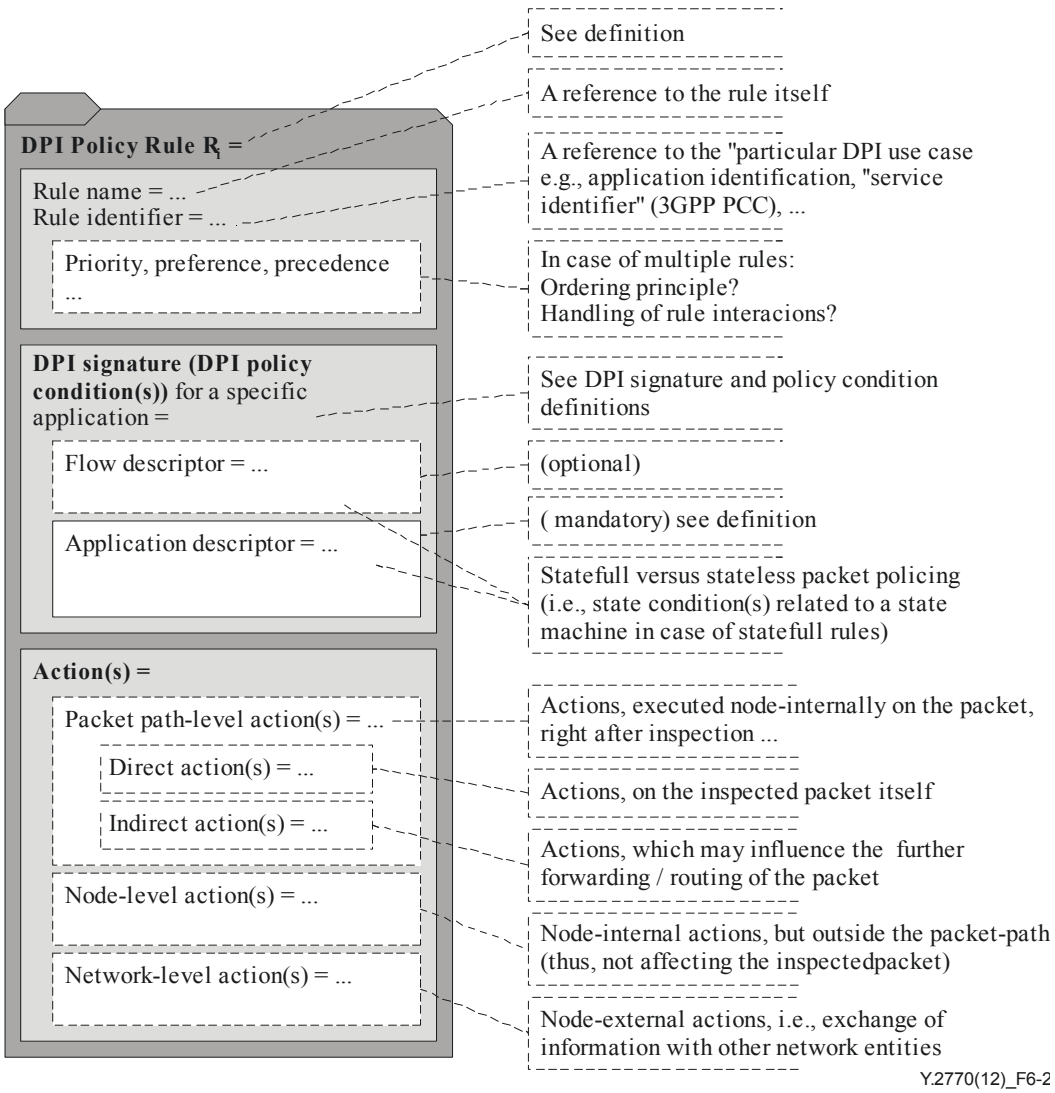


Figure 6-2 – An example of detailed Policy Rule format (in comparison to Figure 1-2/clause 1.2))

The mapping of specific actions to conditions is out of scope of this Recommendation.

6.3.3.2 Requirements

R-6.3.3.2/1: Once an application has been identified by the DPI-FE, it can optionally be possible to extract application specific information.

For example, a URL in HTTP, a media format ("codec type") in real-time transport protocol (RTP), or a RTP session identifier (e.g., SSRC for the RTP source endpoint).

R-6.3.3.2/2: The DPI-FE can optionally be able to work in conjunction with a flow metering function, such as the IPFIX metering process [IETF RFC 5101] and some filtering capabilities, such as [b-IETF RFC 5476].

NOTE – This metering process typically populates the following IPFIX Information Elements (used as flow keys): sourceIPv6Address and destinationIPv6Address, sourceIPv4Address and destinationIPv4Address, protocolIdentifier, sourceTransportPort, destinationTransportPort, etc. However, it is the DPI-FE's role to populate the application tag and the completion of the IPFIX flow identifier (based on the given IPFIX flow key, see also Figure A.1).

6.4 Reporting capability

Reporting concerns the notification (e.g., due to a particular event detected by the DPI-FE) to another functional entity, which is typically located in a remote network element (in the user, control or management plane). The DPI-FE may provide multiple reporting interfaces in support of the "different types of events".

6.4.1 Reporting to the Network Management System (NMS)

6.4.1.1 Interface and protocol for reporting

R-6.4.1.1/1: The export protocol is recommended to follow the IPFIX specification [IETF RFC 5101], and can optionally follow IPFIX extensions.

R-6.4.1.1/2: The export protocol can optionally follow the IPFIX specification [b-IETF RFC 5103] in case of bidirectional flows.

R-6.4.1.1/3: It is recommended that the IPFIX based export protocols use the external interface e2 (see Figure 8.1).

6.4.1.2 Reported information

R-6.4.1.2/1: DPI-FE is required to report inspection results (such as the application tag and potentially application specific information elements) along with the flow specific information to the DPI management plane. The locally updated flow key values (including the typical fields from the flow metering function) can optionally be exported to a policy decision function (e.g., PD-FE defined in [ITU-T Y.2111]).

R-6.4.1.2/2: The reported information is recommended to reuse the IPFIX information elements ([b-IETF IANA IPFIX]), which were initially specified in the IPFIX Information Model [b-IETF RFC 5102].

The flow specific information is specified in the IPFIX information model [b-IETF RFC 5102], for example:

1) Application specific information:
   - Application tag; and
   - Extracted fields such as RTP media format and RTP SSRC.

2) L3/L4 header fields corresponding to IP addresses, L4 ports (e.g., TCP or UDP, NOTE 1), and protocol type;

3) Performance information (like metrics, statistics) *bytes count, packet count*, and *maximum packet size* (NOTE 2);

4) Time information: flow start time, flow end time;

5) Packet associated information: next hop and packet size (NOTE 3);

NOTE 1 – Some listed information elements are not (yet) part of the Internet Assigned Numbers Authority (IANA) IPFIX registry, but they are valid in the context of this Recommendation.

NOTE 2 – The Flow specific information can be generated by packet sampling (PSAMP) mechanism, but when exporting such results to the NMS, the application specific information is recommended be added.

NOTE 3 – New information elements may have to be registered with the IPFIX IANA, according to section 7 "IANA Considerations" of RFC 5102.

6.4.2 Reporting of new, unknown or incorrect application

6.4.2.1 Characteristics of such traffic

There are subtle differences between these application types. They may be characterized by following specific properties, resulting in different application level conditions for their detection:

– new application: e.g., a new version of an application, a new version of an application specific information element (e.g., a new game version within OGP), or a new protocol version; it may be noted that the notion of 'new' reflects the perspective of the DPI service (which may be based on a history of past DPI services);

– unknown application: e.g., an unknown packet type, unknown protocol, unknown "application";

– incorrect application: e.g., a packet carrying incorrect protocol grammar (NOTE), etc.

NOTE – Incorrect protocol syntax could be exploited for a security attack. Affected protocols are typically the ones which are terminated in user equipment (like signalling protocols). [b-ITU-T X.sips] provides an example of a security attack through incorrect SIP syntax.

6.4.2.2 Reporting requirements

R-6.4.2.2/1: The DPI-FE can optionally support reporting new, unknown or incorrect applications upon inspection of the traffic.

6.4.3 Reporting of abnormal traffic

R-6.4.3/1: The DPI-FE can optionally provide a reporting capability related to the detection of *abnormal traffic* upon detection of such traffic.

Abnormal traffic is defined to be the traffic not associated with normal traffic classes (see clause I.6). Normal traffic class is a set of traffic that matches to existing statistical properties of well-defined applications, such as the packet inter-arrival time, arrival order, the size of the PDU of a specific protocol layer, the size of payload, or the traffic volume (at a specific protocol layer).

6.4.4 Reporting of events related to the DPI-PE

This clause describes the events concerning the operational state of the DPI entity and the related reporting requirements.

6.4.4.1 Failure events related to incorrect behaviour of the DPI-PE

The simplest way to depict the management state of the DPI-PE is in terms of two states: "In-Service" (IS) and "Out-of-Service" (OoS).

R-6.4.4.1/1: DPI management is recommended to be based on the state of art (e.g., [ITU-T X.731] and [b-IETF RFC 4268]) and is recommended to support at least the management states of IS and OoS.

R-6.4.4.1/2: Any failure of the DPI-PE, if not architected in a redundant manner, can optionally cause the IS-to-OoS state transition. Such events are recommended be reported.

6.4.4.2 Events related to fault management of the DPI-PE

A DPI-PE provides network interfaces for ingress and egress traffic. Fault may occur on these interfaces.

R-6.4.4.2/1: The DPI-PE is recommended to support an alarm reporting function such as defined in [b-ITU-T X.734].

6.4.4.3 Events related to logging of the DPI Functional Entity

R-6.4.4.3/1: The DPI Functional Entity can optionally support a system logging capability according e.g., *Syslog* [b-IETF RFC 5424]. In such a case, the DPI Functional Entity is an originating point of *Syslog messages*.

It is worth noting that in the case when the inspected packet flow carries logging traffic, the DPI Functional Entity is neither an originating point nor a terminating point of logging messages. In other words, the lookup-key for such a packet flow may be based on an *application descriptor* (related to the *syslog application* layer) and an IPFIX *flow descriptor* (related to the selected *syslog transport* mode). Further information can be found in [b-IETF RFC 5424] and [b-IETF RFC 5426].

6.4.4.4 Events related to the load state and resource consumption of the DPI Physical Entity

A DPI-PE has limited resources for DPI processing. The resource specifics are implementation-dependent and out of scope of this Recommendation.

R-6.4.4.4/1: The DPI Physical Entity is recommended to support reporting of the load level of DPI resource components to the management plane.

For instance, in networks with Emergency Telecommunication traffic (see clause 7.1.1), the DPI process must be able to forward ET traffic through congested network nodes; therefore it is desirable that the network management system be aware of the load level.

6.5 Interaction with a policy decision function

R-6.5/1: DPI-FE can optionally act as a part of the policy enforcement functional entity as defined in [ITU-T Y.2111] and provide the related transport function.

R-6.5/2: The interface between the DPI-FE and RACF can optionally be Rw as defined in [ITU-T Y.2111].

R-6.5/3: The information between DPI-FE and the RACF PD-FE can optionally be exchanged via *existing* (e.g., the *Rw* interface) or *new* RACF interfaces depending on the specific DPI use case.

NOTE – In this case, the RACF needs to be enhanced to cover DPI information (e.g., a protocol signature within a DPI policy rule); the RACF as defined in [ITU-T Y.2111] supports primarily flow-identification based policy rules. The specific RACF reference point would be dependent on the specific DPI use case.

6.6 Traffic control

Clause I.5 provides some high level use cases with respect to the involvement of a DPI function in traffic control scenarios. Following high-level requirements may be derived:

R-6.6/1: The DPI functional entity may optionally be involved in network scenarios with the purpose of traffic control (e.g., traffic control functions as defined by [ITU-T Y.1221], or "bandwidth optimization" scenarios indicated in Appendix I, or traffic rerouting). The DPI-FE is recommended to support corresponding traffic control capabilities.

R-6.6/2: The DPI-FE can optionally support traffic control natively. Nevertheless, the detailed functional requirements for traffic control are out of scope of this Recommendation.

R-6.6/3: The DPI-FE can optionally support interactions with external traffic control functions. The related functional requirements are out of scope of this Recommendation.

6.7 Session identification

There are many terms related to *session* in this Recommendation. All traffic of a session can be unambiguously identified by the DPI-FE since that the "*session descriptor*" is either equal to or a subset of the flow and/or application descriptor (see also clause VII.5).

6.7.1 Requirements for session identification

R-6.7.1/1: The DPI-FE is required to be able to analyse session (e.g., RTP session, HTTP session, IM session, VoIP SIP session) behaviour.

R-6.7.1/2: The DPI-FE is required to be able to track session state.

6.7.2 DPI actions at 'session level'

R-6.7.2/1: The DPI-FE can optionally extract or generate measurement data at the session level (e.g., for monitoring performance metrics concerning a subscriber's quality of experience.

6.8 Inspection of encrypted traffic

There is a common view that DPI signatures can be applied only to unencrypted traffic. Nevertheless, DPI signatures could be applicable to encrypted traffic depending on

- The level of encryption (see clause 6.8.1);
- local availability of the decryption key (see clause 6.8.2);
- inspection conditions based on encrypted information (see clause 6.8.3).

6.8.1 Extent of encryption

Any 'packet' as protocol data unit (PDU) consists of protocol control information (PCI) and service data units (SDU) at various protocol layers. When encryption is applied on the inspected communication path, then encryption may be applied:

- either to the entire protocol stack or only to a part of the protocol stack (NOTE 1), and,
- within a protocol layer, either to the PDU of a layer x (Lx) (i.e., complete Lx-PDU) or only partially (e.g., just the Lx-PCI or Lx-SDU part).

NOTE 1 – Example: an RTP-over-IP packet service may provide encryption on:

a) network layer (e.g., via IPsec transport mode or IPsec tunnel mode);

b) transport layer (e.g., via DTLS); or/and

c) application layer (e.g., via SRTP).

DPI can be performed on any unencrypted part of the packet.

R-6.8.1/1: Awareness of encrypted traffic (from DPI signature perspective): DPI can optionally be performed on all unencrypted information elements of the inspected traffic, dependent on the extent of encryption (NOTE 2).

NOTE 2 – Example: an SRTP-over-IP packet flow may be still inspected in case of DPI signatures, based on information elements on RTP PCI ("RTP header"), UDP PCI ("UDP header"), IP PCI ("IP header"), etc., if just the RTP SDU (containing the IP application data) is encrypted.

R-6.8.1/2: Unawareness of encrypted traffic (from DPI signature perspective): DPI can optionally be performed as a partial DPI (because parts of the DPI signatures could be related to unencrypted packet information elements).

Such a "partial DPI" on encrypted traffic may lead to "limited DPI services" (such as described in Appendix I), but already enough for specific use cases (e.g., if a "coarse granular" identification of an application or protocol would be already sufficient).

6.8.2 Availability of decryption key

R-6.8.2/1: DPI can optionally be applied in case of a local availability of the used encryption key(s). Any DPI enforcement will then imply an initial decryption of (a local copy) the inspected packet.

6.8.3 Conditions for inspections based on encrypted information

R-6.8.3/1: DPI can optionally be supported on encrypted traffic, in case of policy conditions applicable for inspections based on encrypted information (NOTE 3).

NOTE 3 – Example: a bit pattern (which unambiguously identifies a particular packet flow) may be derived by the observation (inspection) of a partially encrypted traffic (see clause 6.8.1). The bit pattern as part of subsequent DPI signatures would be then already available in the encrypted encoding.

6.8.4 IPsec-specific DPI requirements

The requirements stated in subclauses 6.8.1 to 6.8.4 are also valid for IPsec encrypted packets. This Recommendation is focusing on the flow identification aspects of IPsec encrypted traffic. The aspects related to application identification are for further study.

6.8.4.1 General requirements

R-6.8.4.1/1: The DPI-FE can optionally be able to support at least *flow identification* for IPsec encrypted traffic. The corresponding flow descriptor n-tuple can optionally be limited to only the L2 and L3 based elements.

R-6.8.4.1/2: A flow can optionally correspond to the traffic of a single IPsec *Security Association* (SA), or can optionally span multiple SAs.

R-6.8.4.1/3: The SA-based flow identification implies that the 32-bit IPsec *Security Parameter Index* (SPI) can optionally be part of the Flow Descriptor.

6.8.4.2 IPsec tunnel and transport mode

The IPsec protocols (AH and ESP, see below) can be used to protect either an entire IP payload (i.e., the tunnel mode) or the upper-layer protocols of an IP payload (i.e., the transport mode).

R-6.8.4.2/1: The DPI-FE can optionally be able to detect IPsec encrypted traffic in the tunnel mode.

R-6.8.4.2/2: The DPI-FE can optionally be able to detect IPsec encrypted traffic in the transport mode.

6.8.4.3 IPsec AH- protected traffic

The *Authentication Header* (AH) provides data integrity, data origin authentication and limited optional anti-replay services.

R-6.8.4.3/1: The DPI-FE can optionally be able to detect AH- protected traffic based on the corresponding IP protocol number.

6.8.4.4 IPsec ESP- protected traffic

The *Encapsulating Security Payload* (ESP) provides additionally confidentiality.

R-6.8.4.4/1: The DPI-FE can optionally be able to detect ESP- protected traffic based on the corresponding IP protocol number.

6.9 Inspection of compressed traffic

The purpose of compression is to reduce the amount of traffic. For examples:

- "ZIP"-based compression reduces file sizes (relevant to FTP-over-TCP/IP flows);
- "SigComp"-based compression [b-IETF RFC 3320] reduces the size of SIP messages (relevant to SIP-over-L4/IP flows).

6.9.1 Awareness of compression method

R-6.9.1/1: DPI can optionally be supported when local information on the applied compression scheme would be available (e.g., if DPI node is aware that the inspected SIP signalling path is encoded according to clause 8 of [b-ETSI TS 124 229]). Any DPI enforcement would then imply an initial decompression of (a local copy) the inspected packet.

R-6.9.1/2: DPI can optionally be also supported if it is possible to derive the applied compression scheme from the inspected traffic flow (e.g., the particular zip compression method can optionally be derived from file header information elements).

6.10 Detection of abnormal traffic

See Appendix I.6 concerning the background and use cases for distinguishing normal from abnormal traffic.

6.10.1 Requirements for detection of abnormal traffic

R-6.10.1/1: The DPI-FE is required to be able to support detection of abnormal traffic. Namely, the DPI signatures are required to be able to characterize normal and abnormal traffic (e.g., either as a black or white list).

NOTE – DPI policy rule aspects: This capability could imply the check of many metrics with regards to traffic and/or packet characteristics, as well as possibly maintaining a decision tree for the final conclusion concerning normal or abnormal traffic classes.

7 Functional requirements from the network viewpoint

7.1 General requirements

7.1.1 Emergency Telecommunications

The overall design, implementation, deployment and use of DPI functions have to include appropriate measures to prevent negative impacts to the performance and security of Emergency Telecommunications (ET). ET [ITU-T Y.2205] means any emergency related service that requires special handling relative to other services (i.e., priority treatment over regular services). This includes government authorized emergency services, e.g., Emergency Telecommunication Service [ITU-T E.107] and public safety services.

This Recommendation is based on the use of an application tag to identify different application semantics such as application protocol type (e.g., H.264 video, or SIP as an example IP application protocol) in a generic manner. The same application types (e.g., SIP) are used to support both regular services and ET application services. However, this Recommendation does not specify any unique application tag to identify ET application services. Therefore, appropriate precautions would be necessary to prevent negative implications on ET application services.

R-7.1/1: It is required to not interfere with the priority treatment of ET application services traffic over ordinary services.

R-7.1/2: It is required that the overall design, implementation, deployment and use of DPI functions include appropriate measures to prevent negative impacts to the performance of ET application services (e.g., introducing unnecessary delays).

R-7.1/3: It is required that the overall design, implementation, deployment and use of DPI functions include appropriate measures to prevent introduction of security compromises to the integrity, confidentiality or availability of ET communications/sessions.

NOTE – This Recommendation does not provide any stipulations as to how the above requirements are to be met. The requirements could be achieved through the use of functional capabilities, operational measures or a combination of both.

7.2 Data plane, control plane and management plane in DPI node

7.2.1 Traffic planes and traffic types from DPI node perspective

Following the network model of user-, control and management plane (see [b-ITU-T Y.2011]), a DPI node deals with a data path and local decision path (see Figure 7-1). The data path can work either in unidirectional or in the bidirectional mode.

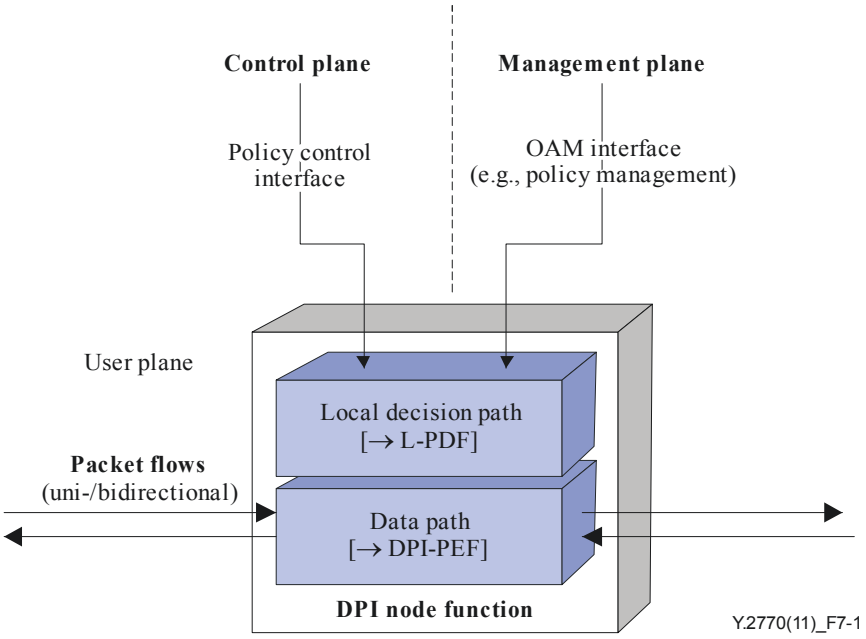


Figure 7-1 – External and internal traffic planes of a DPI node

NOTE 1 – The packet flows are routed/switched along the packet paths, which are often called *data paths* in IP networks (see e.g., [b-IETF RFC 4778]); therefore the term *data plane* is synonymous to *user plane*.

NOTE 2 - The IP data path is also known as IP *media path* (or *bearer path*) in case of IP application data traffic, or IP *signalling path* in case of IP application control traffic [b-ITU-T X.1141].

R-7.2.1/1: A DPI node is required to support the management plane interface for policy management and can optionally support the control plane interface for policy control.

The *Local Decision Path* entity provides the node-internal control and management capabilities.

R-7.2.1/2: A DPI node is required to recognize two kinds of packets (see Figure 7-2):

a) data packets, which belong to customers and carry customer traffic (called “traffic THROUGH”, see [b-IETF opsec]); and

b) control and management packets, which belong to the network provider and have to do with network operations (called “traffic TO”; see [b-IETF opsec]).

The two kinds of packets traverse a “common pipe” (or are “in-band”) or traverse different channels that logically separate data from “out-of-band” control packets (see also [b-IETF RFC 4778], clause 2.2 for an example of management traffic).

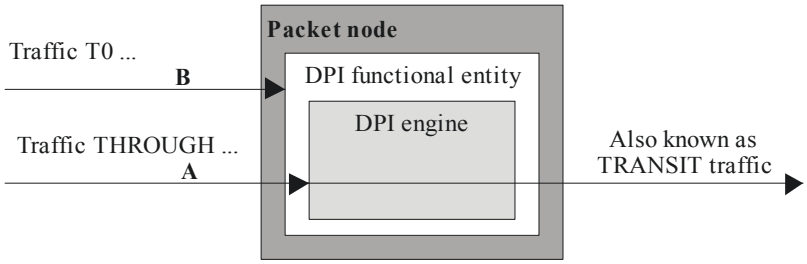


Figure 7-2 – Traffic THROUGH (A) and TO (B) a DPI node

7.2.2 Requirements related to management plane

R-7.2.2/1: DPI-FE is required to support management protocols for configuration management of DPI policy rules.

R-7.2.2/2: DPI-FE is recommended to support management of user's identity information and the relationship between the user and user's applications.

R-7.2.2/3: DPI-FE is recommended to support management of applications and services

- Generate, modify and publish application templates;
- Maintain the relationship between applications and strategies; and
- Provide and manage reservation of user's service;

R-7.2.2/4: DPI-FE is recommended to support management of strategies predefined or generated dynamically. (These strategies can optionally relate to application identification, application control, and user management.)

R-7.1.2/5: DPI-FE is recommended to support management of administration authority. To support hierarchical management, different administrator has different management authority.

7.2.3 Requirements related to control plane

R-7.2.3/1: DPI-FE can optionally support of policy control protocols (like [b-ITU-T H.248.1] for the ITU-T *Rw* reference point as defined in [ITU-T Y.2111]) for control and signalling of DPI policy rules.

7.2.4 Requirements related to user (data) plane

The data (user) plane meets the following optional requirements:

R-7.2.4/1: DPI-FE can optionally support different packet technologies (e.g., xDSL, UMTS, CDMA2000, cable, LAN, WLAN, Ethernet, MPLS, IP, ATM).

7.2.5 Requirements across planes

R-7.2.5/1: DPI-FE can optionally support an aligned protocol grammar for the specification of DPI policy rules. The syntax used at the policy control interface (control plane) and policy management interface (management plane) is recommended be preferably identical. This does not imply the use of the same protocol, but concerns the specification language for (DPI) policy rules (often called Filter Specification Language (FSL), or Policy Specification Language (PSL); see NOTE 1).

NOTE 1 –An example script languages is SIEVE [b-IETF RFC 5228] or PERL, or XML or XACML (eXtensible Access Control Markup Language)

An aligned protocol grammar allows the use of a common data/object model in the policy enforcement path within a DPI node, which is a prerequisite for efficient and fast rule execution as well as the interruption-less update operations on the DPI signature library.

8 Interfaces of the DPI-functional entity

The requirements described in the previous clauses entail the following interfaces:

- between the DPI-FE and remote network entities (see clause 8.1), and
- between DPI-FE internal components (see clause 8.2).

8.1 External DPI-FE interfaces

Figure 8-1 depicts the external interfaces of the DPI-FE:

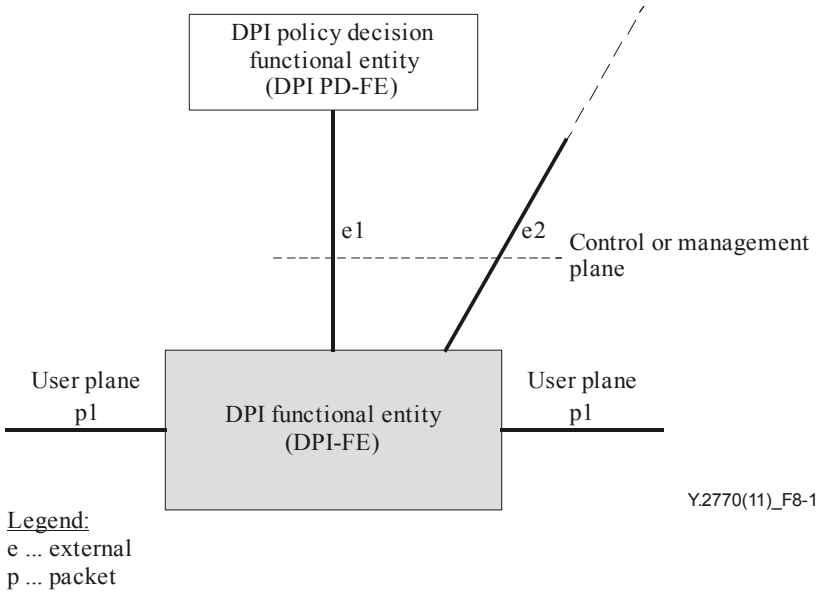


Figure 8-1 – External DPI-FE interfaces

8.1.1 Inspected traffic (p1)

The DPI-FE exchanges packets with remote packet nodes via *p1*. The packet path topology is point-to-point for a DPI-FE acting in the *In-Path DPI* mode. Multipoint topologies are not supported. Interface *p1* covers *bidirectional* packet paths.

The packet path topology for a DPI-FE acting in the *Out-of-Path DPI* mode is related to an endpoint.

8.1.2 Control/management of traffic inspection (e1)

The DPI policy decision functional entity (DPI-PDFE) aims to control or manage the DPI-FE. The information exchanged via *e1* thus concerns the commands for controlling/configuring the packet handling behaviour of the DPI-FE. Such commands could be described in a DPI policy.

Interface *e1* could also support reporting and notification from DPI-FE to DPI-PDFE.

8.1.3 Reporting to other network entities (e2)

Interface *e2* encompasses all possible communication interfaces with remote network entities other than the DPI-PDFE. This interface primarily supports reporting.

8.2 Internal DPI-FE interfaces

Figure 8-2 shows the possible internal interfaces based on the DPI requirements:

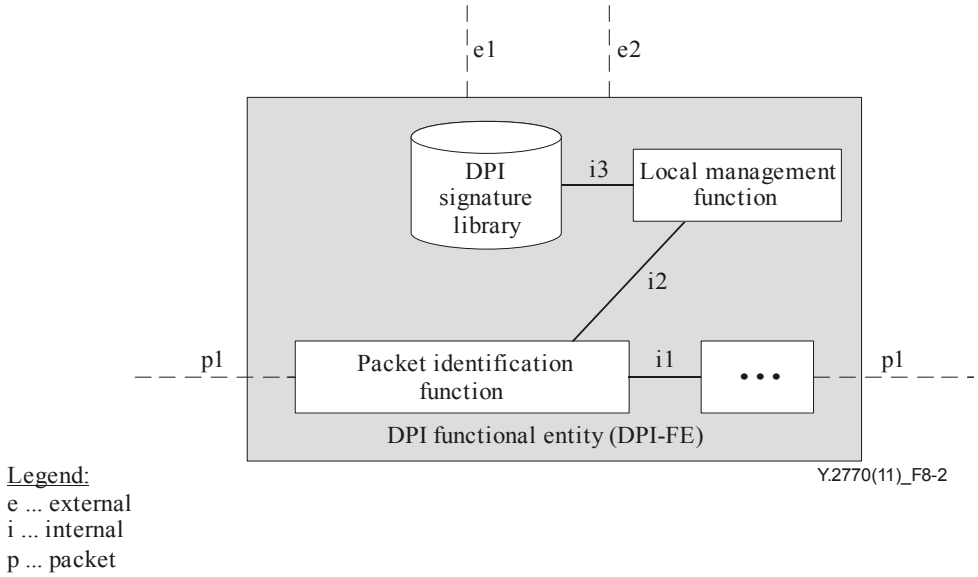


Figure 8-2 – Internal DPI-FE interfaces

There might be further DPI-FE internal functional components and internal interfaces. The internal interfaces are for further study.

8.3 Interface requirements

R-8.3/1: It is recommended that interface e1 follow the requirements in clause 6.5.

R-8.3/2: It is recommended that interface e2 follow the requirements in clause 6.4.1.

9 Security considerations and requirements

This clause describes security threats and defines security requirements for DPI entities in NGN.

9.1 Security threats against DPI entities

The functional entities associated with DPI may are typically located within an NGN operator's *trusted zone* or *trusted but vulnerable zone* as defined in ITU-T Recommendation Y.2701 [ITU-T Y.2701]. The Recommendation identifies the security threats to NGN and defines the requirements for protection against the threats. Since the DPI-related entities are a part of NGN, the conclusions of [ITU-T Y.2701] are applicable to them. Based on [ITU-T Y.2701] the security threats related to the DPI entities are identified as follows:

- Destruction of DPI-related information;
- Corruption or modification of DPI-related information;
- Theft, removal or loss of DPI-related information;
- Disclosure of DPI-related information;
- Interruption of services

The information pertaining to the DPI operations include DPI policy rules with their signatures and DPI exported flow and application information. Destruction, corruption or modification, theft, removal or loss of such information may make it unusable for the DPI operations. In many countries, such information is recommended to be treated according to the national regulatory and policy requirements and must not be disclosed.

Interruption of services may be result of the DoS attacks. Any entity receiving data can be a target of DoS attack. For example, an attacker can indirectly flood a DPI entity with large volume of traffic causing degradation or interruption of the DPI services for the legitimate users.

9.2 Security requirements for DPI entities

The major security requirements for DPI entities are:

R-9.2/1: The DPI-related information residing in DPI entities is required to be protected.

R-9.2/2: If the information is exchanged beyond the NGN operator's *trusted zone*, the DPI-related information is required to be protected between DPI entities and the remote functional entities (e.g., DPI PD-FE, NMS)

R-9.2/3: Mechanisms can optionally be required to mitigate the flooding attack against the DPI FE.

R-9.2/4: Vendors, operators and service providers are required to take into account national regulatory and policy requirements when implementing this Recommendation.

R-9.2/5: The implementers are recommended to employ the existing well-tested mechanisms for meeting the security requirements of this Recommendation. For example, as specified in [ITU-T Y.2704].

ANNEX A

Specification of flow descriptor

(This annex forms an integral part of this Recommendation.)

A.1 Protocol syntactical perspective

The flow descriptor relates to a *data structure* (data object), which may be modelled as *k-Tuple* (see Figure A.1). The data structure consists of *k* information elements (IE) (NOTE 1). The value of *k* is variable and greater than zero[1], but constant for a particular flow. The *information elements* are the ones as contained in the IANA IPFIX registry. There is a *value* associated to each information element. The *association* is typically mathematical equality ('='), but other mathematical relations are not excluded.

NOTE 1 – The IETF IPFIX information elements may be attributed as "key field" or "non-key field".

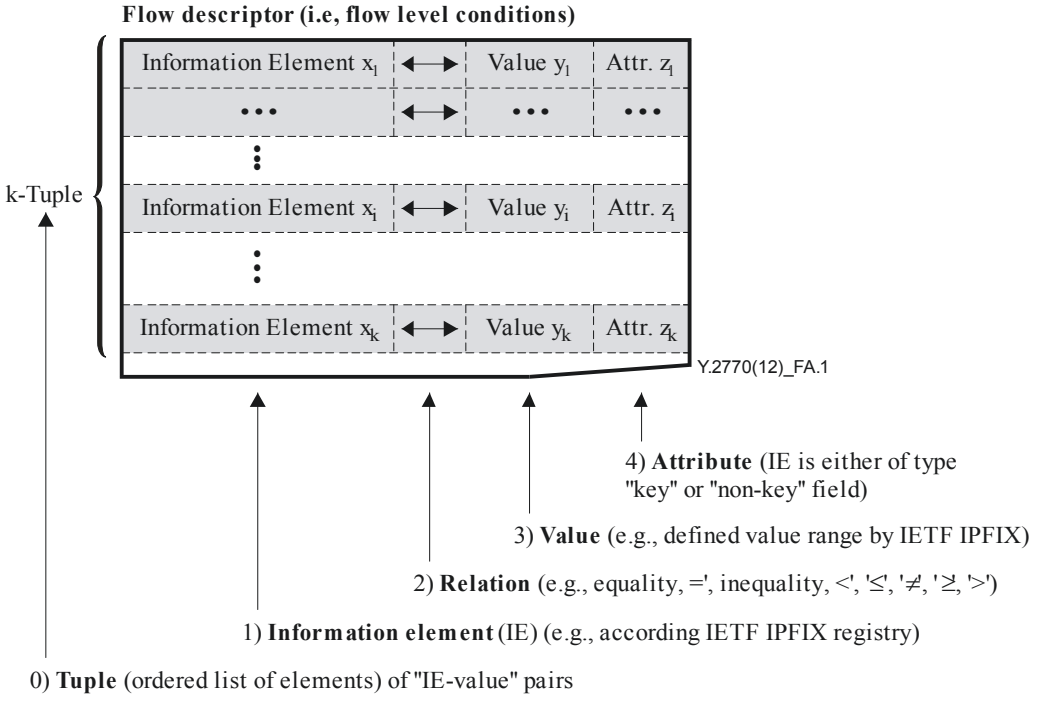


Figure A.1 – The flow descriptor (flow level conditions) from protocol syntactical point of view

The flow level descriptor as a *k*-tuple represents consequently a list of *k* "*name-value pairs*" (NVP); here a sequence of "< *IE* ↔ *value* >" pairs)[2].

A.2 Specifying information element values

In the flow level conditions, the value of an IE may be:

---

1 Note: N = 0 indicates "Flow-independent "

2 Similar to other structures like AVP (<attribute name, value>), parameter-value pair (<parm=value>), etc.

- fully specified

  Full specification represents the case of a complete name-value setting.
- not specified

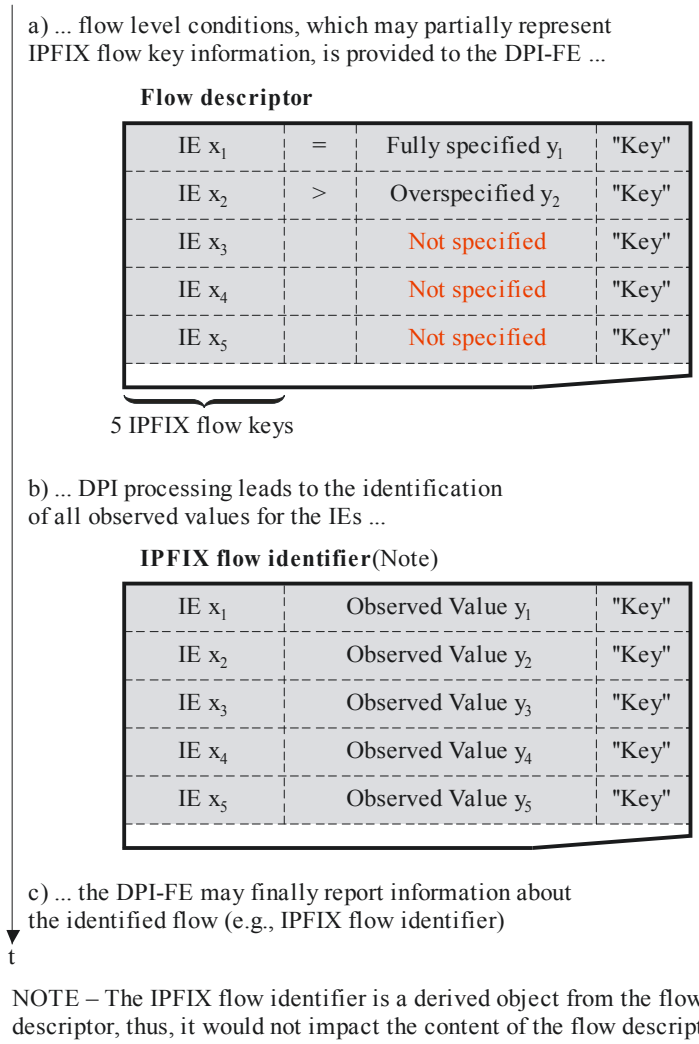
  "*Not specified*" represents the case when there is *not yet any value* assigned to an IE.
- overspecified or

  Overspecification indicates there are multiple possible values for a specific IE.
- underspecified.

  Underspecification indicates wildcarding (e.g., all possible values, or choose value).

A.3 Relation between flow descriptor, IPFIX flow identifier and IPFIX flow key

The example in Figure A.2 provides a 5-tuple flow descriptor and contains 5 IFPIX flow keys. In order to identify a particular flow, the flow descriptor imposes some conditions on the values of these flow keys as defined in clause A.2: the first flow key IE x_1 is "fully specified", the second flow key IE x_2 is "overspecified", while the others IEs are "not specified", as displayed in the part a) of Figure A.2.



NOTE – The IPFIX flow identifier is a derived object from the flow descriptor, thus, it would not impact the content of the flow descriptor.

Figure A.2 – Flow Descriptor, IPFIX Flow Identifier and IPFIX Flow key example

Note that the flow descriptor doesn't impose conditions on the IPFIX flow keys only: indeed, in some circumstances, flow descriptors on non-flow key might be required, for example when a

condition of the TCP flags of the first packet of the flow is required. The fundamental difference between the flow descriptor and the IPFIX flow identifier in the example Figure A.2 is that the flow descriptor contains a "superior than" condition on the IE x_2, ("IE x_2 > value y_2"), while the IPFIX flow identifier contains the observed value for the IE x_2, i.e., value yy_2. The IPFIX flow identifier is composed of the set of observed values for the flow keys, once the DPI functional entity processed the packets and classified them into a flow.

Note that if the exported information (e.g., via an IPFIX flow record) contains each IEs along with the associated observed values, and whether or not the IE is an IPFIX flow key, then there is no need to assign a specified IPFIX flow identifier, as the IPFIX flow identifier is the sum of all this information.

APPENDIX I

Application Scenarios

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

I.1 Introduction

The purpose of this appendix is to list application scenario for DPI-based services. It provides thus a collection of example use cases, which are again demanding for particular DPI support. Correspondent DPI support leads to the identification of DPI requirements, as subject of the main body of this Recommendation. There are basically generic DPI requirements which are use case independent and application-specific DPI requirements, only part of a particular use case.

The application scenarios address basically following high-level questions:

1) *Where* is the DPI function located in the network ("the network location aspect")?

2) *Which* "traffic" entity is inspected ("the application and flow identification aspect")?

3) *What* is the purpose of inspection ("the aspect of packet handling and further proceeding activities")?

Each application scenario could be subsequently translated in use case specific packet policing behaviour, which may be described for the DPI-FE by

- *conditions* for packet inspection (i.e., the implementation of above question (2)) and
- follow-up *actions* (i.e., the implementation of above question (3)).

Such lower level details are out of scope of the example use case illustrations in this Appendix.

I.2 DPI use cases: Application scenarios in packet-based network

In packet-based networks, it is imperative to identify different kinds of services and apply different control mechanisms to provide differentiated services for its subscribers. As a control point in packet forwarding, DPI is often deployed in the following application scenarios, as illustrated in the subsections.

In scope of this clause are IP networks only.

I.2.1 Differentiated services based on service identification

One of DPI's fundamental functions is to provide differentiated services as defined by IETF (see [b-IETF RFC 2474]) for subscribers in a public network and enterprise network. Service identification is the prerequisite for operators to provide differentiated services for its customers. In Figure I.1, DPI is deployed at different layers in the operator's networks and is transparent to the subscribers.

In such scenario, DPI is often deployed as a real-time operation (for real-time and non-real-time end-to-end applications) which makes all services visible and easy to manage. Traditionally, packet forwarding can unveil some information about the carried traffic by extracting and parsing the basic protocol information such as connection address information like, e.g., IP addresses (source, destination) and other lower layer and higher protocol information. This information typically resides in the packet header itself and consequently reveals the principal communication intent, such as HTTP, FTP, and email services, as the port number often indicates the carried services.

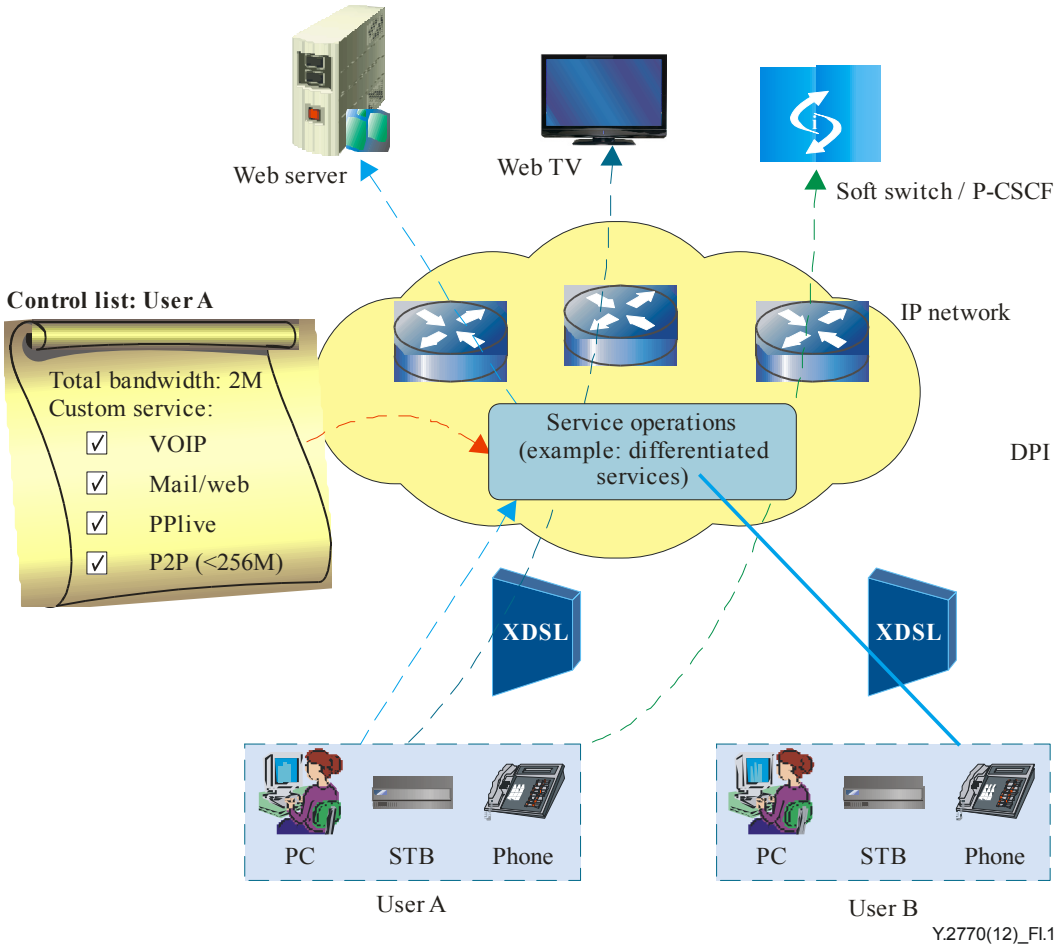


Figure I.1 – Scenario of differentiated services at the example of an IP-based packet network with DPI support

As depicted in Figure I.1, user A, whose available bandwidth is, e.g., 2 Mbit/s, has different network needs including services like VoIP, mail/web, on-line video (e.g., pplive), peer to peer services and so on. When the user's application traffic flows pass through the DPI device in the network, the DPI may implement differentiated services in accordance with the predefined control list (i.e., the policy rules table for DPI in the packet forwarding path) and the identification results.

However, from the perspective of service awareness, it is insufficient to reach any application-related service identifications.

To achieve the purpose of service awareness, DPI is applied to probe deeper into the packets in a multi-services stream for, e.g., content analysis. Ways of service identification are listed below, as it is often used in most service control scenarios, though sometimes they are not sufficient to make the most successful decisions as what kinds of services being carried:

1) Analysis based on layer 4 port number (in case of IP):

   This is the simplest way in classifying the carried services, however a conditional method due to the assumption of the usage of so-called *well-known ports* ([b-IETF IANA Port Number Registry]) as transport endpoint identifiers for the IP application;

2) Analysis by string match:

   Sometimes a typical string which indicates the application type is embedded in the traffic, thus deep inspection into the packet content should be involved to find the exact match (or partial match) which indicates the kinds of services being carried;

3) Analysis by numerical properties:

Analysis by numerical properties involves the investigation of arithmetic and numerical characteristics within a packet, and of a packet or several packets. Some examples of properties analysed include payload length, the number of packets sent in response to a specific transaction, and the numerical offset of some fixed string (or byte) value within a packet.

4) Analysis by behaviour and heuristics:

In such kinds of application scenarios, DPI is deployed as an intelligent component in service identification. In a generic application of this kind, behavioural analysis refers to the way a protocol acts and operates, while heuristic analysis typically boils down to the extraction of statistical parameters of examined packet transactions. In some scenarios, behavioural and heuristic analyses are combined to provide improved services identification capabilities.

In most application scenarios, services after identification can be categorized and marked as one of the following attributes:

1) Quality sensitive and real-time services, such as VoIP services;

2) Quality sensitive but not time-sensitive, such as management and routing information;

3) Best effort services, such as traditional services like HTTP (for web browsing), FTP (for file transfer), and SMTP (for email), etc.;

4) Services unidentified.

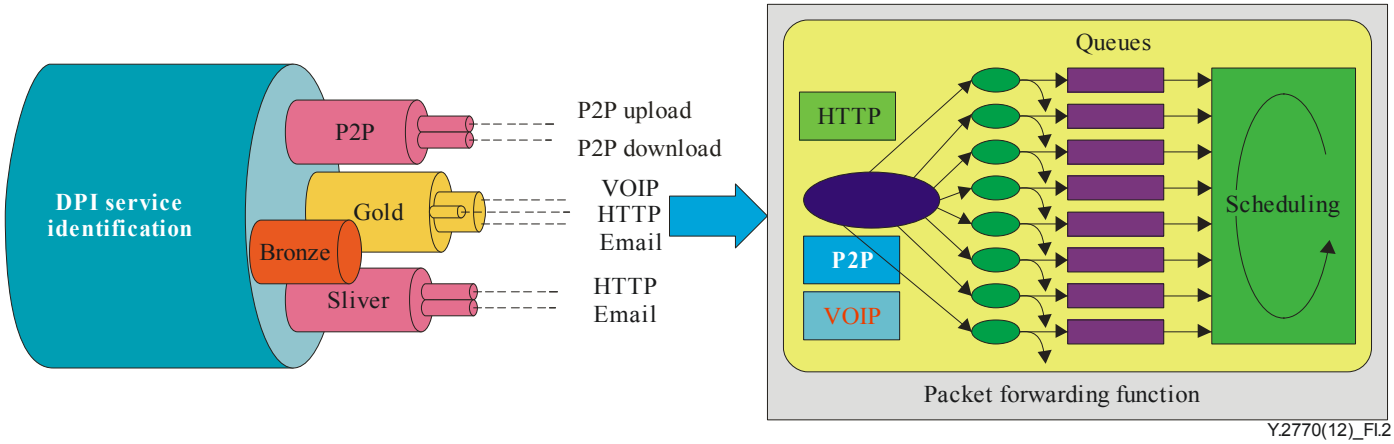
From the perspective of service control, DPI is often used as an auxiliary tool for providers to personalize services to its users, including: new services creation, content filtering to avoid offending the subscribers, resource allocation varying from application to application, etc., including:

- Limited service packages based on subscriber awareness in accordance with service level agreement (SLA);
- Expanded service packages based on subscriber awareness in accordance with SLA;
- Tiered service packages based on time of day and allocated bandwidth amounts for various applications;
- Additional provisioned bandwidth dedicated for a specific user application;
- Quality of service (QoS) assurance for all traffic from a specific user; or
- QoS assurance for traffic of a certain type or from a certain source for a specific user.

I.2.2 Traffic monitoring

Another important scenario where DPI is widely deployed is that DPI is often used as the key control points enabling traffic management: scanning, filtering or forwarding packets based on services identified protocol layer 2 to layer 7 (e.g., in the case OSI X.200 basic reference model). From the perspective of packet forwarding, each service will be delivered as one or more flows in the network. To better control the traffic, a pre-configured or intelligently deduced policy is often applied which makes all the identified services under the operator's supervision as desired. When the services are identified, different traffic of different services will be forwarded based on their attributes, for example, they can be forwarded along different path to their destinations based upon their SLA requirements.

Another aspect of traffic management is resource allocation as resources can be proportionally allocated based on the subscribers' profile and services control policy, as it is showed in Figure I.2.



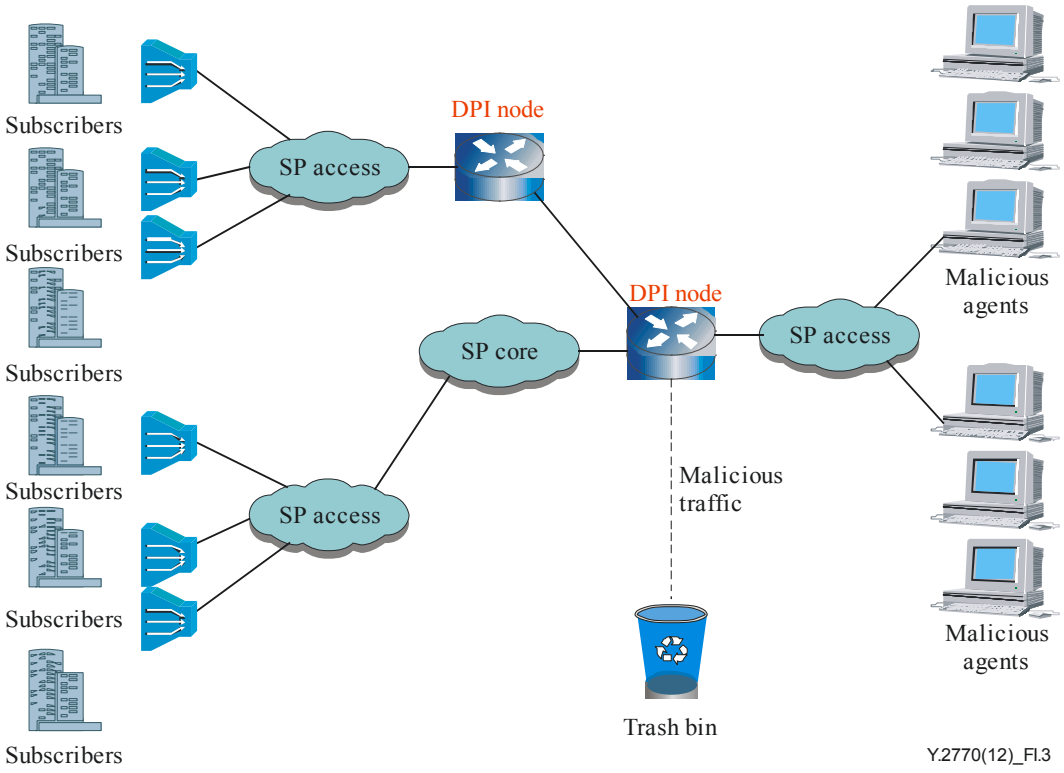
NOTE – The diagram outlines a DPI-FE and a packet forwarding function (PFF) in the packet path. The purpose of the DPI-FE is here related to service-based traffic classification. The PFF provides service-based packet forwarding.

Figure I.2 – DPI used for the purpose of traffic monitoring

I.2.3 Security

DPI may be deployed to provide the capabilities to identify malicious traffic that may degrade user performance, drain network resources, impair infrastructure, and finally make the network unavailable to its subscribers. Most of the malicious traffic disguises itself as normal traffic and is extremely bandwidth consuming, such as: Outgoing spam (NOTE 1), IP scanning and port scanning, etc. Figure I.3 shows a typical application scenario that when malicious traffic is identified, it will be removed by the DPI component from the traffic thus preventing it from spreading into the network.

NOTE 1 – E.g., a DPI function may be a component of an *interactive gateway system for countering spam* according to [b-ITU-T X.1243]. Clause 6 of [b-ITU-T X.1243] illustrates possible methods and policy conditions for DPI-based spam identification (i.e., 'spam' represents here the "DPI application traffic").



NOTE – SP means Service Provider, the network scenario is independent of specific network access technologies (e.g., xDSL, Cable, PON, wireless).

Figure I.3 – DPI deployed to filter out malicious traffic

Therefore, DPI in such kinds of application scenarios provides the packet forwarding process with the following capabilities:

1) On-line monitoring, tracking and analysing possible connections.

2) Real-time identification of malicious attacks. Through inspecting deep into the traffic, DPI alerts the network administrators of possible attacks and providing a range of monitoring and detection tools to track attack launchers, applications, flows, connections, ports, protocols, trends and other parameters. Meanwhile, some of the attack patterns if possible, should also be feed backed to the DPI signature library to make resources unavailable to the attackers.

3) Real-time reporting of possible attacks. Through an automated and flexible early-warning mechanism DPI is applied to inform network administrators of potential threats in advance, enabling them to take appropriate actions against possible attacks. Thus, DPI may be fundamentally a basic function of an intrusion detection system (IDS).

Mitigation of threats through DPI policy rules enforcement. In such scenario DPI is deployed to yield productive measures against possible attacks. Under such circumstances, mitigation of threats would be implemented to avoid the infiltration of the malicious traffic into the network, as it might make the network more vulnerable and even collapse from resource exhaustion.

I.2.4 Traffic statistics and services-based billing

Through the perceptions to subscribers and applications, DPI can provide comprehensive statistics of application flows, which can help the network operator master all the information about network load, and the bandwidth occupation of every application. Service providers often charge their customers based on the services they subscribed as different services may have different billing policies. For example, time-critical and delay-sensitive services often comparatively consuming

more resources and will be charged higher while legacy internet services such as Email, HTTP may exert fewer demands on resources and will be charged much lower. From the perspective of operators, as depicted in Figure I.4, this kind of billing issues are often termed as services-based billing.

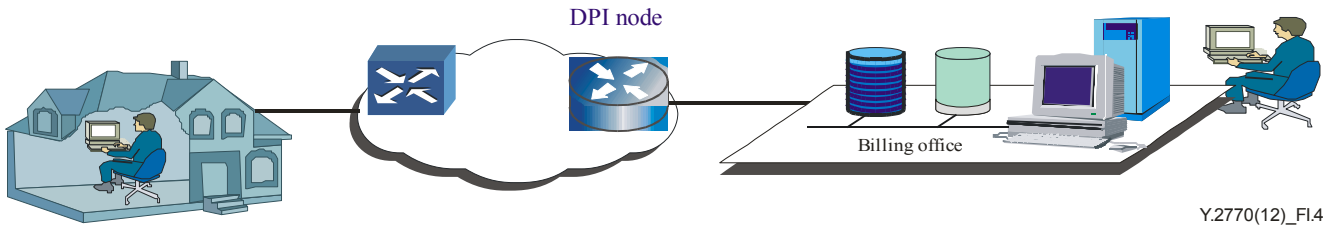


Figure I.4 – Traffic statistics and services-based billing

I.3 DPI use case: Application scenarios of DPI specific to NGN

Currently there are few NGN deployments around the world. We can only deduce some of its possible application scenarios based on its already defined 2-stratum architecture ([b-ITU-T Y.2012]).

Figure I.5 illustrates an example how DPI can be deployed in an NGN environment. The functional blocks included in the dashed box are components that are closely related to the process of both DPI and packet forwarding: the blocks of policy control consists of subscriber policy control and its corresponding policy repository; the block of resource admission control subsystem is responsible for subscriber authentication/admission and resource allocation while the block of DPI will carry out all the functions of packet header analysis and content scanning.

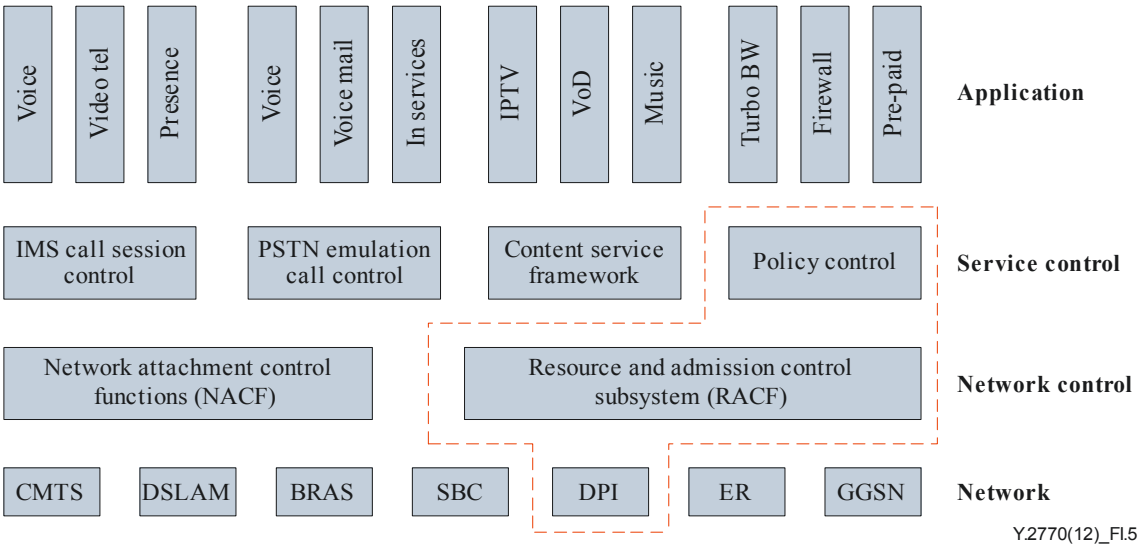


Figure I.5 – Example of DPI's application scenario within the context of NGN [b-ITU-T Y.2012]

Figure I.6 illustrate NGN as a layered architecture. DPI resides in packet forwarding plane, while resource and admission control is a basic function of control plane, with policies being stored in the policy repository. Whenever a new application is detected, DPI will generate a request for resource demand and deliver it to the control plane for resource allocation, even in some cases, DPI can be used to reroute the packets to satisfy SLA requirement.

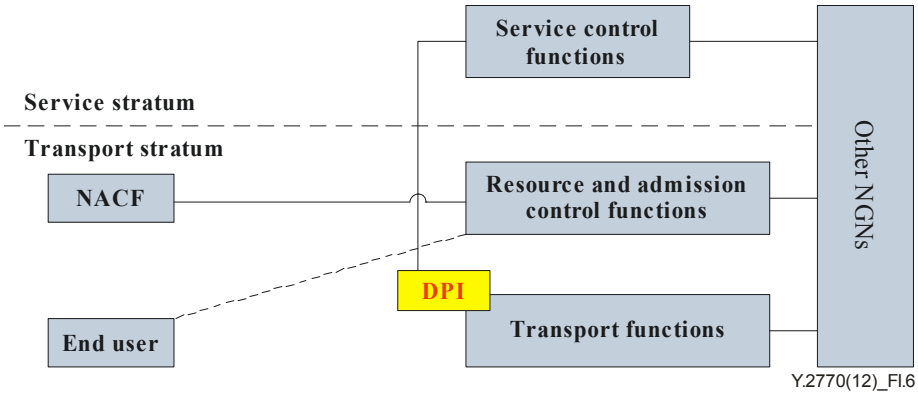


Figure I.6 – Example of DPI's operation within the environment of NGN

Figure I.6 illustrates example locations of DPI policy rule enforcement, e.g., customer premises equipment (CPE) - or host-based on customer premises or network based DPI.

In the scenario above, when DPI is used in the scenario for resource allocation, there are some standardized interfaces within the network architecture. Resource reservation request originated from DPI will be generated whenever a new service is identified. For example, bandwidth allocation and QoS guarantee for a real-time VoIP application. Under such circumstances, DPI is deployed to fulfil the following goals:

1) Monitor network and bandwidth usage. DPI is applied to automatically discover the application and determine the protocol that might affect the network performance and bandwidth usage.

2) Define the policies in accordance with the identified application. Policies can be seen as the tie between application and resource requirements, which in turn determines the QoS attributes of applications, among them are: minimum and maximum bandwidth, traffic prioritization, etc.

3) Enforce the policy and make the policy repository up-to-date at any time.

From the perspective of NGN, DPI is applied to identify the application and generate the raw resource demand; all the remaining processing, including message triggering, delivery and processing will follow the procedures as defined in section 9, ITU-T Recommendation of [ITU-T Y.2111].

As part of application scenario, how DPI components are installed in the network is a big concern, whether it is deployed as in-line mode or by-pass mode, and what functions DPI can fulfil are something that also should be mentioned.

As shown in Figure I.7, a DPI component can be deployed in a network either as an inline device or a bypass device; it depends on the operator's purpose in using them.

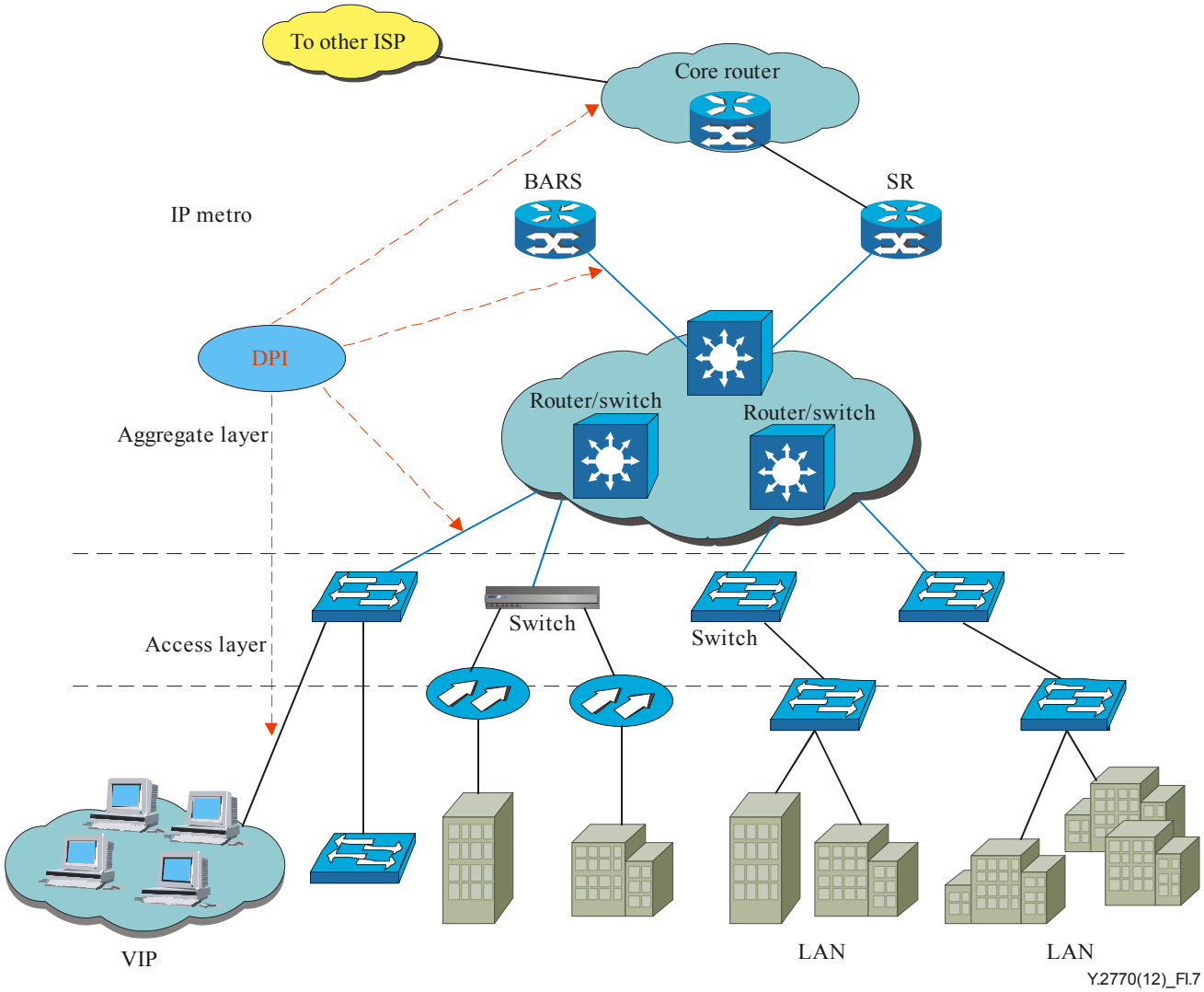


Figure I.7– Scenario of DPI's Deployment in packet-based network

I.3.1 DPI used as a bidirectional tool for service control

Example use case:

Bidirectional DPI is used to monitor closely related traffic in opposite directions. Under such circumstances, DPI FE needs first to set up the association between the opposite traffic, search the rule table to retrieve the appropriate entry and process the rules accordingly.

With its rule tables and bidirectional DPI signatures, DPI can act on both incoming and outing direction on bidirectional traffic.

See clause 6 concerning correspondent requirements for uni- and bidirectional DPI.

I.4 DPI use case: Network- versus Link-oriented DPI

I.4.1 Overview

The DPI signatures (of DPI policy rules) may cover protocol layer 3 and upwards or start already with protocol layer 2 (see clause 3.2.5), which may be distinguished in network-oriented DPI and link-oriented, see Figure I.8. The crucial point relates to the fact that layer 2 information is limited, either to a point-to-point link or to the borders of layer 2 network domain.

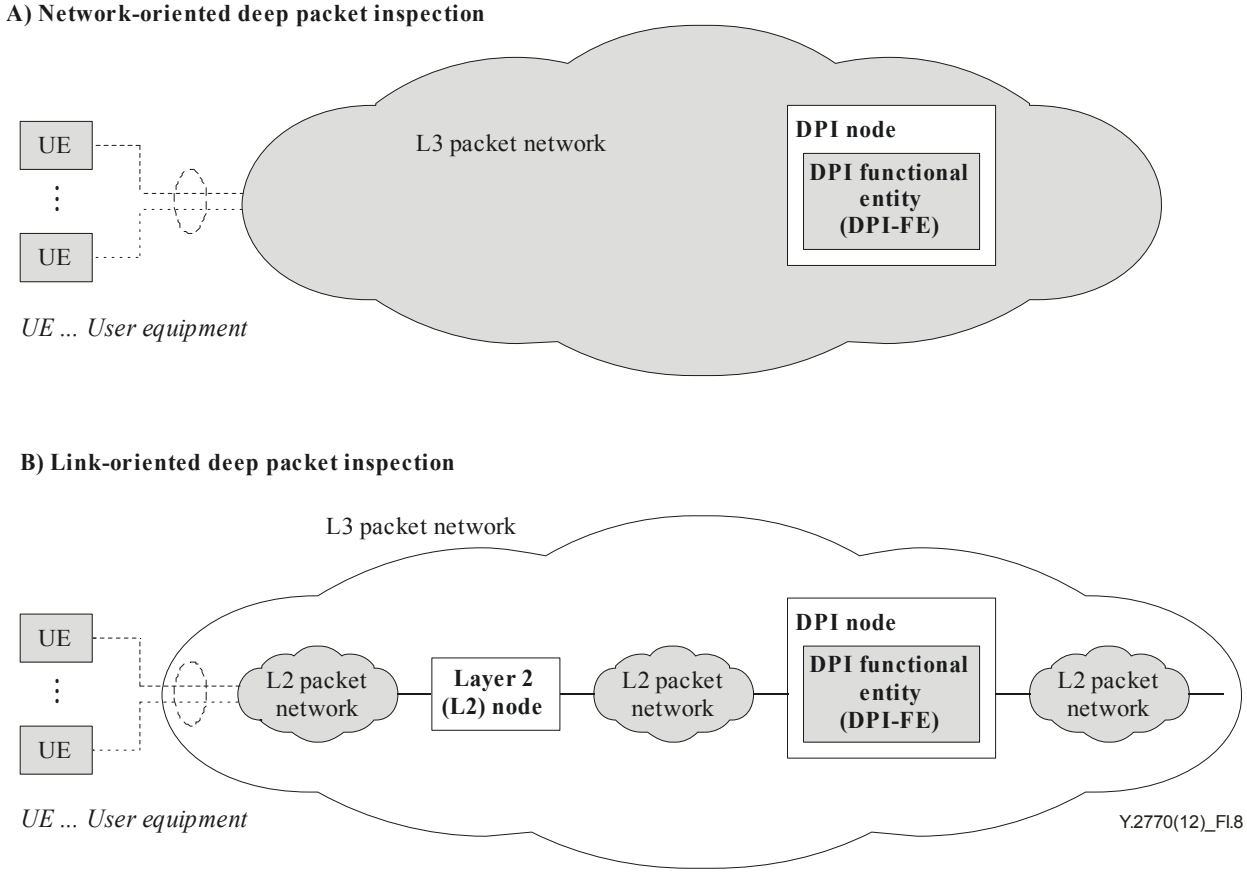


Figure I.8 – Network- vs. Link-oriented DPI

I.4.2 Link-oriented DPI

Link layer protocol control information (PCI) is limited to a L2 network domain, and may consequently change in the end-to-end communication path. Usage of L2-PCI (e.g., ATM VCI, Ethernet Destination Address) as part of DPI signatures may thus question the applicability of such DPI policy rules. The policy decision entity must also be aware of the underlying layer 2 network infrastructure.

However, there are use cases for Link-oriented DPI like for instance:

- end-to-end network relates to a single layer 2 network; or
- L2-VPN (layer 2 virtual private network) with L2-VPN dedicated DPI policy decisions.

L2 network domain specific DPI should be thus supported by DPI functional entities.

I.4.3 Network-oriented DPI

Network-oriented DPI is related to DPI signatures which cover protocol information on network layer (L3) and higher. The L2 PCI (e.g., L2 header, padding) is removed before the DPI-FE. Network-oriented DPI represents a common case for DPI due to the "end-to-end" relevance of the network layer. Which means: there would be *not* any *dependency on the location* of the DPI-FE within the end-to-end packet path, the results would be the same.

NOTES –

a) There are scenarios with L3-PCI modifications between end nodes, e.g., the application of topology hiding (THIG) implies changes of L3 topology information (e.g., NAT

(network address translation) in IPv4 networks), L3-VPN (layer 3 virtual private network).

b) The model in clause I.4 considers the simplest case of a flat protocol stack. There might be however hierarchical (nested) protocol stacks in real networks like tunnelling methods (e.g., MPLS, IPv4-over-IPv6, Generic Routing Encapsulation (GRE, [b-IETF RFC 2784]), GPRS tunnelling in mobile access networks), which may lead to "L3-over-L3" packet types. Such protocol encapsulation is principally covered by the DPI definition (see clause 3.2.5), but not further detailed in this Recommendation.

I.5 DPI use case: Traffic control

DPI may contribute to traffic control functions as part of packet network infrastructures.

I.5.1 Overview of traffic control functions

The following functions are identified for traffic control, e.g., according [ITU-T Y.1221]:

i) Network resource management.

ii) Admission control.

iii) Parameter control, e.g., traffic policing for usage parameter control.

iv) Packet marking.

v) Traffic shaping.

vi) Packet scheduling.

DPI may contribute to traffic control functions in various ways, particularly when the "traffic granularity" would be related to "application traffic", and the implication of DPI-based application identification as part of the traffic control function.

Some use cases.

I.5.2 DPI-based shaping of application traffic

Example use case: traffic shaping may be applied with the purpose in reducing or limiting the packet delay variation of a packet flow. Such a shaping function relates therefore to a specific action, executed on individual packets of the considered traffic flow. However, such a function implies firstly the correct identification of the packets, which would be a DPI-based application identification stage.

Example: shaping of traffic of a distributed, multiuser gaming protocol. Such a shaping function might be beneficial from an overall traffic engineering view for a packet network, and the shaping function would not violate any potential realtime service requirements of the communication.

I.5.3 DPI-based policing of peer-to-peer traffic

Example use case: peer-to-peer users may issue uncontrolled best-effort IP traffic to access network domains. The usage of transport capacity shall be limited per peer-to-peer service ("DPI application identification") and per peer user ("DPI flow identification") via a traffic policer instance, which may police characteristic traffic parameters (e.g., packet rate, byte rate).

I.5.4 DPI-based marking of specific packet types

Example use case: a packet network may provide QoS support by different handling of traffic. The different classes of traffic may be discriminated by a correspondent protocol element in the packet header. The DPI function may be requested to mark packets, of "*application X*", with a specific value.

I.6 DPI use case: Detection of abnormal traffic

I.6.1 Background

A particular packet flow may be characterized by some *traffic parameters*, given as statistical metrics which are derived from probability distribution functions, e.g., the packet inter-arrival time, arrival order, the size of the PDU of a specific protocol layer, the size of payload, or the traffic volume (at a specific protocol layer). Many statistics may be derived and used for the characterization of "traffic" in general (NOTE 1). Normal traffic may be characterized by a correspondent set of such statistical metrics (NOTE 2). Abnormal traffic would be then given by traffic which may not be associated with normal traffic classes.

The DPI signatures may contain an (flow) identifier related to "characteristics of the packet itself" (see clause 3.1.3). Such an identifier type characterizes normal packet traffic from perspective of the DPI-FE.

NOTE 1 – An example of a concrete traffic descriptor is, e.g., defined for IP traffic by clause 3.2.10 in [ITU-T Y.1221].

NOTE 2 – When the term 'traffic' is related to a specific protocol, then the set of traffic metrics is also known as protocol fingerprint, which may be, e.g., constructed from a set of flows of the same protocol (see [b-IEEE GLOBECOM]).

NOTE 3 – [b-ITU-T X.abnot] outlines a possible application related to abnormal traffic, and mentions also explicitly the possible involvement of DPI functions in such scenarios.

I.6.2 Example use cases

Below subclauses provide more detailed use case information. The underlying high-level DPI use case scenario could be in the area of traffic monitoring, security support, usage parameter monitoring, etc.

I.6.2.1 Simple use case: Block packets which carry data attachments of specific sizes

MIME objects may be embedded in many IP application protocols (e.g., HTTP, SMTP, SIP, MSRP, etc.). The category of normal traffic would be comprised of MIME attachments with a byte size within a particular range. Abnormal traffic would be blocked, i.e. IP packets carrying MIME attachments (at any layer above L3) outside that range would be discarded.

This is a simple use case because the correspondent DPI policy rule is fairly short.

I.6.2.2 Complex use case: Scenarios with statistical behaviour analysis

Statistical behaviour analysis (see [b-IEEE GLOBECOM]) is a useful DPI method whenever deterministic inspection methods are either not applicable or not economical. This DPI method may be principally applied for many DPI use cases described in this Appendix.

I.7 DPI use case: Example concerning statistical versus deterministic packet inspection methods

The packet inspection process of a DPI-FE may be described by DPI policy rules. The policy conditions would indicate the identification part for incoming packets. Such an identification approach may be fundamentally following either deterministic or statistical principles. The usage of heuristics is an example for statistical packet inspection. The notion of heuristics summarizes DPI packet identification methods with respect to

- limited available information ("e.g., only partial knowledge about the "application traffic" or even unknown traffic");

- limited resources ("e.g., amount of CPU cycles by the DPI-FE for packet inspection"); or/and
- economical trade-off decisions ("e.g., reduce the length of a search pattern").

For instance following use case where DPI may be part of an application scenario related to detection methods for encrypted network traffic:

A DPI-FE may be requested for distinguishing IPsec ESP-NULL (Encapsulating Security Payload without encryption) packets from encrypted ESP packets. There are two defined DPI-based methods:

- [b-IETF RFC 5879] relates to a *statistical* identification method, based on heuristics;
- [b-IETF RFC 5840] relates to a *deterministical* identification method, based on protocol extensions.

I.8 DPI use case: Example concerning packet modification

In a typical DPI application scenario, it is sometimes necessary that the original packet header and/or payload be modified for the purpose of QoS mapping, removing virus, etc.

I.8.1 DPI use case: Modification of packet header information

I.8.1.1 Background

The term "*packet header*" represents in general *Protocol Control Information* (PCI) data, which may, e.g., also cover "packet trailer" data. Packet header information may be classified in some principal categories. Table I.1 provides an example.

Table I.1 – Principal packet header information categories

Information category:	Example header elements:
1) Address information	• network addresses, • transport addresses
2) Lx-VPN information	• any elements used for VPN identification at a particular protocol layer
3) QoS class information	• IPv4 Type-of-Service (ToS) field, • IPv4/v6 Differentiated Service (DS) field, • IPv6 Traffic Class (TC) field
4) Congestion information	• IPv4 Explicit Congestion Notification (ECN) field
5) Assured transport information	• checksums
6) Others	• …

Not all categories are subject of DPI use cases. It may be further noted that the set of possible actions related to "packet header modification" operations may be split into two categories (from user perspective), like "intrusive actions" and "non-intrusive actions":

- intrusive DPI action: the user could be negatively impacted, e.g., due to service disruption;
- non-intrusive DPI action: the user should typically benefit from such actions, e.g., due to enhanced service quality, due to improved network connectivity (by traversal or bypassing of "blocking nodes"), due to active congestion management, etc.

I.8.1.2 Use case: Modification of IPv4 ToS field

A network often contains many different kinds of services, and each of those services is often attributed with some certain kinds of characteristics reflected by its packet headers. When those packets travel along their forwarding path from source to destination, their header information is often needed to be modified to reflect the services' characteristics. For example, when a packet originated from an IPv4 network forwarded along its path, its IPv4 Type-of-Service (ToS) field is often needed to be modified according to the provisioned mapping mechanism.

I.8.1.3 Use case: Modification of IPv6 TC field

Similar network scenario as in clause I.8.1.2, but modification of 8-bit IPv6 Traffic Class (TC) field instead of IPv4 ToS field.

I.8.2 DPI use case: Modification of packet payload

I.8.2.1 Background

Complementary to modification of packet header information, DPI can also be deployed to do packet payload modification:

- removing viruses (see below sub-clause I.8.2.2);
- transforming content to support gateway functions, e.g., DPI desirable to modify information elements in packet payload carrying IP address information, such as IPv4-to-IPv6 transitioning;
- solving problems related to the rapid pace of change in service protocols, e.g., DPI need to modify packet payload to enable different SIP entities with different capabilities to communicate with each other;
- other packet payload modification, etc.

I.8.2.2 Use case: Removing viruses

The vulnerability of any packet-based network makes it possible that a packet might be contaminated with virus when it is forwarded along its path. In some cases, the packet has to be delivered even though it contains virus as dropping it might introduce some negative impacts on the quality of experience (QoE) of its end users. Under such circumstances, to modify the payload of the packet by removing the virus is desirable.

I.9 DPI use case: Example concerning DPI engine capabilities

I.9.1 Background

The typical relationship of a DPI engine and DPI physical entity is illustrated in Figure I.8 (see the red dashed box). Figure I.8 illustrates the indication of a DPI engine at an example functional model of a DPI-FE. The specific model is not essential here, but described in more detail in Appendix III. Following functional grouping levels are used here:

- *DPI engine* focuses on functions related to DPI scanning, DPI analysing and DPI action execution; and
- *DPI policy enforcement function* (DPI-PEF) summarizes the DPI engine supported functions plus a DPI policy information base here.

This means that the DPI engine provides all packet processing path functions.

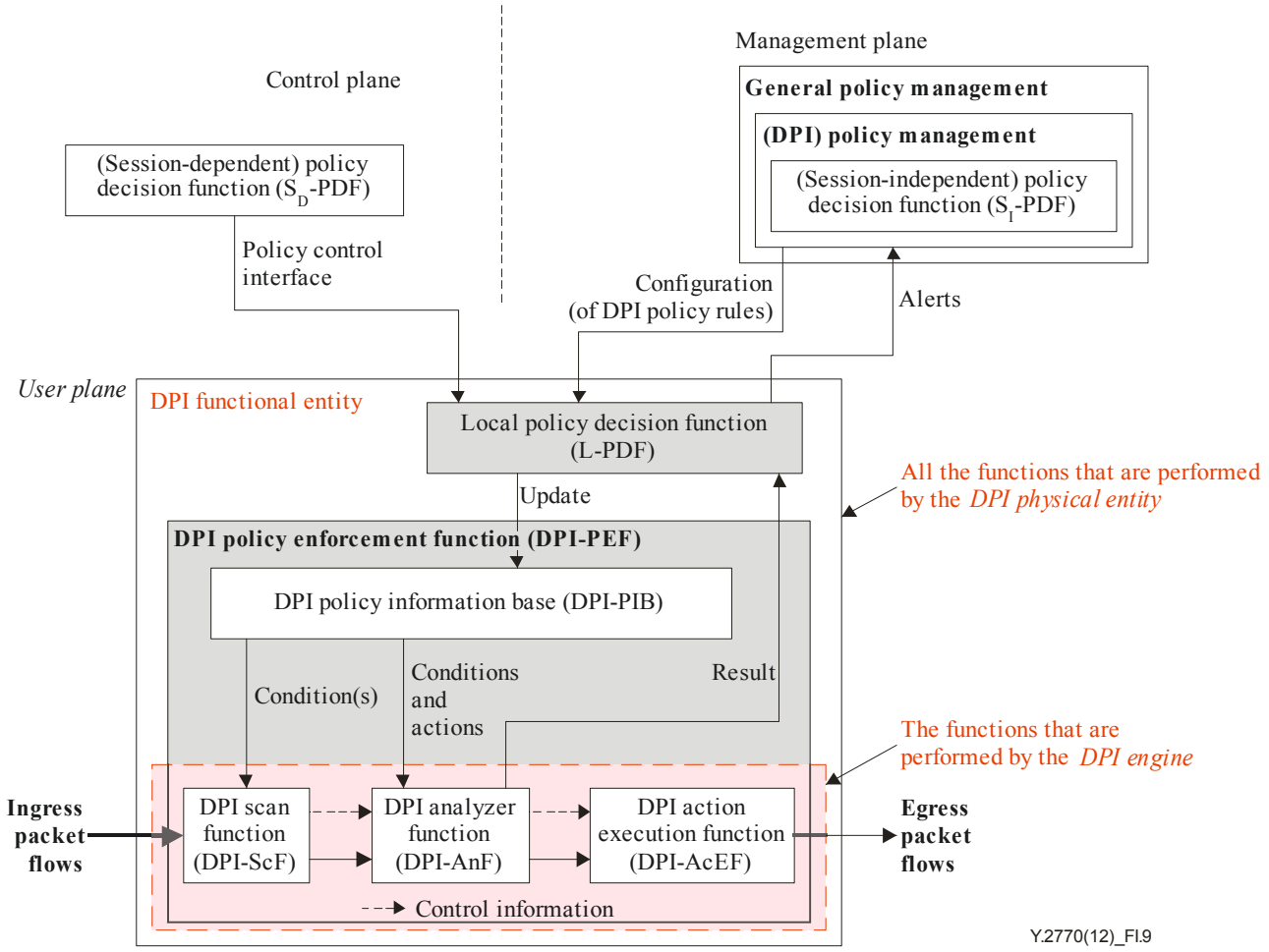


Figure I.9 – The typical relationship of a DPI engine and DPI physical entity

Since the DPI engine does not include the DPI policy information base, the DPI signatures for application (and optional) flow identification would be (also) available in processing elements of the DPI engine. There are some typical DPI signature formats such as:

1) Simple fixed string: A string P of m bytes can be written as $P = b_1b_2\ldots b_m$, where each b_i represents a byte. Such a string type is also known as byte patterns.

2) Composite Patterns:

   a) Negation (!).The notation "!P" is used for the specification of "no appearance of pattern P".

   b) Correlated patterns. If P1 and P2 are two patterns, P3 = P1 + P2 is a correlated pattern, with the meaning that P3 is the concatenation of P1 and P2.

3) Regular Expressions: A regular expression describes a set of strings without enumerating them explicitly. Regular expressions are widely used for pattern matching due to their rich expressive power.

Thus, the DPI engine focuses on DPI application level conditions verification with all these signature formats to do such: simple fixed string matching, composite patterns matching and regular expression matching.

I.9.2 DPI engine use case: Simple fixed string matching for *BitTorrent*

See example DPI policy rule according clause II.4.16. That rule relates to a DPI signature format "simple fixed string" according above introduction. DPI engine need to do simple fixed string matching 20 bytes long in order to identify *BitTorrent*.

APPENDIX II

DPI policy rules examples for packet inspection

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

II.1 Introduction

II.1.1 Purpose

Accompanying material for discussion of DPI use cases and requirements.

II.1.2 Specification level of rules

Packet inspection may be considered as a packet *policing function* (see Appendix VII). The particular "inspection function" may be thus formulated as policy rule. The specification depth may differ in various respects:

- High-level
- Low-level rules (using informal specification language, like a prose language)
- Low-level rules (using formal specification grammar)

II.1.3 Generic rule format

In order to use a common description format, the examples here following below generic rule format, comprised of

- a *rule header* (name, identifier, precedence, etc.) and
- a *rule body* (for conditions, actions).

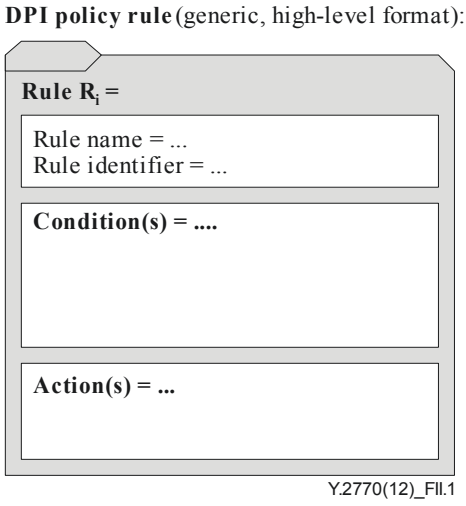


Figure II.1 – Generic rule format

It may be noted again that the specification of *explicit bindings* between actions and conditions is *out* of scope of this Recommendation.

II.2 Example policy rules for Application-dependent, Flow-dependent DPI – Identification order "*1st Application, 2nd Flow*"

II.2.1 Example "Security check – Block SIP messages with specific content types and derive SIP device address"

Table II.2.1 – Example

DPI Policy Rule ("**Security check – Block SIP messages with specific content types and derive SIP device address**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ PDU = SIP message?" AND C_2: "SIP content-type values = {…}"	Then: A_1: "extract flow information (which contains the remote source IP transport address) and …" A_2: "… SIP message" A_3: "Update Statistic"

II.2.2 Example "Detection of Malware"

Here: network-based detection of malicious software ('malware'), which is typically distributed via FTP and HTTP (or SMTP), in contrary to host-based malware detection.

Assumption: a pattern matching technique is sufficient for detecting particular malware. The characteristic patterns of malwares are registered beforehand as signatures, and it detects malwares by comparing these patterns with byte code of a malware. Pattern matching technique is effective for existing malwares whose signature has been created.

Table II.2.2 – Example

DPI Policy Rule ("**Detection of Malware**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ PDU = FTP message?[3]" OR C_2: "L4+ PDU = HTTP message?" AND C_3: "L4+ PDU contains byte pattern "*xyz*[4]""	Then: A_1: "extract flow information (which contains the remote source IP transport address) and …" A_2: "report detected Malware type"

3 The condition here is very abstracted. There would be a dedicated set of conditions for FTP detection behind in reality. See e.g. clause II.2.4.

4 The Malware may be classified using a standardized identification and naming scheme, like e.g. MAEC http://maec.mitre.org ("MAEC is a standardized language for encoding and communicating high-fidelity information about malware based upon attributes such as behaviours, artefacts, and attack patterns.")

II.2.3 Example "Detection of specific video format"

Here: H.264 as example. Rule may be further refined for checking specific subformats.

Table II.2.3 – Example

DPI Policy Rule ("**Detection of specific video format**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "look for video format H.264 by checking for H.264 header information" AND C_2: "packet matches Flow Descriptor = FD "	Then: A_1: "…"

II.2.4 Example "Detection of File Transfer in general"

Here: FTP/TCP-based file transfer, thus an example of an IP application with two (or >2) "flow components", one control flow and "0 to many (parallel)" data flows (separate TCP connections for individual file transfers).

The assumption here: the overall flow level conditions are given by just the 2-tuple of IP connection endpoint addresses (of FTP client and FTP server).

Table II.2.4 – Example

DPI Policy Rule ("**Detection of File Transfer in general**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**

State S1 ("inspect for ftp control traffic"):
If:
C_1: "IP protocol type = "TCP""
AND
{
C_2: "L4 port = «well-known port for ftp control»"
OR
C_3: "L4+ PDU contains FTP Commands …"
}
	Then:
	A_1: "extract flow information (the 2-tuple of *f* = {IP Source Address (SA), IP Destination Address (DA)})"

State S2 ("inspect for ftp data traffic"): If: C_1: "Flow Descriptor = FD (or inverse due to both traffic directions)" AND C_2: "IP protocol type = "TCP"" AND C_3: "L4+ PDU contains FTP DATA header information elements (e.g., *filename*, *file lists*."	Then: A_1: "extract *file name(s)* information and report to …"

That's an example of stateful DPI, abstracted here to just two states for control traffic detection and file traffic detection. The initial state (S1) belongs to the "flow independent" identification category (F_I) and subsequent state S2 is then acting in "flow dependent" identification category (F_D) mode.

II.3 Example policy rules for Application-dependent, Flow-dependent DPI – Identification order "*1st Flow, 2nd Application*"

II.3.1 Example "Security check – Process SIP messages (from a particular user) with specific content types – User identification via flow information"

Assumption: the SIP user may be associated with a particular SIP UA, and the IP transport address of that SIP device is sufficient for user identification.

Table II.3.1 – Example

DPI Policy Rule ("**Security check – Block SIP messages (from a particular user) with specific content types – User identification via flow information**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet matches Flow Descriptor = FD" AND C_2: "L4+ PDU = SIP message?" AND C_3: "SIP content-type values = {…}"	Then: A_1: "... SIP message" A_2: "Update Statistic"

II.3.2 Example "Application-specific traffic policing"

Note: a traffic policing approach beyond, e.g., the flow-level IP byterate policing.

Table II.3.2 – Example

DPI Policy Rule ("**Application-specific traffic policing**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet matches Flow Descriptor = FD" AND C_2: " check whether application-level traffic volume > *x* bytes"	Then: A_1: "... packet …"

II.3.3 Example "Business Card (vCard) application – Correlate Employee with Organization"

NOTE – Monitor email traffic with attached business cards.

Table II.3.3 – Example

DPI Policy Rule ("**Correlate Employee with Organization**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet matches Flow Descriptor = FD " AND C_2: "SMTP packet " AND C_3: "SMTP body contains vCard"[5] AND C_4: "vCard contains "`N:`" AND "`ORG:`" element"	Then: A_1: "extract *name* and *organizational* information, and if available *role* and *title* information; report all information to …"

II.3.4 Example "Forwarding copy right protected audio content"

… by checking on embedded digital watermarks in MP3 data.

NOTE – Stateful DPI policy rule because digital watermark may be distributed across multiple, consecutive RTP packets of the same RTP session

Table II.3.4 – Example

DPI Policy Rule ("**Forwarding copy right protected audio content**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet matches Flow Descriptor = FD " AND C_2: "L4+ = RTP packet" AND C_2: "RTP payload contains "*MPEG-x, layer III audio* ("MP3")" media"[6] AND C_2: "media contains digital watermark *W*"	Then: A_1: "forward packet"

II.3.5 Example "Measurement-based traffic control"

NOTE – There are legacy DPI policy rules, e.g., to discard every packet after the X^{th} received, or discard every packet after a certain traffic volume in terms of bytes is reached, or etc. etc. Such rules may be generalized: they are based on a specific local statistic, which represents a "state condition". Thus, such DPI policy rules contain stateful actions.

5 E.g., search for string "`BEGIN:VCARD`"

6 The RTP-SDU would be checked for ADU frames "Application Data Unit", and ADU's for embedded MP3-frames.

Table II.3.5.1 – Example

DPI Policy Rule ("**Measurement-based traffic control**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet matches Flow Descriptor = FD" AND C_2: "packet matches Application Descriptor = AD"	Then: A_1: "update statistic S" AND A_2: "discard packet *"if statistic S > x'"*"

The 2nd action contains an embedded condition. There are other options for structuring such a rule. E.g., by such a rule format:

Table II.3.5.2 – Alternative rule structure

DPI Policy Rule	
Conditions	**Actions**
If: C_1: "packet matches Flow Descriptor = FD" AND C_2: "packet matches Application Descriptor = AD "	Then: A_1: "update statistic S"
Post-policy Rule	
Conditions	**Actions**
If: C_3: "statistic S > x"	Then: A_2: "discard packet"

II.3.6 Example "Detection of a specific transferred file from a particular user"

That's a modification of the ftp example from clause II.2.4. The flow level conditions may be given and more detailed, dependent on available user information. The policy conditions related to application identification may require detailed application level conditions, such as FTP application payload of control commands and/or data with regards to, e.g., a specific file name.

II.4 Example policy rules for Application-dependent, Flow-independent DPI

II.4.1 Example "Security check – Block SIP messages (from a particular user) with specific content types – User identification via application information"

Assumption: the SIP user may be associated with a particular SIP UA, and the SIP From header information is sufficient for user identification. This is a flow-independent scenario.

Table II.4.1 – Example

DPI Policy Rule ("**Security check – Block SIP messages (from a particular user) with specific content types – User identification via application information**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ PDU = SIP message?" AND C_2: "SIP From header = …" AND C_3: "SIP content-type values = {…}"	Then: A_1: "... SIP message" A_2: "Update Statistic"

II.4.2 Example "Security check – Block SIP messages (across entire SIP traffic) with specific content types"

Table II.4.2 – Example

DPI Policy Rule ("**Security check – Block SIP messages (across entire SIP traffic) with specific content types**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ PDU = SIP message?" AND C_2: "SIP content-type values = {…}"	Then: A_1: "... SIP message" A_2: "Update Statistic"

II.4.3 Example "Checking resource locators in SIP messages"

Table II.4.3 – Example

DPI Policy Rule ("**Checking resource locators in SIP messages**") R_x: "xyz", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "SIP From header = URI_X"	Then: A_1: "... Message" A_2: "Update Statistic" A_3: "Alert Alarm Management"

II.4.4 Example "Deletion of a particular audio channel in a multi-channel media application"

Table II.4.4 – Example

DPI Policy Rule ("Identify 2nd audio channel in a multi-channel AMR application") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "Packet contains the hexadecimal string = "0x2321414d525F4D43312E300a""[7] AND C_2: "next 32-bit word contains the hexadecimal string = "0x00000002""[8]	Then: A_1: "... Audio Frame within SDU" A_2: "Update Statistic"

II.4.5 Example "Identify particular host by evaluating all RTCP SDES packets"

Table II.4.5 – Example

DPI Policy Rule ("Identify particular host by evaluating all RTCP SDES packets") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "RTCP SDES packet contains SDES item"" AND C_2: "CNAME syntax = "user@host" OR "only host"" AND C_3: "*host* element is encoded in FQDN syntax" AND C_4: "CNAME value = "xyz""	Then: A_1: "send alert"

II.4.6 Example "Measure Spanish Jabber traffic"

DPI use case: counting Jabber messages with Spanish text.

7 = ASCII character string: "#!AMR_MC1.0\n" (see [b-IETF RFC 4867], the *magic number* for multi-channel AMR)

8 = 32 bit channel description field(see [b-IETF RFC 4867])

Table II.4.6 – Example

DPI Policy Rule ("Measure Spanish Jabber traffic") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ header type = XMPP stream header" AND C_2: "XMPP message content contains XML element = "`xml:lang='es'`"	Then: A_1: "update statistic …"

II.4.7 Example "Blocking of dedicated games"

… of category "distributed realtime games with OGP for communication".

Table II.4.7 – Example

DPI Policy Rule ("Blocking of dedicated games") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ header type = OGP header" AND C_2: "OGP content contains element *GameName* = xyz"	Then: A_1: "silently ... IP packet"

II.4.8 Example "Statistics about Operating Systems of game consoles"

… of category "distributed realtime games with OGP for communication".

Table II.4.8 – Example

DPI Policy Rule ("Blocking of dedicated games") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ header type = OGP header" AND C_2: "OGP content contains *ServerFlags* = xyz"	Then: A_1: "update statistic for OS type (Linux, Windows, Mac, Dedicated or Proxy)"

II.4.9 Example "Measure abnormal traffic with respect to packet sizes"

In general: there is a plethora of metrics associated to the characteristics of Lx-PDUs as such, or Lx-PCI (header) and Lx-SDU (payload). One key metric is 'size', with associated statistics like minimum, mean, maximum or a distribution function in general. If the metric is related to a protocol layer above flow identification, then we got application dependent DPI.

E.g., check for too large instant messages:

Table II.4.9 – Example

DPI Policy Rule ("**Measure abnormal traffic with respect to packet sizes (here MSRP)**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ protocol type = MSRP" AND C_2: "L4-SDU size > x"	Then: A_1: "update statistic"

II.4.10 Example "Detect abnormal MIME attachments in multiple application protocols"

Example of generic type "MIME over message protocol X".

Table II.4.10 – Example

DPI Policy Rule ("**Detect abnormal MIME attachments in multiple application protocols**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ protocol type $\subset$ {HTTP, SIP, SMTP, …}" AND C_2: "L4 message content contains MIME attachment" AND C_3: "MIME attachment size > x"	Then: A_1: "update statistic"

II.4.11 Example "Identify uploading *BitTorrent* users"

NOTE – *BitTorrent* is a peer-to-peer file sharing protocol.

Table II.4.11 – Example

DPI Policy Rule ("**Identify uploading *BitTorrent* users**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ protocol type = HTTP" AND C_1: "http control = GET request header" AND C_2: "http message content contains element *uploaded* with a value > 0"	Then: A_1: "extract element *peer_id* from http message and …"

II.4.12 Example "Measure *BitTorrent* traffic"

… by identifying to particular file types on occurrence of BENcode.

Table II.4.12 – Example

DPI Policy Rule ("**Identify uploading *BitTorrent* users**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "packet content contains *bencoded* strings" OR C_2: "packet content contains *bencoded* integers" OR C_3: "packet content contains *bencoded* lists" OR C_4: "packet content contains *bencoded* dictionaries"	Then: A_1: "update statistic …"

II.4.13 Example "Blocking Peer-to-Peer VoIP telephony with proprietary end-to-end application control protocols"

Here an example which supposes a condition, specified at a higher hierarchy as regular expression.

Table II.4.13 – Example

DPI Policy Rule ("**Blocking Peer-to-Peer VoIP telephony …**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "*regular expression which describes all the conditions which must be true*"	Then: A_1: "block IP packet"

II.4.14 Example "Specific handling of old IP packets"

DPI use case: specific handling (e.g., counting) of IP packets with "expired lifetime".

Table II.4.14 – Example

DPI Policy Rule ("**Specific handling of old IP packets**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "IP header "TTL" < *X*" AND C_2: "*application specific condition …*"	Then: A_1: "…e.g., update statistic …"

II.4.15 Example "Security check – SIP Register flood attack (using a SNORT rule)"

That example illustrates how the SNORT rule format looks like in the generic rule format of this document.

Just one example (out of hundreds from www.snort.org):

```
### FLOODING BY SIP MESSAGES #####
#e.g., Rule for alerting of REGISTER flood attack:
alert ip any any -> $SIP_PROXY_IP $SIP_PROXY_PORTS \
(msg:"REGISTER message flooding"; content:"REGISTER"; depth:8; \
threshold: type both, track by_src, count 100, seconds 60; \
sid:5000005; rev:1;)
```

Below tables indicates the transformed rule.

Table II.4.15 – Example

DPI Policy Rule ("**Security check – SIP Register flood attack**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "for any IP transport connection (i.e., flow independent" AND C_1: "L4+ PDU = SIP message?" AND C_2: "look for SIP header 'REGISTER' in first eight bytes of message"	Then: A_1: "store flow related information (e.g., IPFIX Flow Identifier)" AND A_2: If threshold condition ("*more than 100 REGISTER's from same source during one minute*") is valid, then send alert "*`REGISTER message flooding`*"

II.4.16 Example "Detection of *BitTorrent* traffic"

This example is based on a DPI signature proposal by [b-Subhabrata] and characterized by a pattern (here: patterns relates to specific byte values) at a *known location* in the packet payload):

The communication between the *BitTorrent* clients starts with a handshake followed by a never-ending stream of length-prefixed messages. The *BitTorrent* header of the handshake messages assumes following format: *<a character (1 byte)> <a string (19 byte)>*.

The first byte is a fixed character with value '19', and the string value is '*BitTorrent protocol*'. Based on this common header, the following signature (see also Table II.4.16) is used for identifying *BitTorrent* traffic:

- The first byte in the TCP payload is the character 19 (0x13).
- The next 19 bytes match the string '*BitTorrent protocol*'.

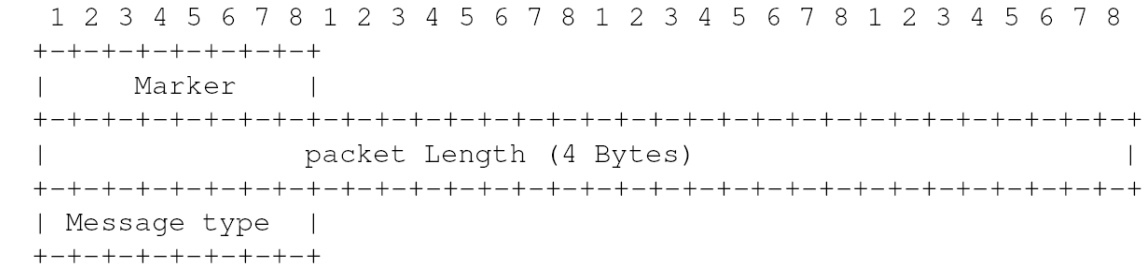
Table II.4.16 – Example

DPI Policy Rule ("**Detection of *BitTorrent* traffic**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ protocol type = TCP" AND C_2: "1st byte in TCP payload = 0x13" AND C_3: "next 19 bytes in TCP payload = '*BitTorrent protocol*'"	Then: A_1: "…"

II.4.17 Example "Detection of *eDonkey* traffic"

This example is based on a DPI signature proposal by [b-Subhabrata] and characterized by the *length* of certain part of a packet:

- *eDonkey* packets, both signalling and downloading TCP packets have the following common *eDonkey* header directly following the TCP header:



- where the *marker* value is always 0xe3 in hex, the *packet length* is specified in network byte order and the value is the byte length of the content of the *eDonkey* message excluding the marker 1 byte and the length field 4 bytes. Utilizing these discoveries, the following signature is used for identifying *eDonkey* packets.
- For TCP signalling or handshaking data packets, two steps to identify *eDonkey* packets (see also Table II.4.17):
  - The first byte after the TCP header is the *eDonkey marker*.
  - The number given by the next 4 bytes is equal to the size of the entire packet after excluding both the IP and TCP header bytes and 5 extra bytes.

Table II.4.17 – Example

DPI Policy Rule ("**Detection of *eDonkey* traffic**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ protocol type = TCP" AND C_2: "TCP packet types relates to signalling or handshaking data" AND C_3: "1st byte in TCP payload = 0x03" (the *eDonkey marker value*) AND C_4: "Value of bytes 2 to 5 of TCP payload = "the size of the TCP payload minus 5 extra bytes""	Then: A_1: "…"

II.5 Example policy rules for mixed ("stateful") Application-dependent, Flow-independent/Flow-dependent DPI

II.5.1 Example "Detecting a specific Peer-to-Peer VoIP telephony with proprietary end-to-end application control protocols"

Example for a bidirectional, stateful DPI policy rule. Note: the rule itself illustrates just the principal proceeding, but is not necessarily sufficient for a real deployment.

Table II.5.1 – Example

DPI Policy Rule ("**Detect TELEPHONYSERVICEX session establishment**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
State S1: If: $C_{1,1}$: "L4 protocol = UDP" AND $C_{1,2}$: "L4 payload size = 18" AND $C_{1,3}$: "3rd payload byte = 0x02"	Then: $A_{1,1}$: "save source IP transport address" $A_{1,2}$: "save destination IP transport address" $A_{1,3}$: "save first two bytes of L4 payload" Comment: the flow descriptor are based here on the 5-tuple for the UDPoIP transport connection and locally stored.
State S2: If: $C_{2,1}$: "Flow Descriptor in *reverse* direction = … (see $A_{1,1}$ / $A_{1,2}$)" AND $C_{2,2}$: "L4 payload size = 11" AND $C_{2,3}$: "first two bytes of L4 payload = … (see $A_{1,3}$)" AND $C_{2,4}$: "3rd payload byte: lower nibble = 7"	Then: none
State S3: If: $C_{3,1}$: " Flow Descriptor in *initial* direction = … (see $A_{1,1}$ / $A_{1,2}$)" AND $C_{3,2}$: "L4 payload size = 23" AND $C_{3,3}$: "first two bytes of L4 payload = … (see $A_{1,3}$)" AND $C_{3,4}$: "3rd payload byte: lower nibble = 3"	Then: none
State S4: If: $C_{4,1}$: " Flow Descriptor in *reverse* direction = … (see $A_{1,1}$ / $A_{1,2}$)" AND $C_{4,2}$: "L4 payload size = 18" AND $C_{4,3}$: "3rd payload byte = 0x02"	Then: $A_{4,1}$: "report "successfully TELEPHONYSERVICEX detected"

II.6 Examples of multiple, different DPI policy rules for the same DPI application

II.6.1 Example "Detection of *Remote Telnet*"

The DPI policy rule may request to inspect for the (IP) application *Remote Telnet*. There are then two options:

1) the application identification may be solely based on a rule with just the *flow level conditions* if a well-known port is used (see Table II.6.1); or

2) another rule: an additional *application level conditions* would be required if application may not be derived from flow level condition (see Table II.6.2).

Table II.6.1 – Example

DPI Policy Rule ("**Detection of *Remote Telnet* based on well-known port value**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4 Port = 107"	Then: A_1: "report Application Tag with value "*Remote Telnet*""

Table II.6.2 – Example

DPI Policy Rule ("**Detection of *Remote Telnet* based on inspection of IP application protocol itself**") R_x: "…", Id = …, precedence = …	
Condition(s)	**Action(s)**
If: C_1: "L4+ PDU contains *Remote Telnet* specific control commands …"	Then: A_1: "report Application Tag with value "*Remote Telnet*""

This example illustrates application identification with and without "application payload" inspection".

II.7 Further examples

II.7.1 Example for application detection without independent of flow descriptor usage or not

The purpose could be the detection of application "image objects encoded in *JPEG File Interchange Format* (JFIF)" (NOTE 1). This DPI application could be detected by monitoring traffic with or without a particular rule for flow identification because just an application-level condition would be sufficient..

NOTE 1 – Such a data object is typically carried within the payload of an application layer protocol. The DPI policy rule could contain the condition for searching the so-called *JFIF-Tag* equal to "`FF E0 00 10 4A 46 49 46 00 01`".

APPENDIX III

Policy Enforcement Process

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

III.1 Introduction

This appendix provides complementary information concerning the used concepts, terminology and example functional models in this Recommendation. This appendix is orthogonal to the requirements, as specified in the main body of this Recommendation.

III.2 (DPI) Policy rule

III.2.1 Concept

It may be noted that the used concept of a policy rule (see clause 3) is generic and thus applicable for DPI-specific *policy rules*, but also non-DPI related policing like for medium depth packet inspection (MPI) or shallow-depth packet inspection (SPI). Figure III.1 summarizes the used policy rule concept by this Recommendation. There are one or multiple (DPI) policy rules R_{DPI} in place, called DPI policy rule set $\underline{\underline{\boldsymbol{R}}}_{DPI}$. The rules are stored in the DPI-PIB, which determines consequently the behaviour of the DPI-FE.

Any individual rule $R_{DPI,i}$ may contain one or multiple policy *conditions* C_k and one or multiple policy *actions* A_m. It may be noted that every DPI policy requirement (as specified by this Recommendation) may be satisfied by this policy rule concept.

III.2.2 (DPI) Policy condition

The *policy condition* represents a logical function (which may lead typically to a compare operation or search operation on elements of the evaluated packet). Such a logical function may contain itself one or multiple literals (also known as *simple* versus *compound policy conditions*). Compound policy conditions are typically specified as Boolean normal form (like *disjunctive normal form* (DNF) or *conjunctive normal form* (CNF)), which may imply a correspondent conversion between the high level policy management and decision process towards the execution in the packet path by the DPI-FE (e.g., a compound policy condition in format "$(\neg C_1 \wedge C_2) \vee C_3$) " must be converted).

NOTE 1 –The applied structure (simple vs. compound) of policy conditions, and whether a policy rule provides only a single condition or multiple conditions per rule, is not relevant for this Recommendation. Because each specification structure for policy conditions (and rules) may be converted to each other. They are adequate from functional perspective.

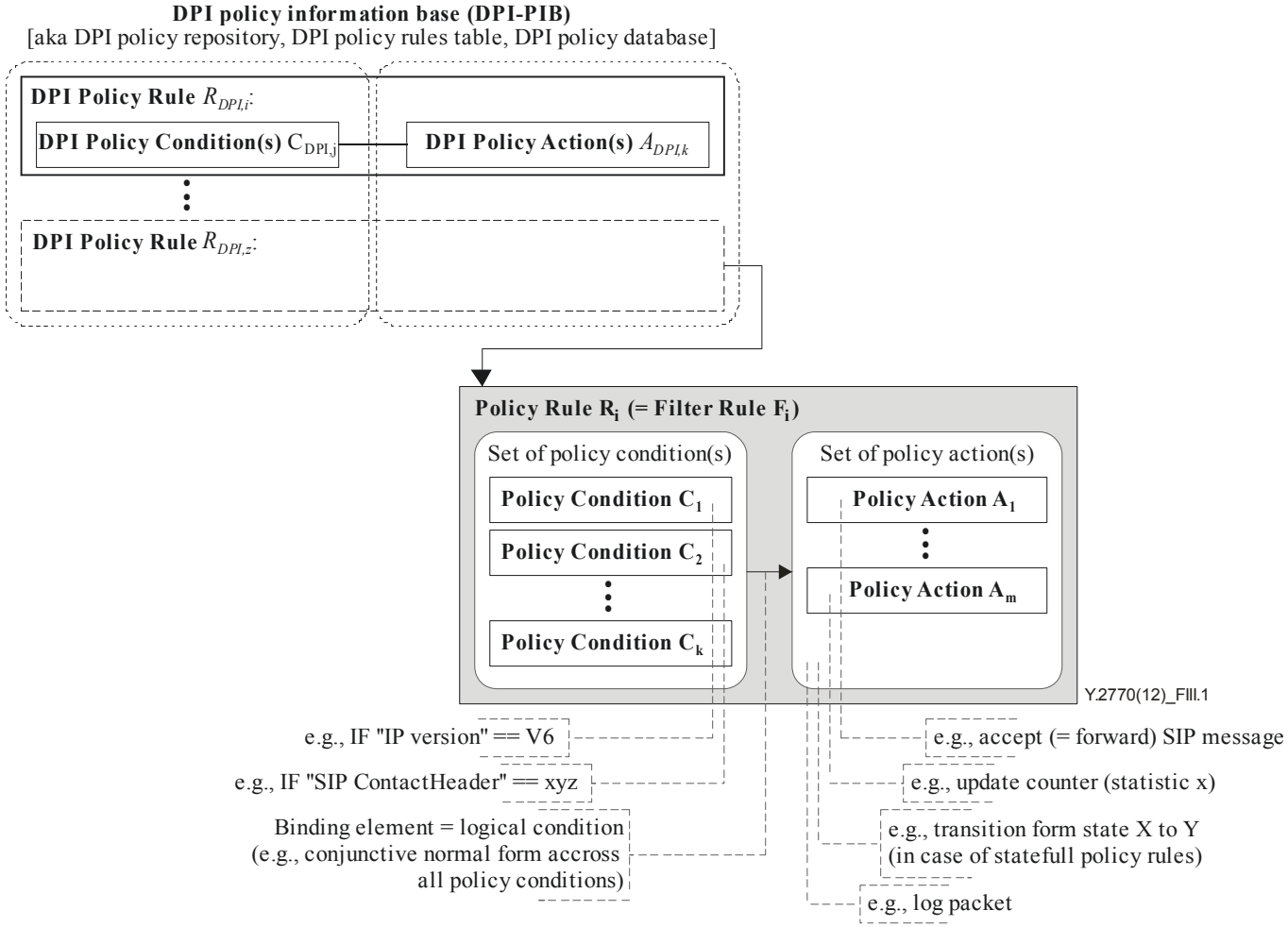


Figure III.1 – Policy Enforcement Process – Example Structure of a (DPI) Policy Rule

It may be reminded again that the specification of explicit bindings between actions and conditions, and also the specification of entire DPI policy rules as such is out of scope of this Recommendation (see clause 1.1).

III.2.3 Hierarchical (DPI) policy conditions or/and (DPI) policy rules

The packet path processing by the DPI-FE may be either just straightforward in terms of a "flat" link between action(s) and condition(s), or hierarchical by embedded, nested, recursive, etc. rule structures. For instance, a top-level policy rule may contain a pointer (in the rule action) to a subsequent set of policy rules etc. See also clause III.3.

The example model with two functional stages of "packet scanning" and subsequent "packet analysing" represent a hierarchical concept in terms of distributed processing of policy conditions.

III.3 (DPI) Policy Enforcement

III.3.1 Staged Process Model

III.3.1.1 Overview

The policy rule structure implies a *staged processing* model by checking firstly the policy *conditions* (1) and a subsequent execution of identified policy *actions* (2). Figure III.2 provides an example model of such a processing structure.

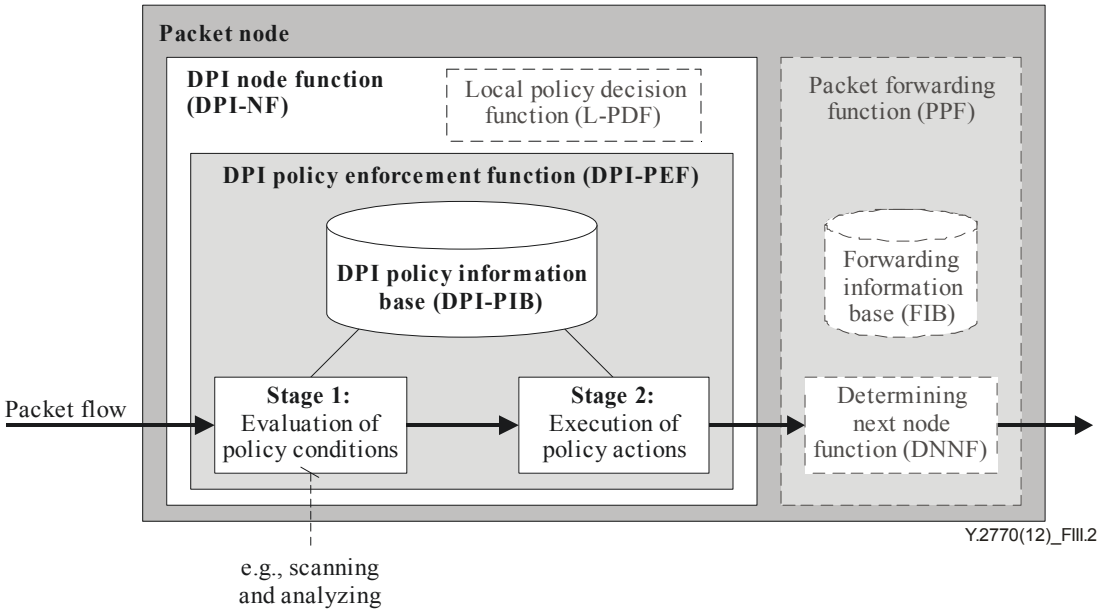


Figure III.2 – Policy Enforcement Function – Example of a 2-stage Rule Enforcement Process (Note: the PFF is only present for *In-Path DPI* mode)

The rule processing stages may be refined, e.g., by the introduction of *hierarchical stages* for the evaluation of different *set* of policy conditions, see Figure III.3. The (technical) motivation for such staged structures is related to goals, e.g., the improvement of rule processing efficiency or maximization of DPI packet processing rates.

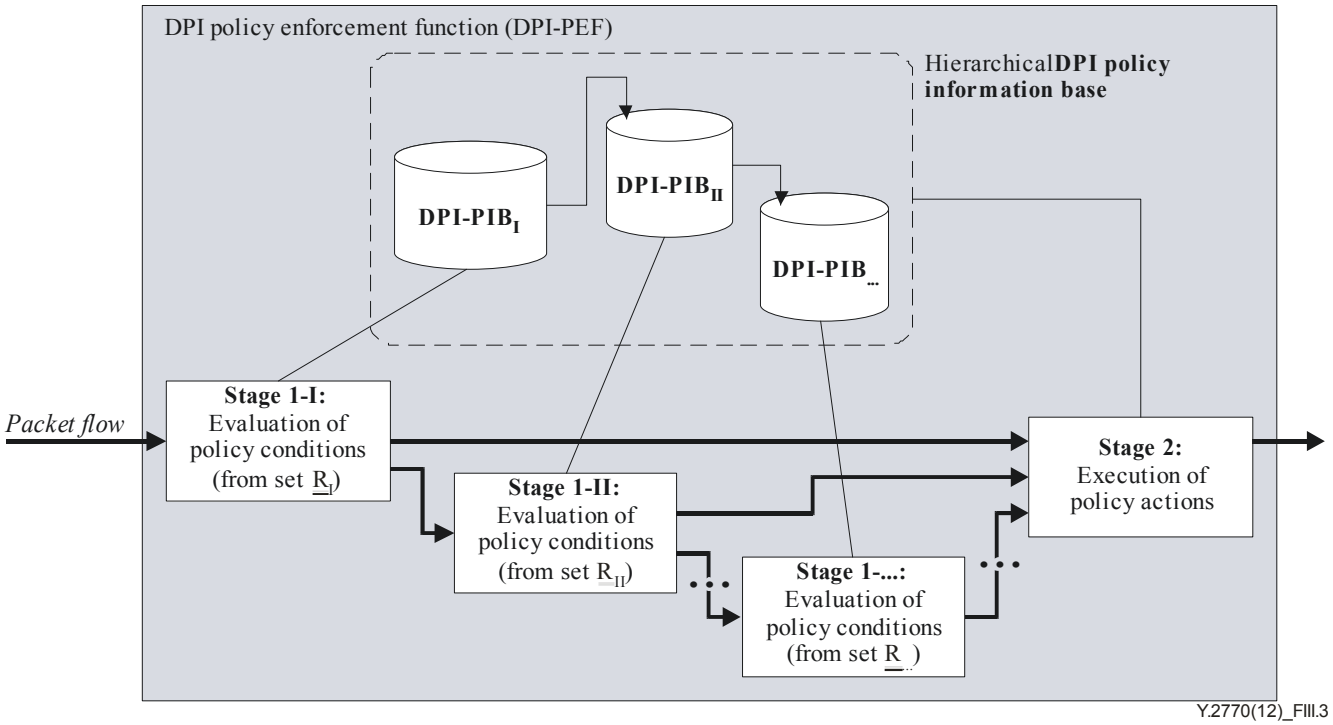


Figure III.3 – Policy Enforcement Function – Hierarchical Evaluation Stages for Policy Conditions

NOTE 3 –Generic example to Figure III.3: the first set of policy conditions (related to rule set $\underline{\underline{R}}_I$ from *DPI-PIB*$_I$) may represent just a few, $L_{3,4}HI$ only related conditions with the goal in a coarse granular classification of packet flows. The next stage (related to rule set $\underline{\underline{R}}_{II}$) is optional, dependent of further, more detailed classification objectives. And so on. The final stage may contain just the most complex DPI rules, e.g., a condition for a specific application-level security threat.
Such a hierarchical structure allows to reduce the number of policy rules (and thus also conditions) to be performed down to the absolute minimum per packet.

III.3.1.2 Motivation and some examples

Any functional-to-physical mapping scenario may be driven by performance objectives for specific implementations. For instance by a hierarchical policy processing model mapped on a multi-processor or multi-process environment by using a serial execution organization may negatively impact performance objectives like DPI-NF-internal *packet transfer delay*, even violate any real-time objectives. However, there might be even a physical multi-stage "DPI processing pipeline", driven by performance objectives like the *optimization* of the overall DPI *packet throughput* by a DPI node. Such a *maximization* approach would be dependent on concrete DPI policing scenarios, which may benefit from hierarchical DPI processing.

Some examples (by referring to the generic model by Figure III.3):

Example 1 - DPI for TCP-based IP applications:

Stage 1-I may focus on the separation in TCP and non-TCP traffic, and Stage 1-II may then just enforce the TCP-specific rule. Such an approach would address the two performance objective in a) minimizing the transfer delay for non-TCP packets (because the resources for TCP-specific rules are save) and b) the maximization of the overall packet rate for TCP and non-TCP traffic;

Example 2 - DPI for SIP traffic:

Stage 1-I may focus on SPI-dedicated policy rules like the policing for "flooding attacks", any violating SIP message would be then just forwarded to (and discarded by) Stage 2, and Stage 1-II may continue with SIP DPI rules on SIP header elements, and Stage 1-III could provide an inspection of the SIP message body. Again, it's an hierarchical approach, driven by performance optimization.

Example 3 - DPI for QoS support:

Stage 1-I may provide a packet classification on a coarse granularity, - like specific QoS support on aggregate level, and Stage 1-II may continue with the classification on fine granularity level, e.g., identifying 'flows' (within an aggregate) on a very low level. The hierarchical approach is here also reflecting packet actions on aggregate and flow level.

Example 4 - DPI for measurement support:

A DPI entity may be used in the context of the generation of local measurements (see flow metering process in clause 6). Thus, such a "DPI-correlated statistic" is tight to a *Flow Descriptor*. E.g., a correspondent DPI policy rule "measure the traffic volume for RTP packets with video codec type 'H.264' and H.264 profile 'x'" relates to policy conditions up to L7 and the action in "updating statistic *xyz*". It would be questionable in locating such a rule already in Stage 1-I due to possible inefficient processing because of other possible "flow packets" like RTP audio packets, RTCP, or non-RTP traffic etc.

Example 5 - DPI for general IP security:

Stage 1-I may focus on' legacy' policy rules, related to well-known security attacks and subsequent Stages may address more complex policy rules, just related to a specific traffic portion (which was identified by the previous Stage).

Example 6 - DPI for MPLS traffic:

A similar hierarchical rule partitioning as in previous examples, always dependent on the concrete applied DPI policy rules, may allow processing performance improvements.

Example 7 - DPI for LxVPN traffic:

A similar hierarchical rule partitioning as in previous examples, always dependent on the concrete applied DPI policy rules, may allow processing performance improvements.

Staged DPI processing models are thus justified in many scenarios.

III.3.2 Processing Stage 1: Packet Classification

The first stage relates to the evaluation of the policy conditions against an incoming packet, see Figure III.2. This operation corresponds to a DPI entity internal *classification function* because the set of policy conditions may be abstracted by a "traffic class". Every incoming packet is thus firstly assigned to a particular internal processing class. This internal *packet classification* function is also known as *packet identification*, because each classified packet may be identified by a lookup-key(see also Appendix VII), which again is given by the policy conditions. Packets with the same results with regards to the flow level conditions belong to the same traffic *flow*.

III.3.3 Processing Stage 2: Action Execution

Action(s) are executed if identified by the set of policy conditions. There are three basic behaviour cases from packet path perspective:

- packet unmodified forwarded;
- packet modified forwarded (e.g., by "QoS tagging" packet header elements, by translated network address information); or
- packet not forwarded (e.g., by discard).

The overall (DPI) PEF provides consequently a *filtering* behaviour for traffic flows (out of an overall traffic aggregate) and for individual packets (out of a specific flow). Such policy rules are thus also termed as *filter rules*. The terms policy rule and filter rule are synonym in this Recommendation.

III.4 Notes to Staged Process Models

The two processing stages for DPI policy conditions, the *scanning function* and *analysing function*, may lead to an overall 3-stage processing model. Such a staged model may be motivated by similar processing models in other computer areas, e.g., the translation of computer programming source code in executable instructions by interpreters or compilers, because there are a lot of commonalities between policy enforcement processing (of packet traffic) and the compilation processing (of programming languages), from perspective of a runtime environment.

The two stages of lexical and syntactical analysis in the parser model of a compiler are very similar to the processing model of (DPI policy) conditions by the two stages of DPI scan function and DPI analyser function.

APPENDIX IV

Policy Specification Languages

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

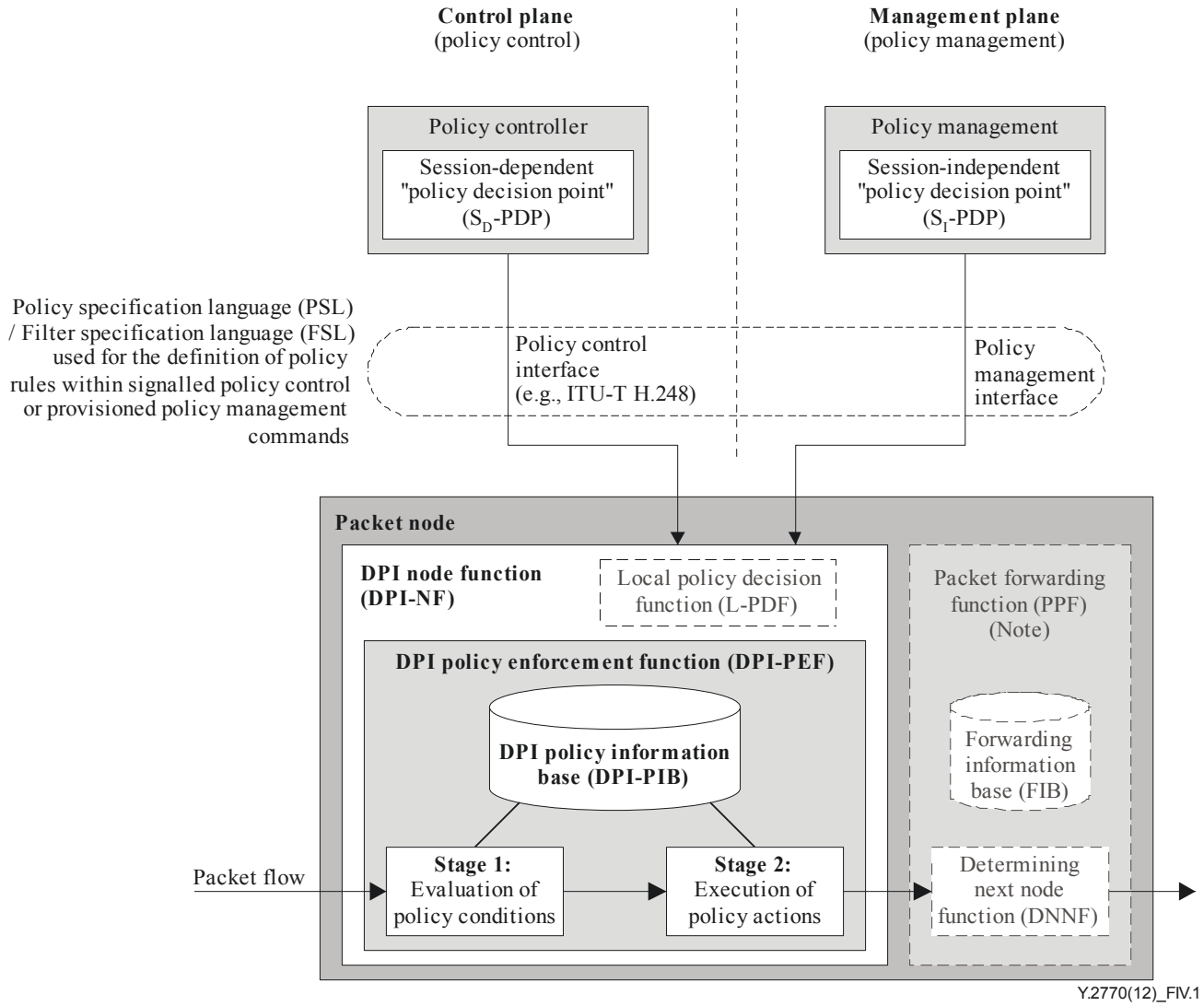
IV.1 Introduction

This appendix provides complementary information on *Policy Specification Languages* (PSL), also known as *Policy Expression Language* (PEL) of *Filter Specification Language* (FSL). The definition of a PSL is out of scope of this Recommendation. However, the question of PSLs is related here to multiple network interfaces, like a *control plane* policy control interface between a remote policy decision function and the DPI node function and a *management plane* policy management interface between network (policy) management functions and the DPI node function. There are thus inherent protocol requirements across multiple, different network interfaces concerning the "transport[9] of policy rule sets" down to the DPI-FE (see also Figure IV.1).

IV.2 PSL for Policy Control and Policy Management Interfaces

Figure IV.1 provides a summary of a typical network scenario. The policy operations by the control plane and network plane addressing both the same objects of the policy enforcement path in the user plane. Thus, an aligned PSL usage across all relevant interfaces would be crucial for efficient DPI node functions.

9 E.g., via high-level push mode or pull mode operations between policy decision entities and the policy enforcement processing path.



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

Figure IV.1 – Policy Specification Language (PSL) – PSL for Policy Control and Policy Management Interfaces

IV.3 Survey of possible PSLs (non-exhaustive list)

(DPI) policy rules are enforced on *Protocol Data Units* (PDU) in general, briefly called packets in this Recommendation. The *objects* of DPI are therefore parts of or entire PDUs. A PSL must consequently provide specification means for the definition of such objects ("*data structure*") and methods executed on these objects (i.e., "*operations*", related to policy conditions and policy actions). Table IV.1 provides a list of example standardized protocols (NOTE 1), which may be candidates for DPI-capable PSLs. The example PSLs provide initial support for the specification of such *data objects* or/and considered *operations*.

NOTE 1 – There might be also proprietary protocols around, particularly for management interfaces (like command line interfaces (CLI) or man-machine interfaces (MMI)).

Table IV.1 – Example list of *Policy Specification Languages* (PSL) (aka *Policy Expression Language* (PEL), *Filter Specification Language* (FSL))

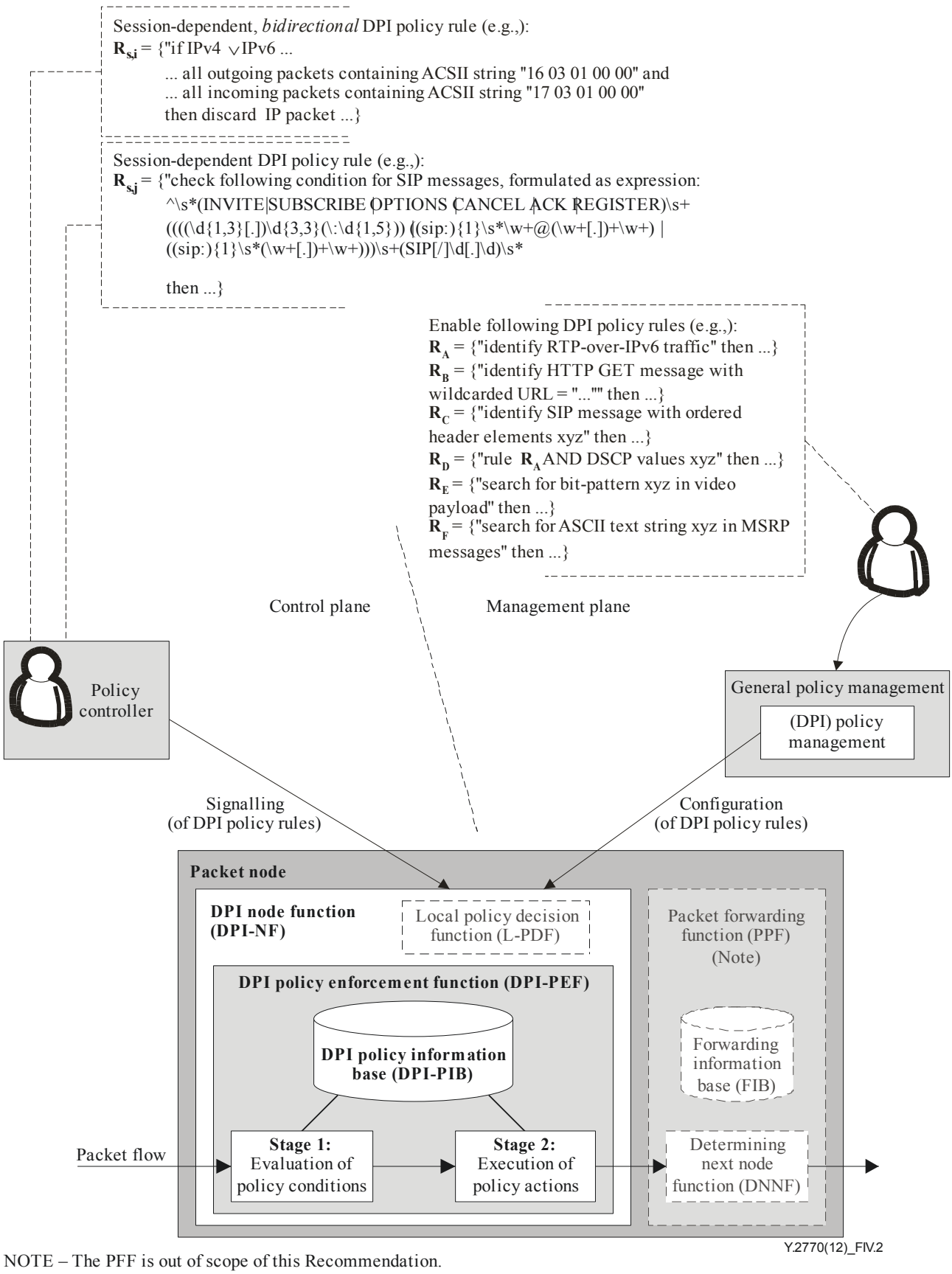
No	Policy Specification Language	PSL elements for L_2HI, $L_{3,4}$HI, L_{4+}HI & L_7PI	Reference
1	SAML – Security Assertion Markup Language (SAML 2.0)	FFS	[b-ITU-T X.1141]
2	XACML – eXtensible Access Control Markup Language (XACML 2.0)	FFS	[b-ITU-T X.1142]
3	Open Service Access (OSA) Application Programming Interface (API); Part 13: Policy management Service Capability Feature (SCF)	FFS	[b-3GPP 29.198-13]
4	SIEVE – An Email Filtering Language	FFS	[b-IETF RFC 5228]
5	BPEL – Business Process Expression Language	FFS	[b-OASIS BPEL]
6	BPML – Business Process Modelling Language	FFS	[b-OMG BPML]
7	SNMP (with Middlebox Communication (MIDCOM) Protocol Semantics)	FFS	[b-IETF RFC 5189]
8	SNMP Policy Based Management MIB (= PIB)	FFS	[b-IETF RFC 4011]
9	H.248 – Gateway Control / Policy Control Protocol	FFS	[b-ITU-T H.248.1]
10	COPS – Common Open Policy Service Protocol	FFS	[b-IETF RFC 2748]
11	DIAMETER	FFS	[b-IETF RFC 3588]
12	XCAP – XML Configuration Access Protocol	FFS	[b-IETF RFC 4825]
13	PEEM Policy Expression Language (by Open Mobile Alliance)	FFS	[b-OMA OMA-TS-PEEM_PEL-V1]
14	PACKETTYPES	FFS	[b-PacketTypes]
15	APF – A Packet Filter	FFS	[b-APF]
16	RTAG – Real-Time Asynchronous Grammars	FFS	[b-RTAG]
17	TAP/APC – Timed Abstract Protocol & Austin Protocol Compiler	FFS	[b-TAP]
18	GAPAL – Generic Application-Level Protocol Analyser and its Language	FFS	[b-GAPAL]
19	…		
NOTE – 'FFS' means for further study. The evaluation of potential PSLs against support for DPI-based policy rule specifications is out of scope of this Recommendation.			

IV.4 PSLs on different network levels

It may be worth to consider PSLs on different network levels. There might be very high-level PSLs with focus on behavioural policy definitions, using natural languages. On the other side could be low-level PSLs, close to the program code ("e.g., configurations of policy rules at API level") of packet-path processing components for policy enforcement (e.g., ASIC, FPGA, network processor,

general purpose CPU), using a formal specification approach, which is also a prerequisite for the detection of possible rule interaction problems.

Figure IV.2 illustrates just some examples for policy rule specifications.



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

Figure IV.2 – Policy Specification Languages – *Example* (DPI) Policy Rules (on different network levels)

IV.5 Recommendations for selected PSLs

Figure IV.3 outlines a possible PSL architecture concept, which would satisfy the requirements of:

(R1) *single*, aligned PSL for policy control and policy management;

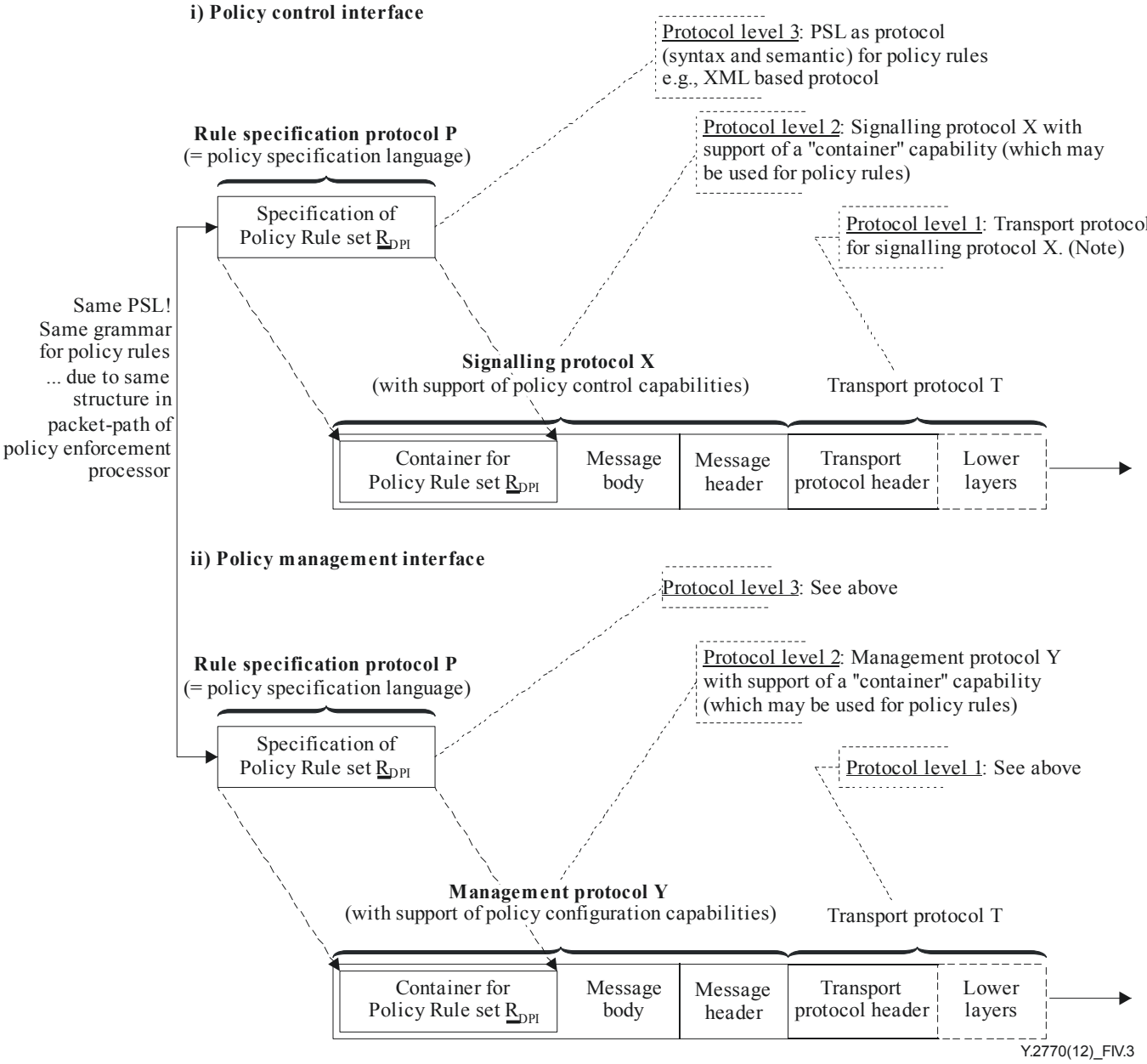
(R2) PSL *decoupled* from a control plane *signalling protocol*, thus PSL-independent of a dedicated signalling protocol;

NOTE – Concept is already well established in many protocols, principle is equal to the "MIME concept for electronic mail", i.e. a *multipurpose extension* capability by the "carrier protocol". A "multipurpose extension" mechanism would also allow different PSL types.

(R3) ditto for *management protocol*;

(R4) the specification of a Policy Rule set $\underline{\underline{R}}_{DPI}$ (but also $\underline{\underline{R}}_{non\text{-}DPI}$) would be embedded in a container of the underlying signalling or management protocol;

(R5) alignment of object models and information bases (e.g., between PIBs on PEF-level and policy decision/management entities).



NOTE – Many signalling protocols are designed "transport-independent", thus support of multiple t ransport modes.

Figure IV.3 – Policy Specification Languages – Possible PSL Architecture Concept

Figure IV.3 shows an abstract rule specification protocol *P* (as PSL), which is preferably used by network entities in the control and management plane. Any aligned PSL leads to aligned PIBs. Any Policy Rule set ***R_DPI*** is carried by signalling (*X*) or management (*Y*) protocols.

APPENDIX V

DPI in layered protocol architectures

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

V.1 DPI versus non-DPI

The elements for DPI may be subject of policy conditions, given by an overall policy rule concept for defining the functional behaviour of a DPI entity. See an example in Figure V.1. It should be noted that DPI is applicable for all kind of layered protocol architectures.

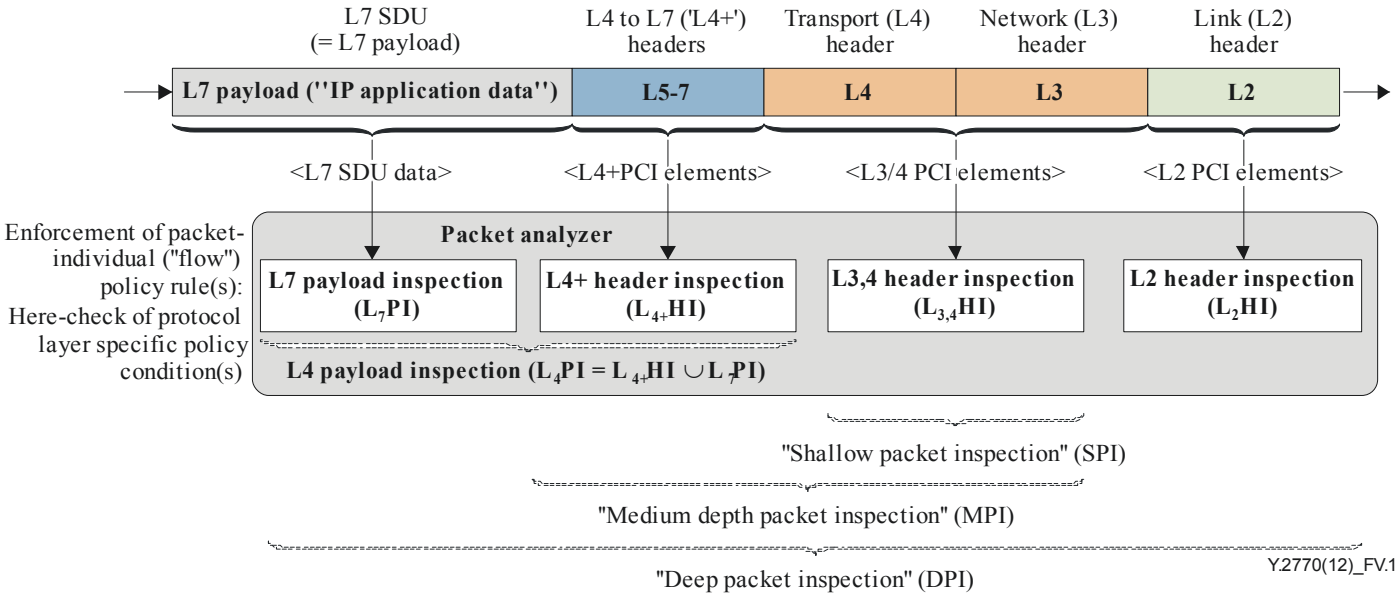
NOTE 1 – The information elements used by these identifiers are tight to a layered protocol architecture, see clause 3.2.16 for flow level conditions. This concept allows to discriminate DPI from non-DPI (see an example in NOTE 2).

NOTE 2 – Nonstandard terms, often used in literature:

"Shallow Packet Inspection" (SPI) = $L_{3,4}HI$

"Medium Depth Packet Inspection" (MPI) = $L_{3,4}HI \cup L_{4+}HI$

"Deep Packet Inspection" (DPI) = $L_2HI \cup L_{3,4}HI \cup L_{4+}HI \cup L_7PI = L_2HI \cup L_{3,4}HI \cup L_4PI$



Legend:
LX Protocol layer X
SDU Service data unit ("packet payload")
PCI Protocol control information ("packet header")
PDU Protocol data unit ("packet")

Figure V.1 – Packet Inspection – Illustration of Terminology

V.2 Example reference models for some layered protocol architectures

V.2.1 DPI for packets according IETF-BRM protocol layering

The generic DPI definition of clause 3.2.5 may be adapted for network scenarios with concrete protocol layering models. Here at the example of IETF IP network models.

Definition – DPI for packets according IETF-BRM protocol layering (abbreviated as $DPI_{IETF-BRM}$)**:** The IETF basic reference model, given by [IETF RFC 791], relates to the OSI-BRM without protocol layers L5 and L6. The $DPI_{IETF-BRM}$ is thus based on an absolute protocol layering model. There is $DPI_{IETF-BRM}$ in case of policy rules for *deep* packet inspection (NOTE 1) with policy conditions primarily based on elements related to protocol layers *above the transport layer*.

NOTE 1 – This does not exclude other indicated methods for DPI application identification in the main body of this Recommendation.

The IP protocol stacks in use may consider a refined reference model by, e.g., additional protocol layers between the application and transport layer. For instance, the application level framing protocol RTP in case of RTP traffic (see Figure V.2).

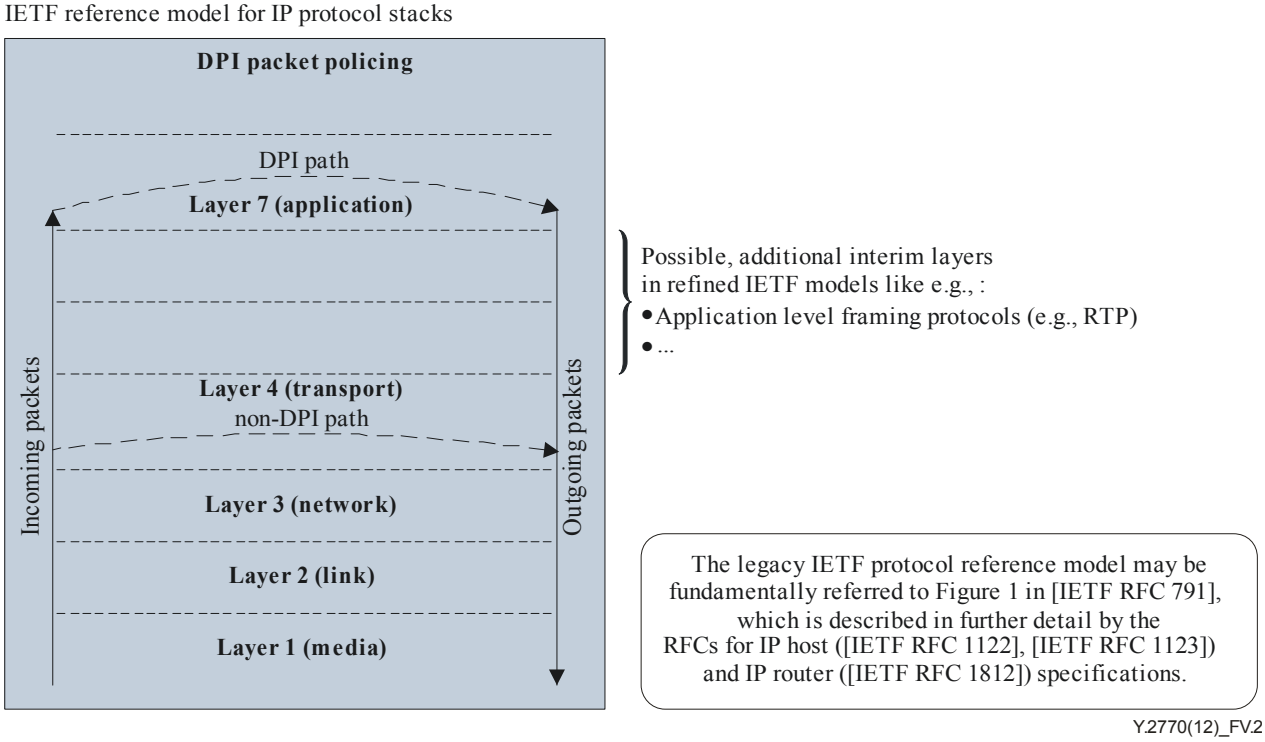


Figure V.2 – IETF (Basic and extended) Reference Model for IP

V.2.2 DPI for packets according other IETF reference models

The reference model for IETF IP networks of Figure V.2 may be further refined. Considering present IP networks in pre-NGN and NGN solutions, then there are often more detailed protocol stack architectures due to, e.g., access network type specific stacks, tunnelling methods, IPv4-to-IPv6 network transitioning scenarios or application specific framing and sub-layering. However, the very majority of such heterogeneous IP stacks could be mapped on three example reference models, called here *basic*, *extended* and *tunnelled* reference model (BRM, ERM, TRM), see Figure V.3.

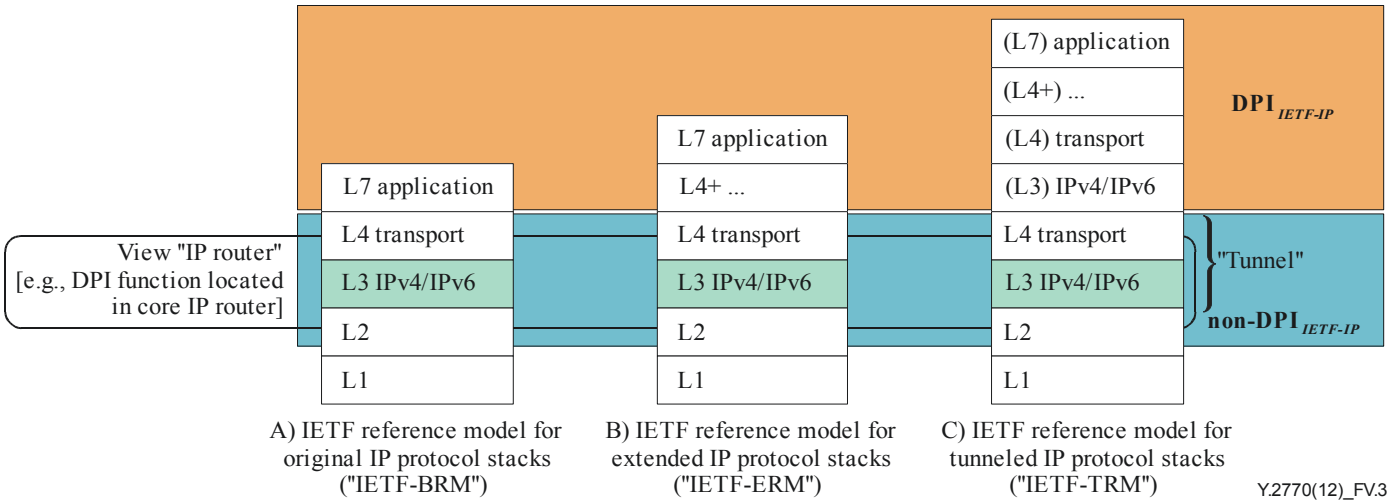


Figure V.3 – DPI examples for packets according other IETF reference models

It may be concluded that above definition for $DPI_{IETF\text{-}BRM}$ would be still valid for the other reference models, from perspective of an IP hop entity (NOTE 2).

NOTE 2 – The aspect of "deep" is thus dependent on the *location* of the DPI-FE. E.g., above discrimination between DPI and non-DPI could be different for a DPI-FE integrated in an IP node which terminates a tunnelling protocol.

APPENDIX VI

Formal specification of major terminology

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

VI.1 Introduction

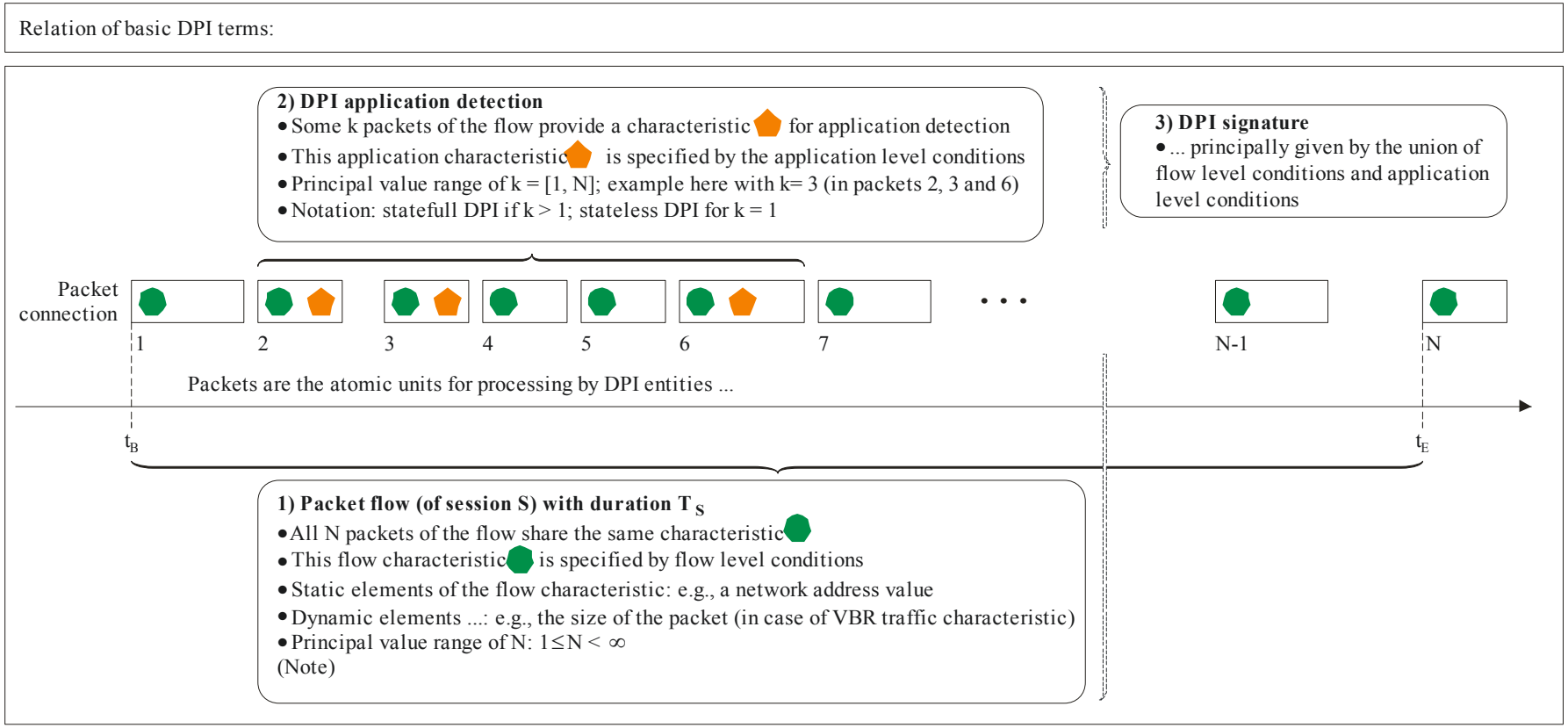
Terminology is defined in clause 3. There are some crucial terms, which are related to each other in scope of DPI. The purpose of this appendix is to highlight the principal relations. The scope of this Appendix is on the terms *flow descriptor (flow level conditions)*, *application descriptor (application level conditions)* and *DPI Signature*.

Using a formal description for the term definitions allows a more precise elaboration and indication of the differences.

Where there are discrepancies between this Appendix and clause 3, the main body of this Recommendation take precedence over this Appendix.

VI.2 Summary and illustration of terms

Figure VI.1 provides a high-level summary of the underlying concepts and relations of these terms.



t_B Arrival time of 1st packet (of a flow) at DPI entity
t_E Arrival time of last packet (of a flow) at DPI entity
t_S Session duration ($= t_E - t_B$)

NOTE – There'll be a maximum value in practice due to the finite length of communication associations in reality.

Figure VI.1 – Illustration of the three major terms *flow descriptor*, *application descriptor* and *DPI signature*

VI.3 Using a formal description technique for the terms

ABNF is used as an example formal language in this Appendix.

VI.3.1 Formal specification of flow descriptor (flow level conditions)

Table VI.3.1 provides a formal description of the flow level conditions, which is in line with the prose specification for *flow* (clause 3.1.3) and flow descriptor/flow level conditions (clause 3.2.16).

Table VI.3.1 – Formal specification of flow descriptor (flow level conditions)

ABNF (shortened)	Comments
`Flow Descriptor = CompoundCondition`	The flow descriptor relates to a logical function, which is effectively a set of rule conditions enforced for packet (flow) policing.
`CompoundCondition = DNF (*SimpleCondition) / CNF (*SimpleCondition)`	DNF Disjunctive Normal Form CNF Conjunctive Normal Form
`SimpleCondition = "(<Variable> MATCH <Value>)"`	Elementary condition structure.
`Variable = ...`	Flow-specific *Information Element* (e.g., according IETF IPFIX registry)
`MATCH = "=" / "<" / ...`	Match operator (the relation between variable and value; NOTE 1).
`Value = ...`	given by IE-specific defined value range
NOTE 1 – Inclusive other operators, e.g., for heuristics, behavioural, or statistical relations e.g., "nearly match", "approximative match", etc.	

VI.3.2 Formal specification of application descriptor (application level conditions)

Table VI.3.2 provides a formal description of the application level conditions, which is in line with the prose specification of clause 3.2.2.

Table VI.3.2 – Formal specification of application descriptor (application level conditions)

ABNF (shortened)	Comments
`Application Descriptor = CompoundCondition`	Conceptually the same as the Flow Descriptor.
`CompoundCondition = DNF (*SimpleCondition) / CNF (*SimpleCondition)`	See Table VI.3.1.
`SimpleCondition = "(<Variable> MATCH <Value>)"`	See Table VI.3.1.
`Variable = ...`	Generic Information Element (NOTE 1,2)
`MATCH = "=" / "<" / ...`	See Table VI.3.1.
`Value = ...`	See Table VI.3.1.
NOTE 1 – The location (within the protocol data unit) of this IE may be additionally limited on a particular range of the packet (e.g., via a bit-offset). NOTE 2 – There could be also internal (state) variables in case of stateful DPI.	

VI.3.3 Formal specification of *DPI Signature*

Table VI.3.3 provides a formal description for DPI signature, which is in line with the prose specification of clause 3.2.14.

Table VI.3.3 – Formal specification of *DPI Signature*

ABNF (shortened)	Comments
`Signature = Application Descriptor AND 0*1Flow Descriptor`	Any signature comprises at least an *Application Descriptor* plus an optional *Flow Descriptor* (NOTE 1).
NOTE 1 – Leading to the two principal scenarios of Flow-dependent and Flow-independent DPI.	

APPENDIX VII

Illustration of terminology

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

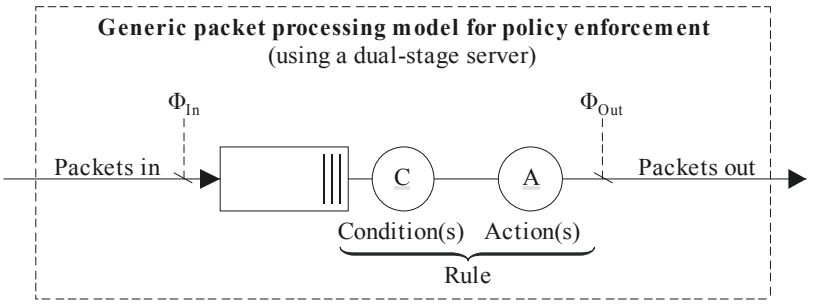
VII.1 Introduction

The main body of the Recommendation defines requirements for DPI. Some of these requirements refer to *identification* of packets (and aggregates like flow) and indicate possible *actions* after identification. The terms of *filtering*, *classification*, *modification*, etc. of packets are used in this context. There are commonalities and differences between these terms as used in the scope of this Recommendation.

The purpose of this appendix is to illustrate the underlying concept.

VII.2 Rule-oriented Packet Processing

The consideration of *packet inspection* as *rule-based packet processing* allows to depicting the key differences between the used terms. Figure VII.1 shows the generic model, based on the rule definition according clause 3.1.2.



Legend:

Φ_{In}	Ingress packet rate
Φ_{Out}	egress packet rate
A	Action(s) of a (policy) rule
C	Condition(s) of a (policy) rule (determine the packet identifier)
PDU	Protocol data unit ("packet")

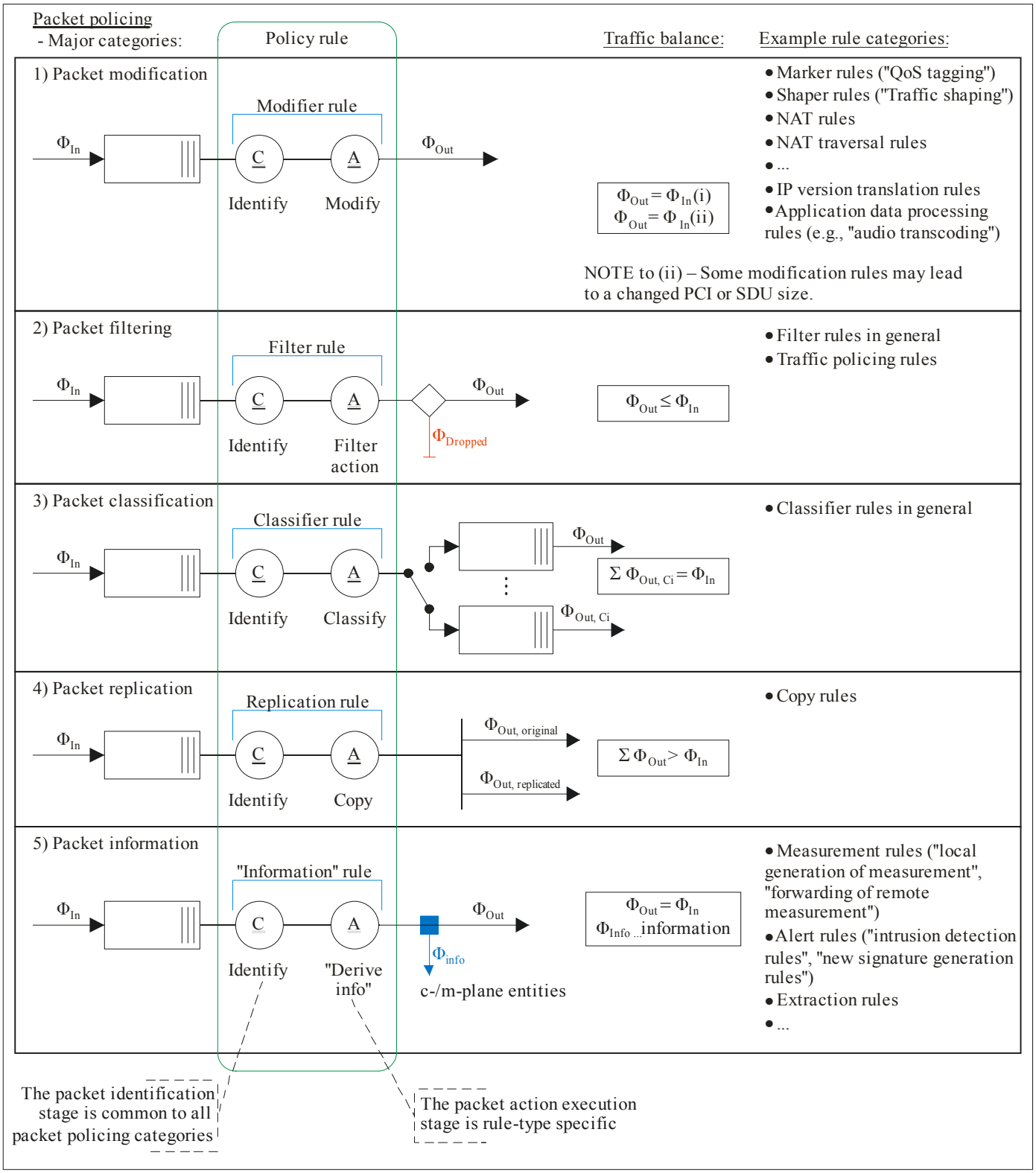
Figure VII.1 – Generic Model for Rule-oriented Packet Processing

The generic model uses a dual-stage server, given by the processing of *conditions* and *actions* as principle parts of a rule. Such a separation allows to indicate, e.g., that all basic requirements of DPI share the common characteristic of *packet identification*.

The generic model may be applied on the DPI requirements (as specified in the main body of this Recommendation), see next sub-clause.

VII.3 Major Categories of Packet Policing

The very majority of DPI requirements may be mapped on five high-level categories of packet policing (see Figure VII.2).



NOTE – All categories are relevant for DPI, however, not necessarily all indicated example rules are subject of DPI. The main body of this Recommendation takes precedence in case of discrepancies

Figure VII.2 – Specific Models for the major categories of Packet Policing

Similarities:

- each particular rule (and category) may be associated with the generic *policy rule* concept (clause 3.1.2);

- the *packet identification* stage is common to all rule categories.

Differences:

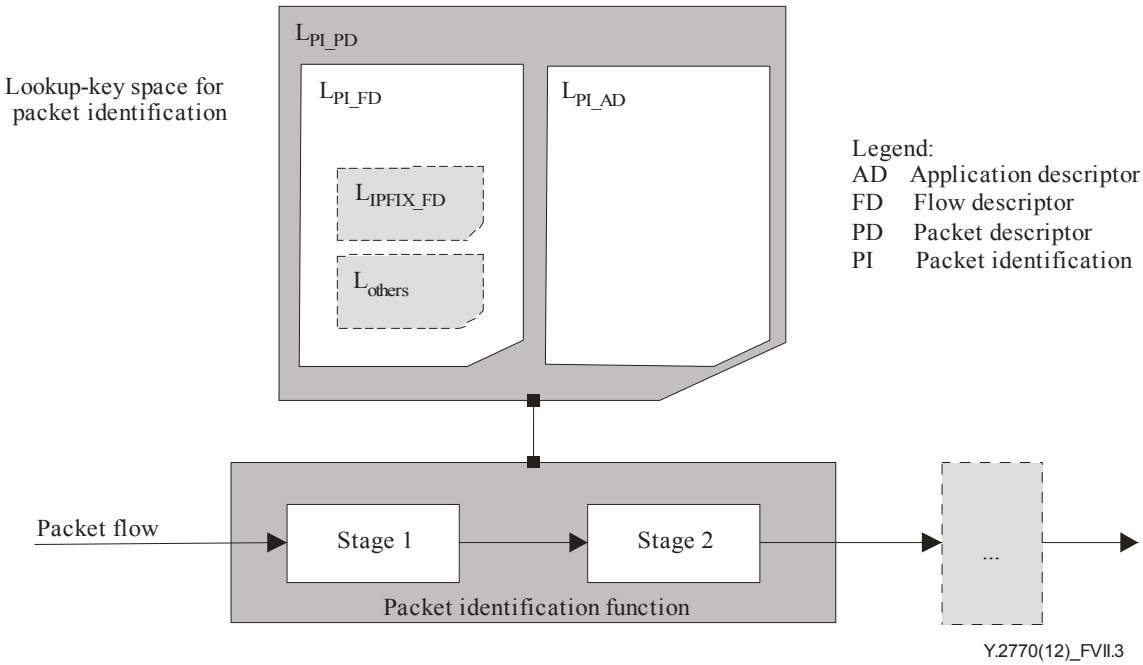
- the particular *action(s)* executed subsequently allow to distinguish different categories (e.g., packet *filtering* vs. packet *classification*).

NOTE – It may be reminded again that some DPI requirements from the main body of this Recommendation indicate just principle actions, but without any detailed specification (because this is beyond the scope of this Recommendation). E.g., the generic "filter action" in above model provides in reality a plethora of possible detailed actions (like for instance "silently discard", "discard plus alert", etc.).

It may be concluded that any (deep) *packet inspection* function may be considered as a packet policing function.

VII.4 Packet descriptor

Packet identification (PI) is the first function executed (by the DPI-FE) on an incoming packet (see clause VII.3). The identification is based on a lookup-key L_{PI_PD}, see Figure VII.3, which itself contains elements (conditions) for *application identification* (abbreviated as L_{PI_AD}) and optional *flow identification* (abbreviated as L_{PI_FD}).



NOTE 1 – Packet identification may be basically stateful or stateless. The identifiers may consequently contain elements for state information.

NOTE 2 – The identifier spaces of Flow Descriptor and Application Descriptor are typically disjoint, but may also overlap in some DPI scenarios.

NOTE 3 – Identification stages 1 & 2 represent the "Application Identification" and "Flow Identification" functions. The "Flow Identification" function is optional. If both identification functions, then any orders ...

Figure VII.3 – Packet identification (as part of packet inspection) process, Lookup-Key for Packet Inspection (L_{PI})

The packet descriptor (PD) relates to the lookup-key used by the DPI-FE for identifying an incoming packet. Thus, the PD reflects the view of the DPI-FE as network element. From network perspective ("end-to-end") will be application level conditions and (optionally) also the flow level conditions provided to the DPI-FE. Both application and flow descriptor spaces are typically disjoint because there should normally not any overlapping between application level conditions and flow level conditions (NOTE 1). Thus, the concept of a packet descriptor may be used in order to structure the descriptor used for DPI:

PD = ***FD*** + ***AD***

FD = ***IPFIX-FD*** + ***Others-FD***

Others-FD = L2, L3 or/and L4 related information elements which are not (yet) in IPFIX registry (e.g., related to SCTP, DCCP)

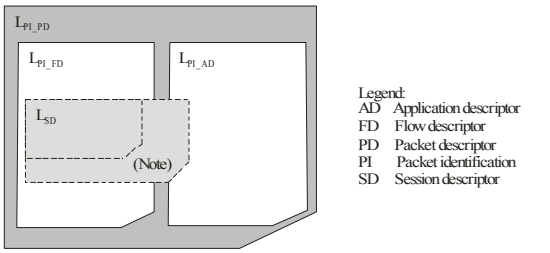
AD = NOT (FD)

It may be noted that the ***Others-FD*** may disappear in future for the case that the extended IETF IPFIX registry would cover all elements required for DPI-based flow identification.

NOTE 1 – Overlapping descriptor spaces are not excluded and actually are not any issue in practice. Effectively it could mean, e.g., the usage of the same *Variable* or even the same *SimpleCondition* in flow descriptor and application descriptor (see also clause VII.3)

VII.5 Session descriptor

There are DPI requirements on *session identification* in clause 6. This Recommendation does not support a single session concept only, rather a generic view. The correspondent session descriptor may be also not always equated with a particular *FD* or *AD*, because the *SD* space may overlap both, see Figure VII.4.



NOTE – The lookup-key for session identification is typically based upon at least a flow identifier.

Y.2770(12)_FVII.4

Figure VII.4 – Session descriptor

The SD may be a subset of the applied PD for a particular DPI service:

$$\boldsymbol{SD \subset PD}$$

Example:

There might be an *audio session* within a multimedia IP call (as, e.g., peer-to-peer service). The audio session is allowed to use multiple, specific media formats (i.e., audio encodings). There might be DPI policy rule for checking specific media encodings.

There may be following conditions for the correspondent descriptors:

– *FD* = elements for identifying the end-to-end UDP transport connection;

– *AD* = elements for RTP source identification (RTP SSRC) and a black or white list for media formats (e.g., RTP payload type identifiers);

– *SD* = FD plus RTP SSRC plus identifiers for allowed audio formats.

VII.6 Terminology on identification, classification and filtering of packets, flows and traffic

This Recommendation uses some terms related to operations executed on packets (but also higher-level traffic aggregates like flows, etc.) in scope of DPI functions. Such functions may be categorized as, e.g., illustrated in this Appendix. Figure VII-5 provides a summary and the relation between these terms. The terms identification, classification, filtering and others are sometimes used in a synonymous manner in this Recommendation, because e.g. of a more high-level consideration of a requirement etc.

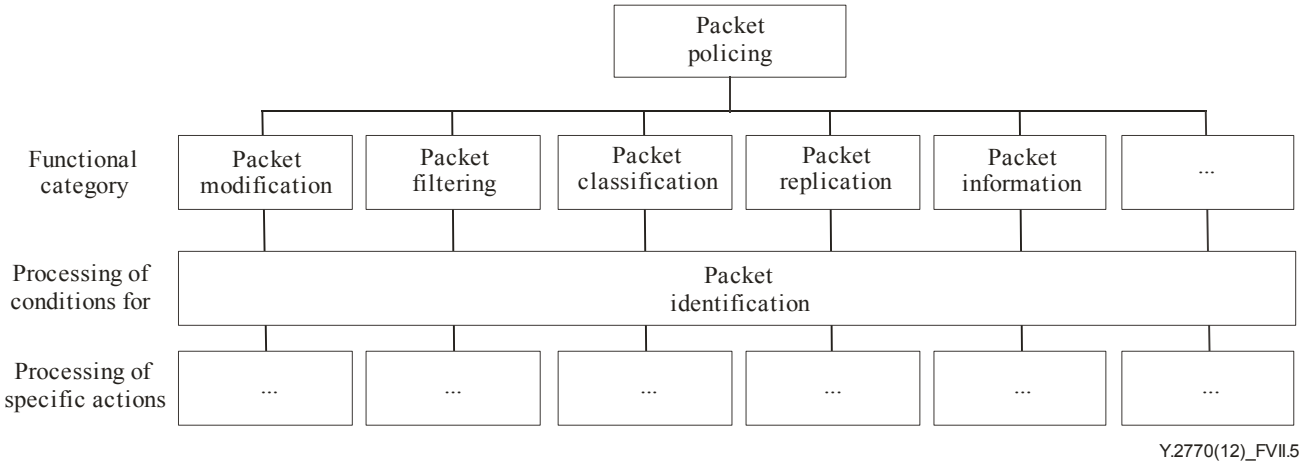
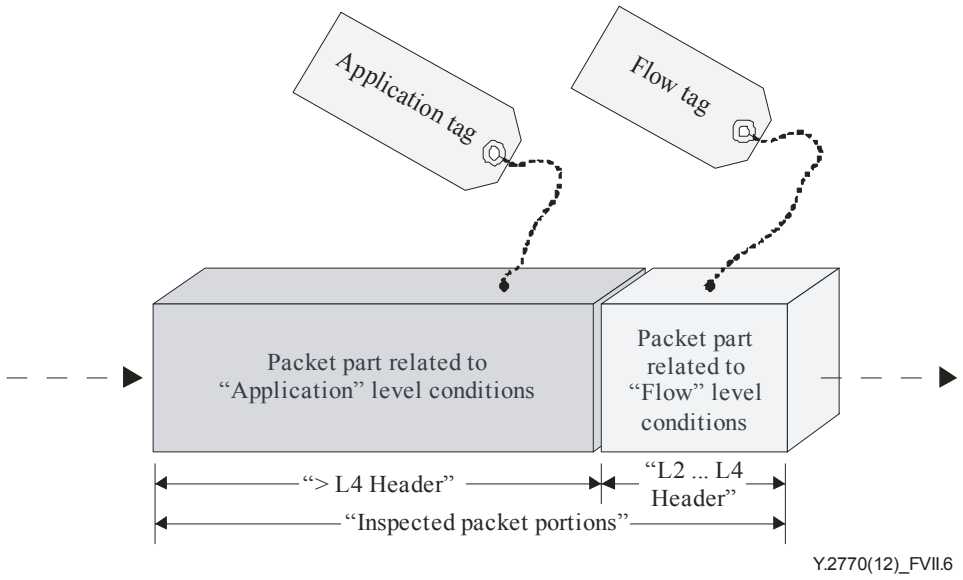


Figure VII-5 – Terminology overview related to packet policing

VII.7 Application and flow tag

The DPI requirements sections of this Recommendation (e.g., clause 6) refer to the principles of *application identification* and *flow identification*. There are correlated naming, identifier and description principles, which may be abstracted by correspondent *tags*, see Figure VII-6.



NOTE: This is an example only. the example here uses an absolute protocol layering modEL, LIKE osi-BSM. The boundary between application and flow information may be also varying in other examples.

Figure VII-6 – Principle terms related to application identification and flow identification

Any successfully inspected packet could be "identified", at least by the mandatory "application tag". The optional "flow tag" leads to the discrimination of the two DPI modes of flow-dependent and flow-independent DPI (see Figure VII-7).

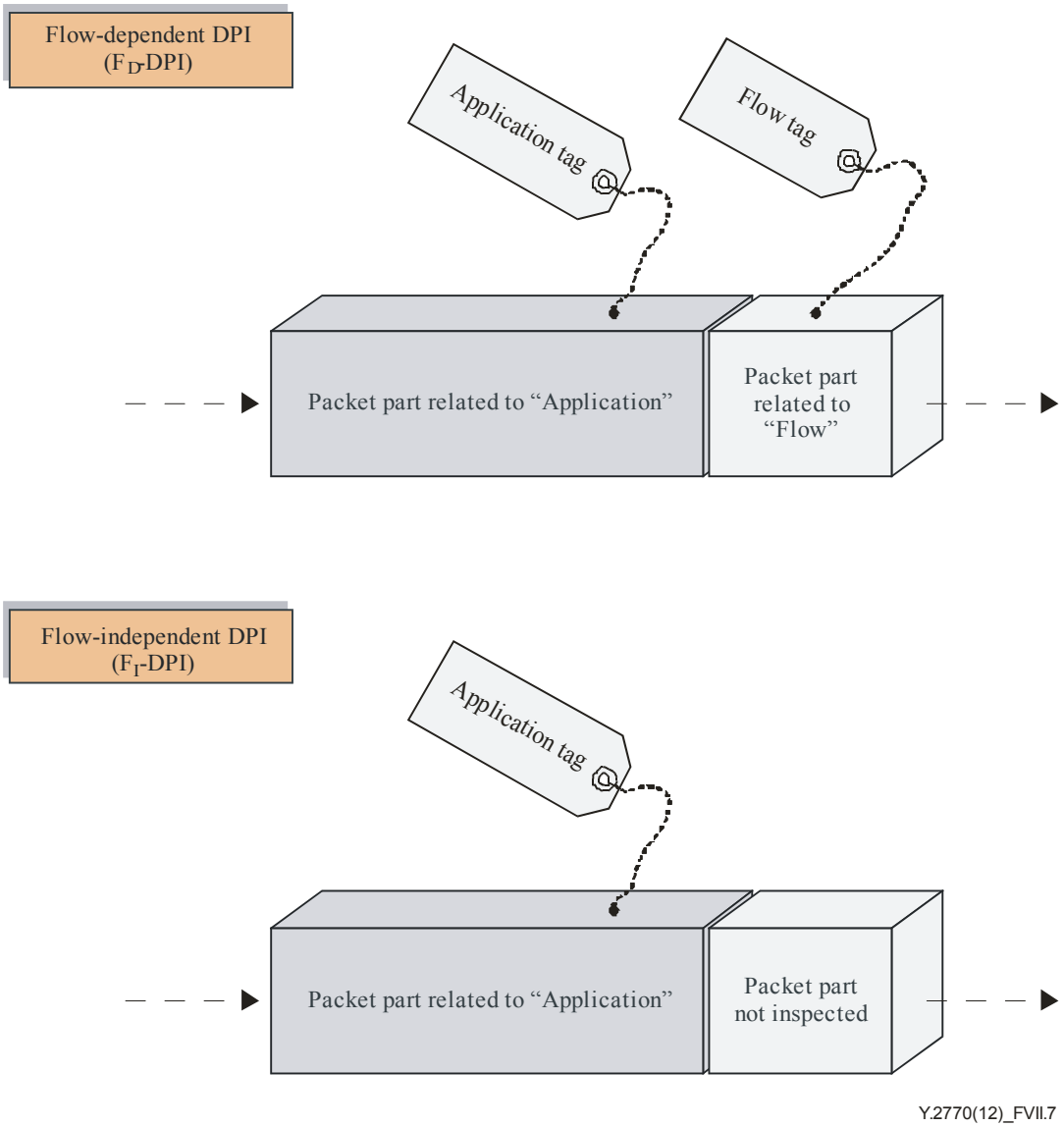


Figure VII-7 – Principle terms related to application identification and flow identification

This Recommendation provides information about the application tag, but any further detailed concept about flow tags is out of scope.

Bibliography

[b-3GPP 29.198-13] 3GPP Open Service Access (OSA) Application Programming Interface (API), Part 13: Policy management Service Capability Feature (SCF).

[b-APF] H. D. Lambright & S. K. Debray. APF: A modular language for fast packet classification. Dept of Computer Science, University of Arizona, Tucson, August 30, 1996. http://www.cs.arizona.edu/~debray/Publications/filter.html .

[b-ETSI TS 123 203] ETSI TS 123 203 (2011), Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Policy and charging control architecture (3GPP TS 23.203 version 10.4.0 Release 10).

[b-ETSI TS 124 229] ETSI TS 124 229 (2009), Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE;IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3 (3GPP TS 24.229 version 9.4.0 Release 9).

[b-ETSI ES 282 003] ETSI ES 282 003 (2011), Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Resource and Admission Control Sub-System (RACS): Functional Architecture.

[b-GAPAL] N. Borisov, D. J. Brumley, and H. J. Wang, Generic Application-Level Protocol Analyzer and its Language. Proceedings of the Network and Distributed System Security Symposium, NDSS 2007, San Diego, USA, 2007.

[b-IETF opsec] IETF draft-ietf-opsec-filter-caps (expired, 2008), Filtering and Rate Limiting Capabilities for IP Network Infrastructure. http://tools.ietf.org/html/draft-ietf-opsec-filter-caps-09.

[b-IETF IANA IPFIX] IP Flow Information Export (IPFIX) Entities. http://www.iana.org/assignments/ipfix/ipfix.xhtml.

[b-IETF IANA Port Number Registry] Service Name and Transport Protocol Port Number Registry. http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml.

[b-IETF RFC 1950] IETF RFC 1950 (1996), ZLIB Compressed Data Format Specification version 3.3.

[b-IETF RFC 2474] IETF RFC 2474 (1998), Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Header.

[b-IETF RFC 2748] IETF RFC 2748 (2000), The COPS (Common Open Policy Service) Protocol.

[b-IETF RFC 2784] IETF RFC 2784 (2000), Generic Routing Encapsulation (GRE).

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

[b-IETF RFC 3320] IETF RFC 3320 (2003), Signaling Compression (SigComp).

[b-IETF RFC 3550] IETF RFC 3550 (2003), RTP: A Transport Protocol for Real-Time Applications.

[b-IETF RFC 3588] IETF RFC 3588 (2003), Diameter Base Protocol.

[b-IETF RFC 3871] IETF RFC 3871 (2004), Operational Security Requirements for Large Internet Service Provider (ISP) IP Network Infrastructure.

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

[b-IETF RFC 4268] IETF RFC 4268 (2005), Entity State MIB.

[b-IETF RFC 4778] IETF RFC 4778 (2007), Current Operational Security Practices in Internet Service Provider Environments.

[b-IETF RFC 4825] IETF RFC 4825 (2007), The Extensible Markup Language (XML) Configuration Access Protocol (XCAP).

[b-IETF RFC 4867] IETF RFC 4867 (2007), RTP Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs.

[b-IETF RFC 5102] IETF RFC 5102 (2008), Information Model for IP Flow Information Export.

[b-IETF RFC 5103] IETF RFC 5103 (2008), Bidirectional Flow Export Using IP Flow Information Export (IPFIX).

[b-IETF RFC 5189] IETF RFC 5189 (2008), Middlebox Communication (MIDCOM) Protocol Semantics.

[b-IETF RFC 5228] IETF RFC 5228 (2008), Sieve: An Email Filtering Language.

[b-IETF RFC 5424] IETF RFC 5424 (2009), The Syslog Protocol.

[b-IETF RFC 5426] IETF RFC 5426 (2009), Transmission of Syslog Messages over UDP.

[b-IETF RFC 5476] IETF RFC 5476 (2009), Packet Sampling (PSAMP) Protocol Specifications.

[b-IETF RFC 5840] IETF RFC 5840, Wrapped Encapsulating Security Payload (ESP) for Traffic Visibility.

[b-IETF RFC 5879] IETF RFC 5879, Heuristics for Detecting ESP-NULL Packets.

[b-IEEE GLOBECOM] IEEE GLOBECOM 2008, “Online Identification of Applications Using Statistical Behavior Analysis”.

[b-ITU-T H.248.1] Recommendation ITU-T H.248.1 (11/2009), Gateway Control Protocol: Version 3.

[b-ITU-T X.734] Recommendation ITU-T X.734 (1992), Information technology - Open Systems Interconnection - Systems Management: Event report management function.

[b-ITU-T X.1141] Recommendation ITU-T X.1141 (2008), Security Assertion Markup Language (SAML).

[b-ITU-T X.1142] Recommendation ITU-T X.1142 (2008), Extensible Access Control Markup Language (XACML).

[b-ITU-T X.1243] Recommendation ITU-T X.1243 (2010), Interactive gateway system for countering spam.

[b-ITU-T X.abnot] Recommendation Draft ITU-T X.abnot (2011), Abnormal traffic detection and control guideline for telecommunication network.

[b-ITU-T X.sips] Recommendation ITU-T X.sips (2011), A framework for countering cyber attacks in SIP-based services.

[b-ITU-T Y.2011] Recommendation ITU-T Y.2011 (2004), General principles and general reference model for Next Generation Networks.

[b-ITU-T Y.2012] Recommendation ITU-T Y.2012 (2004), Functional requirements and architecture of the NGN.

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

[b-OASIS BPEL] OASIS Standard BPEL (2007), Web Services Business Process Execution Language Version 2.0. http://www.oasis-open.org/specs/index.php #wsbpelv2.0.

[b-OMG BPML] Object Management Group (OMG) BPML (2002), - Business Process Modeling Language Version 1.0. http://www.bpmi.org/bpml-spec.htm.

[b-OMA OMA-TS-PEEM_PEL-V1] Open Mobile Alliance OMA-TS-PEEM_PEL-V1 (2007), PEEM Policy Expression Language Technical Specification", Draft Version 1.0.

[b-PacketTypes] P.J. McCann, S. Chandra, PacketTypes: Abstract Specification of Network Protocol Messages. In SIGCOMM '00: Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pages 321–333, New York, NY, USA, 2000. ACM Press.

[b-RTAG] D.P. Anderson & L.H. Landweber, A grammar-based methodology for protocol specification and implementation. In SIGCOMM '85: Proceedings of the ninth symposium on Data communications, pages 63–70, New York, NY, USA, 1985. ACM Press.

[b-Subhabrata] Subhabrata S., Spatscheck O. and Wang D. (2004), Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures, The 13th International World Wide Web Conference (WWW2004, http://www.www2004.org/), May 17 - 22, 2004.

[b-TAP] T. McGuire, The Austin Protocol Compiler. http://apcompiler.sourceforge.net/.