(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 October 2018 (18.10.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/188738 A1

(51) International Patent Classification:
*H04L 12/725* (2013.01) *H04L 12/841* (2013.01)

(21) International Application Number:
PCT/EP2017/058703

(22) International Filing Date:
11 April 2017 (11.04.2017)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **NEC LABORATORIES EUROPE GMBH** [DE/DE]; Kurfürsten-Anlage 36, 69115 Heidelberg (DE).

(72) Inventors: **RIPKE, Andreas**; Kurpfalzstraße 21, 69151 Neckargemünd (DE). **HASSELMEYER, Peer**; Katharina-Staritz-Str. 17, 60438 Frankfurt (DE).

(74) Agent: **ULLRICH & NAUMANN**; Schneidmühlstraße 21, 69115 Heidelberg (DE).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

**Published:**
— *with international search report (Art. 21(3))*

WO 2018/188738 A1

(54) Title: PACKET HANDLING METHOD AND APPARATUS FOR NETWORK SERVICE FUNCTIONS

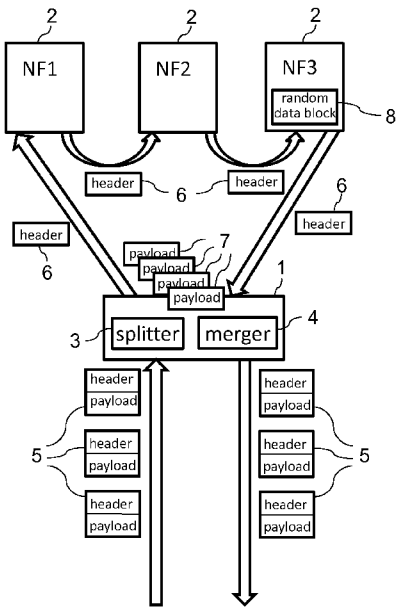



Fig. 1

(57) Abstract: In one aspect the present invention describes an efficient packet handling system for network functions (2) which involves a packet header/data split. This is a use case for dynamic virtual function acceleration (DVFA). Specifically, the invention discloses a packet handling method and apparatus for network service functions (2) includes splitting network packets (5) received at an ingress node (1) for a number of network service functions (2) into two packet parts, including a header part (6) that comprises the header or a portion of the header of the respective network packet (5), and a payload part (7) that comprises the remaining rest of the respective network packet (5) including the network packet's (5) payload, and passing on to the network service functions (2) only the header parts (6) of the network packets (5).

# PACKET HANDLING METHOD AND APPARATUS FOR NETWORK SERVICE FUNCTIONS

Generally, the present invention relates to a packet handling method used in an ingress node for a number of network service functions, and to a packet handling apparatus.

In today's network operation, service function chains, SFCs, are experiencing a widespread use. Typically, a service function chain contains a sequence of various network service function instances (hereinafter, sometimes also denoted network function, network service function or briefly NF) such that network packets of a traffic flow, which is forwarded over an SFC, are processed by the various network service function instances, wherein each service function instance applies a particular service function. Upon completion of a network packet's processing by a network service function instance, the network packet is forwarded to the next network service function instance of the SFC. Typical network functions include services like content caching, NAT (Network Address Translation), TCP optimization, video transcoding, HTTP header enrichment, IDS (Intrusion Detection System), etc.

For instance, network functions are deployed by network operators to enforce their policies and to provide additional services on top of plain connectivity (for reference, see M. Honda et al.: "Is it still possible to extend TCP?", in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference* (IMC '11), ACM, New York, NY, USA, 181-194).

Currently, there is a growing trend of implementing network service functions by means of software running in virtual machines, as envisioned in the case of Network Function Virtualization (NFV). NFV implements multiple network functions (NFs) comprising multiple network elements including functions like firewalling, header mangling, intrusion detection, load balancers, etc. In most application scenarios, most prominently service function chaining (SFC) as described above, multiple such functions are used in sequence.

If multiple NFs reside on the same physical host zero copy schemes exist to store incoming packets in a shared memory segment and to allow the respective NFs to access (read or write) the packet in this shared memory segment without any copy operations to transfer the packet from one NF to the next one.

However, in the case of network functions distributed across multiple physical hosts, which is one of the basic assumptions of NFV, NFs are not located on a single physical node and packets therefore need to be transferred between the different physical nodes. Disadvantageously, the packet transfer incurs overhead in terms of transmission time, latency, and processing.

In view of the above it is an objective of the present invention to improve and further develop a packet handling method and apparatus for network service functions in such a way that the above issues are overcome or at least partially alleviated.

In accordance with the invention, the aforementioned objective is accomplished by a method, comprising splitting network packets received at an ingress node for a number of network service functions into two packet parts, including a header part that comprises the header or a portion of the header of the respective network packet, and a payload part that comprises the remaining rest of the respective network packet including the network packet's payload, and passing on to the network service functions only the header parts of the network packets.

Furthermore, the above objective is accomplished by an apparatus comprising a splitter module that is configured to split network packets received at an ingress node for a number of network service functions into two packet parts, including a header part that comprises the header or a portion of the header of the respective network packet, and a payload part that comprises the remaining rest of the respective network packet including the network packet's payload, and to forward to the network service functions only the header parts of the network packets.

According to the invention it has been recognized that many cases of processing network packets by a number of (i.e. one or more) network service functions

involve an overhead in terms of transferring unnecessary information contained in the network packets to the network service functions and also between the network service functions (in case of a chain of NFs, where the network packets are forwarded hop-wise from one NF to a subsequent NF of the chain).

Network packets, being the objects that are processed by network service functions, consist of headers carrying routing/delivery-relevant information and payload data relevant for the receiving application. Payload data usually represents the bigger share of the entire network packet data. Even though many network functions like header mangling require only the information residing in the header, conventionally, the complete packet including header and payload is passed between all network functions on the packet's path. The number of network functions working on header data only is expected to increase, as the percentage of encrypted data (which takes an increasing part of the overall Internet traffic with the proliferation of https, QUIC and others) is growing and more and more network functions are not able to retrieve useful information from the (encrypted) payload data. Hence, a lot of data is unnecessarily moved between network functions.

If multiple NFs reside on the same physical host, zero copy schemes can be applied, as described above. Passing header and (unneeded) payload data is therefore not an issue in this case. However, in deployments such as NFV implementations, where network functions do not reside on a single physical host, but are distributed across multiple physical nodes, network packets need to be transferred between the different physical nodes. The packet transfer incurs overhead in terms of transmission time, latency, and processing. As payload data takes up the larger part of packets, most overhead is incurred by transferring that payload, even though network functions will not be looking at that data. Therefore, in accordance with embodiments of the present invention, a portion of a network packet including at least the network packet's payload (possibly together with a certain portion of the packet header) is cut off from the packet, and only the remaining rest of the network packet is actually transferred to and across the relevant network functions. Consequently, the present invention provides an efficient method and apparatus of getting rid of the overhead of unnecessarily

transferring packet payload data. Implementations of the invention in connection with NFV deployments present a use case for dynamic virtual function acceleration (DVFA).

Aspects of the invention extend to corresponding systems, methods, and computer program products such as computer readable storage media having instructions stored thereon which are operable to program a programmable processor to carry out a method as described in the aspects and embodiments set out below or recited in the claims and/or to program a suitably adapted computer to provide the apparatus recited in any of the claims.

According to an embodiment, after the network packets have been processed by the number of network service functions, a merger module may re-assemble the payload parts with the corresponding header parts. The merger module may be collocated with the splitter module. In particular, both modules may reside together on the ingress node for the number of network service functions. According to an embodiment, during the time period between the packet splitting and the packet merging, i.e. basically during the time the network packets are being processed by the number of network service functions, the network packets' payload parts may be stored in a memory component of the splitting module.

According to an embodiment the separated header parts may be forwarded to the number of network service functions with a payload identifier incorporated therein. This payload identify may be used by the merger module in order to perform proper re-assembly of the network packets.

According to an embodiment the separated header parts may be forwarded to the number of network service functions with information on the network packet's original payload size incorporated therein. This information may also be used by the merger module in order to perform proper re-assembly of the network packets. Furthermore, this information may be used for supporting legacy network service functions, i.e. network service functions that expect/require receiving full network packets. For instance, such legacy network service functions, upon receipt of a separated header part, may complement the header part to a full network packet

by appending to the header part a dummy payload block. According to an embodiment, such dummy payload blocks, which may contain random data or any other predefined pseudo data, may be cached at the network service function. In particular, legacy network service function may be pre-provisioned with a pool of cached dummy payload blocks of different lengths. In this case the checksum for multiple different lengths of the cached dummy payload blocks may be pre-computed already.

According to an embodiment the above-mentioned payload identifier may be merged into the appended dummy payload block. This may be particularly useful in case of the involvement of legacy network service functions in order to ensure that the payload identifier survives transformations of the header parts possibly performed by such legacy network service functions.

According to an embodiment it may be provided that splitting of network packets is only executed after having performed a check of whether each of the number of network service functions processes network packets by inspecting and/or modifying only the network packets' header information. For instance, in case any of the number of network service functions performs IDS (Intrusion Detection System) related activities, splitting would be omitted since such network service functions require the network packets' payload in order to perform their tasks.

According to an embodiment it may be provided that the splitting of network packets is performed in such a way that the header parts comprise all those portions of the network packets' headers that are to be processed by any of the number of network service functions. Generally, it should be noted that the size of the header information depends on the protocols used and the requirements of the network service functions involved. Therefore, in case of TCP/IP packets, for instance, the split may be performed in such a way that only the IP header information is separated into the header part, while the TCP header information is assigned to the payload part.

There are several ways how to design and further develop the teaching of the present invention in an advantageous way. To this end it is to be referred to the

dependent patent claims on the one hand and to the following explanation of preferred embodiments of the invention by way of example, illustrated by the drawing on the other hand. In connection with the explanation of the preferred embodiments of the invention by the aid of the drawing, generally preferred embodiments and further developments of the teaching will be explained. In the drawing

Fig. 1 is a schematic view illustrating the basic principle of a packet handling method used in an ingress node for a number of network service functions in accordance with an embodiment of the present invention,

Fig. 2 is a schematic view illustrating a splitting process of a TCP/IP packet in accordance with an embodiment of the present invention, and

Fig. 3 is a schematic view illustrating a packet handling apparatus in accordance with an embodiment of the present invention.

Fig. 1 schematically illustrates the basic principle of a packet handling method and apparatus for network service functions in accordance with an embodiment of the invention. The illustrated scenario includes a node 1 that functions as ingress point to a service function chain, SFC. Exemplarily, the SFC includes three (virtual or physical) network functions 2, denoted NF1, NF2 and NF3. The ingress node 1 may be part of a datacenter that operates the network functions 2.

As shown in Fig. 1, a splitter module 3 and a merger module 4 are collocated at the ingress node 1. Incoming network packets 5 received at the ingress node 1 are first analyzed whether they – in combination with the network functions 2 of the SFC that has to be passed by the respective network packets 5 – are suitable for split packet processing. Such analysis can be performed, e.g., on a per-packet basis or, more efficiently, on a per-flow basis. Generally, network functions that only process (e.g., inspect and/or modify) the header information of a network packet can be qualified as network functions suitable for split packet processing. From a packet perspective, in general, network packets carrying encrypted

payload are suitable, since (almost) any processing on the payload is impossible due to its encryption.

If suitability can not be conceded, the network packets 5 received at the ingress node 1 will not be processed by splitting module 3. Instead, the network packets 5 will be handled conventionally, i.e. they will be directly forwarded to the first network function 2 of the SFC, i.e. NF1 in Fig. 1.

However, if suitability is given, the network packets 5 will be processed by the splitter module 3, which splits the network packets 5 into a header part 6 and into a payload part 7. While the header part 6 includes the header or at least a portion of the header of a network packet 5, the payload part 7 includes the remaining rest of the network packet 5, i.e. the network packet's payload together with the part of the header not included in the header part 6.

The accurate decision of where exactly to perform splitting of a network packet 5 can be made depending on the protocols used and/or the requirements of the NFs involved. In any case, the more header information is contained within the header part 6, the more effective is the invention in terms of bandwidth saving, latency improvement and transmission time reduction. On the other hand, separated header part 6 needs to cover all those header information which is to be processed by the NFs.

In this context, it should be noted that generally the size of a network packet's header depends on the used protocol. For instance, Fig. 2 shows the structure of a TCP/IP network packet including the IP header, the TCP header and the network packet's payload. In this case, if the network functions of a SFC, which has to be passed by such TCP/IP network packets, perform processing solely of the IP header, the splitting could be performed in such a way that the IP header is included in the header part 6, while the TCP header together with the network packet's payload is included in the payload part 7 (splitting option 'A'). On the other hand, if the SFC contains at least one network function that performs any processing with respect to the TCP header, the splitting would be performed such that both the IP header and the TCP header are included in the header part 6,

while only the network packet's payload is included in the payload part 7 (splitting option 'B'). Of course, as will be easily appreciated by those skilled in the art, other splitting options can be applied likewise, including splitting options that assign one part of a particular header to the header part 6, while another part of the same header is assigned to the payload part 7. For the IP header, this is exemplarily illustrated by splitting option 'C'.

In any case, as shown in Fig. 1, only the header part 6 is forwarded into the NF chain, where the header part 6 is successively processed by NF1, NF2 and NF3. Consequently, the overhead in terms of transmission time, latency, and processing incurred by the packet transfer from the ingress node 1, via the network functions 2 and back to the ingress node 2 is significantly reduced compared to conventional processing where the entire packet including the payload data (which typically makes up the larger part of a network packet) is transferred along the path.

As indicated in Fig. 1, the payload part 7 is taken by the splitter module 3 and transferred to the merger module 4. Since in the scenario illustrated in Fig. 1 both entities are co-located, this transfer may be realized by a simple passing of a reference. However, if both modules 3, 4 do not reside on the same physical node, the payload needs to be sent over a network. Therefore, implementations in real deployments should seek to locate splitter module 3 and merger module 4 as close together as possible.

In order to enable proper re-assembly of the network packets 5 after NF processing, the header part 6 is forwarded into the NF chain with a payload identifier incorporated therein. This payload identifier can be used to locate the corresponding payload part 7 at the merger module 4.

If any of the NF 2 of the SFC is not adapted to receive the header part only (such as in the case of a legacy NF), a filtering process on the respective NF node may re-assemble the header part 6 with a (cached) payload block 8, e.g. containing random data. This process, which in Fig. 1 is assumed to apply with respect to the last NF 2 of the SFC (i.e. for the network function denoted 'NF3' in Fig. 1), may be executed by, e.g., a smart network interface controller, NIC, on the NF node or by

a function inside the NF node's operating system. In case of virtual NFs this process could also be executed by a function inside the hypervisor. It should be noted that a pool of cached payload blocks might only be provisioned in case of legacy NFs that expect to receive full network packets 5.

While the cached payload may consist of the maximum length possible, packet length and the checksum have to be adjusted in the header part 6 according to the incoming packet 5. The checksum for the payload part 7 could be already pre-computed for multiple different lengths and may then need only a small adjustment. In this embodiment, the splitter module 3 is preferably configured to incorporate information on the original size of the payload of a network packet 5 into the separated header part 6 of this packet 5.

With respect to a proper reunification or reassembly of network packets 5 at the merger module 4, it should be noted that the payload identifier incorporated into the header part 6, as described above, needs to survive any transformations done by the network functions 2 (or at least needs to be restored before passing a packet 5 on to the next hop of the SFC). This might be critical in case of legacy NFs 2, since they, in general, will be neither aware of the network packet split nor of the provided payload identifier. One way to ensure that the payload identifier is properly maintained during a network packet's 5 passage through the SFC and that the merger module 4 is thus able to properly assemble header part 6 and payload part 7, is to merge the payload identifier, potentially cryptographically signed, into the locally attached dummy payload part 8, containing e.g. a random data block.

After a NF 2 has finished processing a network packet 5, the header part 6 of the packet 5 is forwarded to the subsequent NF 2 of the path or, as the last step, to the merger module 4. In case of a re-assembled packet an outgoing process on the NF 2 (executed, as already mentioned above, by, e.g., a smart NIC, a function inside the operating system, or a function inside the hypervisor) disassembles the network packet 5 and forwards the header part 6 only.

At the NF graph egress, the packet 5 is re-assembled by the merger module 4, the checksum in the header is fixed, and the packet 5 is routed to the next hop.

Fig. 3 illustrates a packet handling apparatus for network service functions in accordance with an embodiment of the present invention. It should be noted that, for instance, this apparatus could function as the ingress node 1 of Fig. 1.

As depicted in Fig. 3, the main components of the apparatus include a processor 101 and a memory 102.

When being in operation, incoming packets received by the apparatus from a network via an input port 103 will be processed by a network interface controller, NIC, 104. The controller 104 may perform its operation either by cooperation with the processor 101, or by using its own processing capabilities (in case of so called smart NICs). In any case, the operation performed includes splitting of incoming network packets into two packet parts: a first packet part, denoted header part, comprising the header or a portion of the header of the respective network packet, and a second packet part, denoted payload part, comprising the remaining rest of the respective network packet, in particular the network packet's payload. For example, splitting can be performed by the controller 104 determining the starting address and length of a packet's TCP/IP header and starting address and length of the packet's payload.

While the controller 104 writes the payload parts into memory 102, the header parts are processed separately. Specifically, the controller 104 incorporates a payload identifier into each header part for the purpose of later reunification of network packets, and forwards each header part to processor 101. The processor 101 is configured to analyze the header parts and to identify the relevant SFC for each network packet. Accordingly, processor 101 forwards the packets via a respective one of controllers 105 and an associated output port 106. In case the illustrated apparatus functions as an ingress node of a datacenter offering an NFV deployment, it is assumed that the respective network service function of a particular SFC can be reached via the datacenter's internal bus system 107.

Once a header part has been processed by the relevant network service functions and is received back at the processor 101, either the processor 101 itself or the controller 104 reassembles the network packets by extracting the payload identifier from the header apart and by retrieving the corresponding payload part from memory 102. The reassembled network packet is then outputted via port 103.

Many modifications and other embodiments of the invention set forth herein will come to mind the one skilled in the art to which the invention pertains having the benefit of the teachings presented in the foregoing description and the associated drawings. Therefore, it is to be understood that the invention is not to be limited to the specific embodiments disclosed and that modifications and other embodiments are intended to be included within the scope of the appended claims. Although specific terms are employed herein, they are used in a generic and descriptive sense only and not for purposes of limitation.

# Claims

1. A method, comprising:

splitting network packets (5) received at an ingress node (1) for a number of network service functions (2) into two packet parts, including a header part (6) that comprises the header or a portion of the header of the respective network packet (5), and a payload part (7) that comprises the remaining rest of the respective network packet (5) including the network packet's (5) payload, and

passing on to the network service functions (2) only the header parts (6) of the network packets (5).

2. The method according to claim 1, further comprising:

re-assembling the payload parts (7) with the corresponding header parts (6) after the network packets (5) have been processed by said number of network service functions (2).

3. The method according to claim 1 or 2, comprising:

forwarding the header parts (6) to said number of network service functions (2) with a payload identifier incorporated therein, and

using the payload identifier for proper re-assembly of network packets (5).

4. The method according to any of claims 1 to 3, comprising:

forwarding the header parts (6) to said number of network service functions (2) with an information on the network packet's (5) original payload size incorporated therein.

5. The method according to any of claims 1 to 4, comprising:

at those of the number of network service function (2) that require receiving full network packets, complementing the header parts (6) to full network packets by appending to each header part (6) a dummy payload block (8) containing random data.

6. The method according to any of claims 1 to 5, comprising:

pre-provisioning those of the number of network service function (2) that require receiving full network packets with a pool of cached dummy payload blocks.

7. The method according to claim 6, comprising:

pre-computing the checksum for multiple different lengths of said cached dummy payload blocks.

8. The method according to any of claims 3 to 7, comprising:

merging said payload identifier into the appended dummy payload block (8).

9. The method according to any of claims 1 to 8, wherein the splitting of network packets (5) is only executed after having performed a check of whether each of said number of network service functions (2) processes network packets (5) by inspecting and/or modifying only the network packets' (5) header information.

10. The method according to any of claims 1 to 9, wherein the splitting of network packets (5) is performed in such a way that the header parts (6) comprise all those portions of the network packets' (4) headers that are to be processed by any of said number of network service functions (2).

11. An apparatus, in particular for executing a method according to any of claims 1 to 10, the apparatus comprising a splitter module (3) that is configured

to split network packets (4) received at an ingress node (1) for a number of network service functions (2) into two packet parts, including a header part (6) that comprises the header or a portion of the header of the respective network packet (4), and a payload part (7) that comprises the remaining rest of the respective network packet (4) including the network packet's (4) payload, and

to forward to the network service functions (2) only the header parts (6) of the network packets (4).

12. The apparatus according to claim 11, further comprising merger module (4) that is configured to re-unify the payload parts (7) with the corresponding header parts (6) after the network packets (4) have been processed by said number of network service functions (2).

13. The apparatus according to claim 11 or 12, wherein said splitter module (3) and said merger module (4) are collocated on said ingress node (1).

14. The apparatus according to any of claims 11 to 13, further comprising a memory component (102) configured to store network packets' (4) payload part (7) for the time the network packets (4) are being processed by said number of network service functions (2).

15. A computer-readable medium having stored thereon a computer program product comprising instructions to cause the apparatus of claim 11 to execute a method according to any of claims 1 to 10.

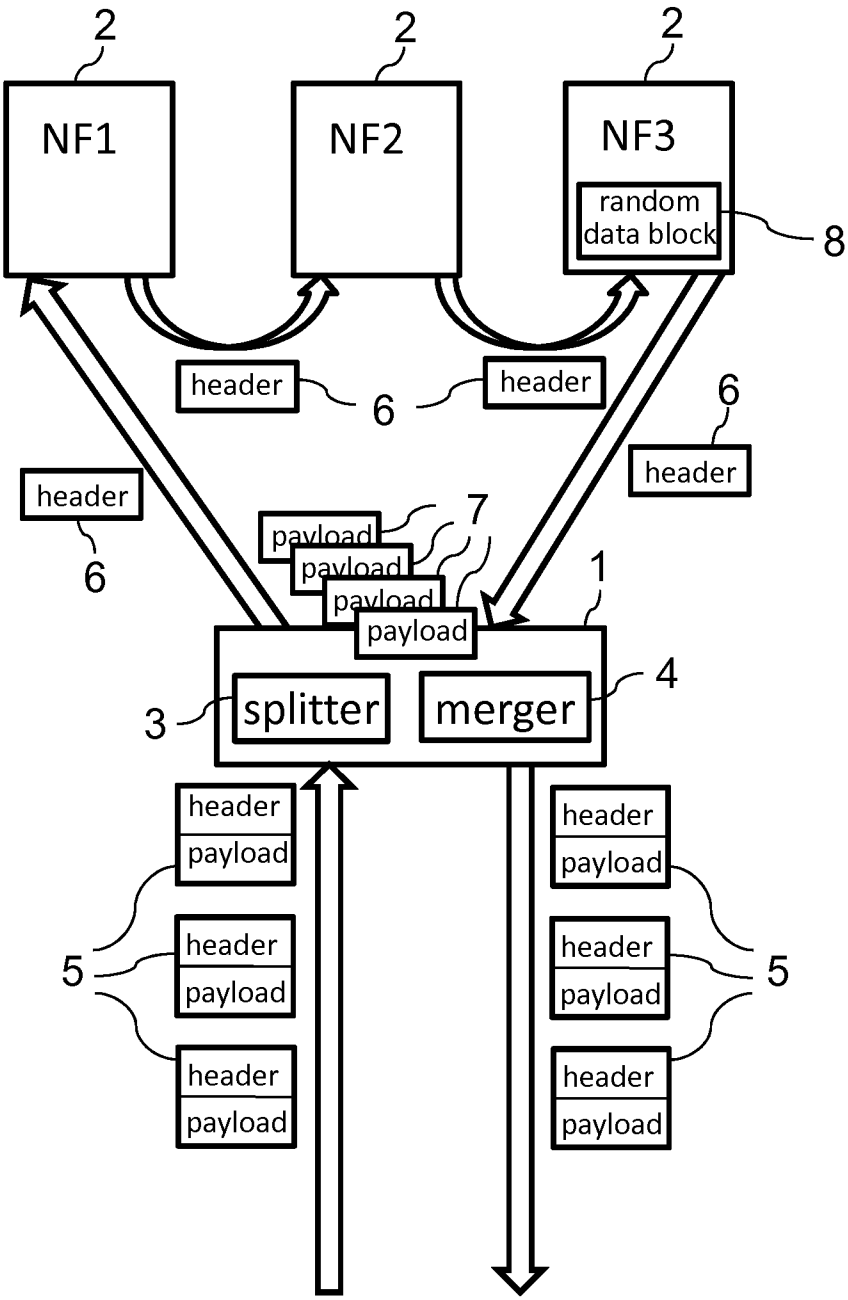

2
2
2
NF1
NF2
NF3
random data block
8
header
6
header
6
header
6
header
6
payload
payload
payload
payload
7
1
splitter
merger
3
4
header
payload
header
payload
header
payload
5
header
payload
header
payload
header
payload
5

Fig. 1

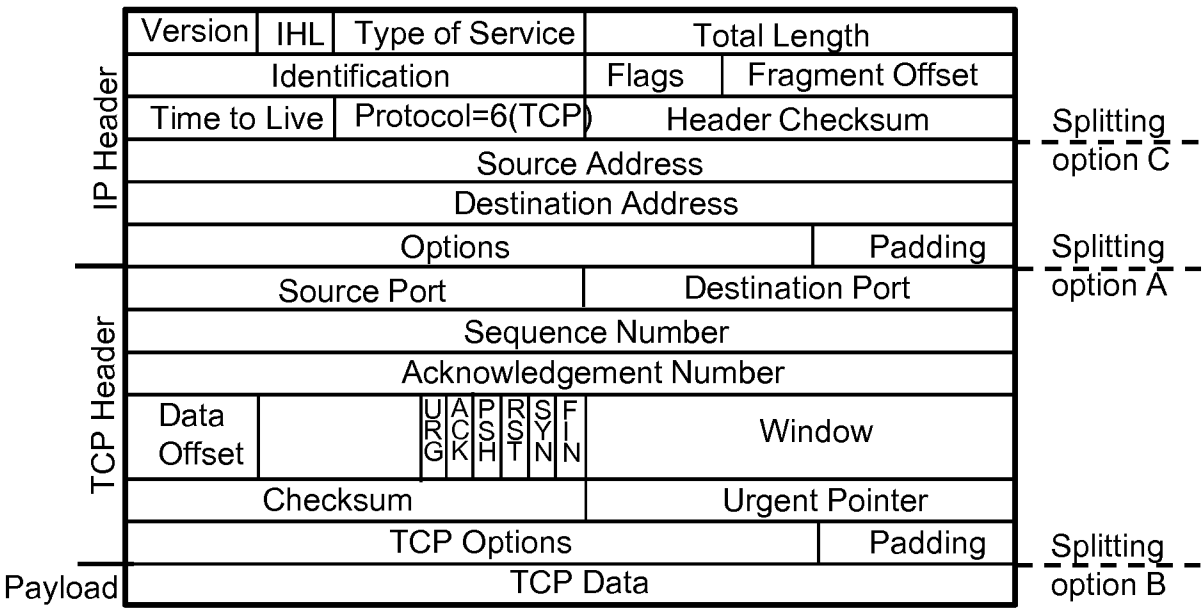

IP Header
TCP Header
Payload
Version
IHL
Type of Service
Total Length
Identification
Flags
Fragment Offset
Time to Live
Protocol=6(TCP)
Header Checksum
Source Address
Destination Address
Options
Padding
Source Port
Destination Port
Sequence Number
Acknowledgement Number
Data Offset
URG
ACK
PSH
RST
SYN
FIN
Window
Checksum
Urgent Pointer
TCP Options
Padding
TCP Data
Splitting option C
Splitting option A
Splitting option B

Fig. 2

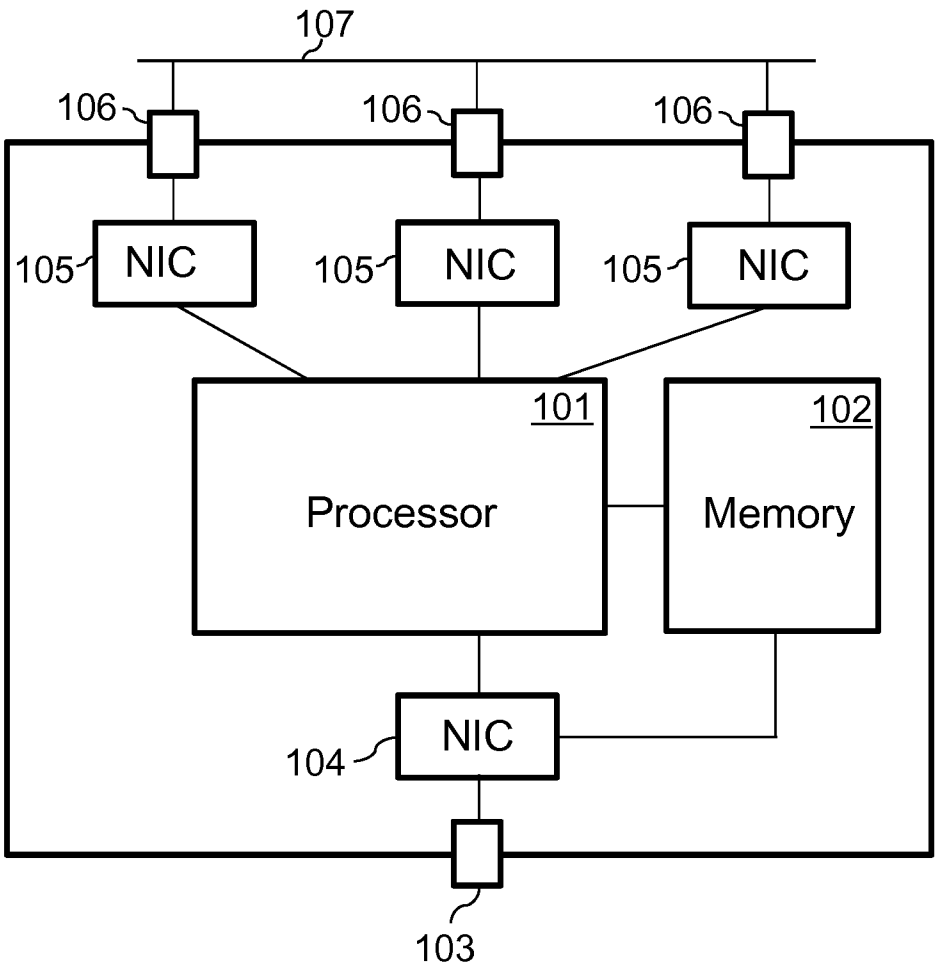

107
106
106
106
105
NIC
105
NIC
105
NIC
101
Processor
102
Memory
104
NIC
103

Fig. 3

# INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2017/058703

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/725 H04L12/841
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | WO 03/091857 A2 (TRANSWITCH CORP [US]; DEFORCHE KOEN [BE]; VERBRUGGEN GEERT [BE]; DE CO) 6 November 2003 (2003-11-06)<br>page 8, paragraphs second, third; figures 1a, 1b<br>page 10, paragraphs second, third<br>page 1, paragraph third<br>----- | 1-15 |

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 1 December 2017 | 14/12/2017 |

| Name and mailing address of the ISA/ | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040,<br>Fax: (+31-70) 340-3016 | Tsuchiya, Kuni |

2

# INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2017/058703

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| WO 03091857 A2 | 06-11-2003 | AU 2003228900 A1 | 10-11-2003 |
| | | CN 1663188 A | 31-08-2005 |
| | | EP 1523829 A2 | 20-04-2005 |
| | | WO 03091857 A2 | 06-11-2003 |