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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 November 2017 (16.11.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/196216 A1

(51) International Patent Classification:
H04L 12/26 (2006.01)

(21) International Application Number:
PCT/SE2016/050430

(22) International Filing Date:
12 May 2016 (12.05.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **TELEFONAKTIEBOLAGET LM ERICSSON (PUBL)** [SE/SE]; ., 164 83 Stockholm (SE).

(72) Inventors: **MORADI, Farnaz**; Lindhagensterassen 27, 112 18 Stockholm (SE). **FLINTA, Christofer**; Sköldgatan 4, 118 63 Stockholm (SE). **MEIROSU, Catalin**; Järvstigen 6, 170 71 Solna (SE). **JOHNSSON, Andreas**; Greta Arwidssons Väg 20, 754 43 Uppsala (SE).

(74) Agent: **SJÖBERG, Mats**; Ericsson AB, Patent Unit Kista, RAN Implementation & Core (PU-KRIC), 164 80 Stockholm (SE).

(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, 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, KN, KP, KR, 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).

(54) Title: A MONITORING CONTROLLER AND A METHOD PERFORMED THEREBY FOR MONITORING NETWORK PERFORMANCE

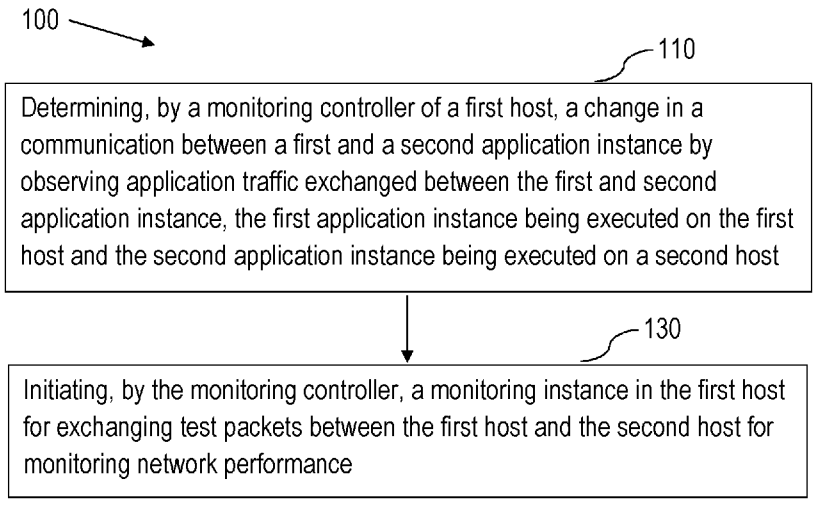

Fig. 1a

(57) Abstract: A monitoring controller and a method for monitoring network performance are provided. The comprises determining (110), by a monitoring controller of a first host, a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The method further comprises initiating (130), by the monitoring controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

WO 2017/196216 A1

[Continued on next page]

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *of inventorship (Rule 4.17(iv))*

Published:

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

A MONITORING CONTROLLER AND A METHOD PERFORMED THEREBY FOR MONITORING NETWORK PERFORMANCE

Technical field

[0001] The present disclosure relates to monitoring of network performance and in particular to monitoring of network performance in a network comprising virtualised compute and network resources.

Background

[0002] In a cloud environment, monitoring the network performance as perceived by applications and e.g. Virtual Network Functions, VNFs, running inside virtual machines or containers is important for identifying performance degradations, which may lead to violation of service level agreements. Monitoring the network performance may also be important for conducting root-cause analysis.

[0003] Application instances which run in virtual machines or containers may dynamically and frequently be started, scaled, migrated, and stopped. Therefore, keeping track of the lifecycle events of the application instances is required for automatic and real-time network performance monitoring.

[0004] Network monitoring may be performed using active or passive measurement methods. In active measurements, test packets are injected from a probe (aka a measurement point, MP) in the network and are received in another probe. The test packets may be collected in the receiver probe or reflected back to the sender probe. The test packets may be time stamped in both probes at sending and arrival events. In this way it is possible to study interaction between test packets (which may also be referred to as probe packets) and the cross traffic and draw conclusions about the network characteristics and the cross traffic dynamics. This approach is often called “active monitoring”. In passive measurement, in contrast, network traffic is being observed without injecting additional test packets.

[0005] Both active and passive monitoring methods may be used for network performance measurement and troubleshooting of performance degradations. Examples of active monitoring tools include ping (Internet Control Message Protocol (ICMP)/ Two-Way Active Measurement Protocol (TWAMP)/ traceroute, iperf, netperf, etc. For many active measurement methods, a monitoring process is required on both the sender and the receiver side for generating test packets, time stamping, or reflecting the test packets.

[0006] When network degradation problems occur, existing monitoring systems typically identify the degradation and trigger alarms. For troubleshooting and identifying the cause, active monitoring tools may be used to localise the problem. Such tools often require the source and destination IP address (or other type of address) of the hosts that run the applications experiencing the degradation.

[0007] For example, a service orchestrator, or network manager, may have the knowledge about which VNFs belong to a service chain and communicate with each other, so a monitor controller or management systems have to query the service orchestrator about e.g. the location, IP address, and required monitoring metrics. Since the application instances may dynamically and frequently change, e.g. due to scaling, this information has to be continuously updated.

[0008] When a network problem such as performance degradation happens, the source and destination addresses might even not be known for a monitoring and troubleshooting tool to be launched.

[0009] A centralised monitoring controller/manager needs to have an interface to the service orchestrator to obtain information about the VNF instances or applications that communicate with each other to be able to instantiate and setup the required monitoring functions next to them. Such interface creates a significant overhead for the service orchestrator in case of frequent scale-in / scale-out and VNF migration processes. It also requires standardisation (or, alternately, continuous systems integration and testing) in order to guarantee that

a monitoring controller developed by one vendor is able to communicate with an orchestrator developed by a different vendor.

[00010] One problem with some existing network monitoring tools is that they require a lot of information to be generated and stored, which puts a significant load on network resources for computing, storing and analysing the information. Another problem is that many existing tools are directed to one specific execution environment.

Summary

[00011] The object is to obviate at least some of the problems outlined above. In particular, it is an object to provide a monitoring controller and a method for monitoring network performance. These objects and others may be obtained by providing a monitoring controller and a method performed by a monitoring controller according to the independent claims attached below.

[00012] According to an aspect, a method for monitoring network performance is provided. The method comprises determining, by a monitoring controller of a first host, a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The method further comprises initiating, by the monitoring controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00013] According to an aspect, a monitoring controller of a first host for monitoring network performance is provided. The monitoring controller is configured for determining a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host; and for initiating a monitoring instance in the first host for

exchanging test packets between the first host and the second host for monitoring network performance.

[00014] The monitoring controller and the method may have a plurality of possible advantages. One possible advantage is that the monitoring function may be distributed within local hosts, wherein no centralised monitoring function may be needed. Another possible advantage is that the solution provides distributed zero-touch and dynamic instantiation and configuration of active monitoring functions when required based on passively observed application instance communication.

Brief description of drawings

[00015] Embodiments will now be described in more detail in relation to the accompanying drawings, in which:

[00016] Figure 1a is a flowchart of a method for monitoring network performance, according to an exemplifying embodiment.

[00017] Figure 1b is a flowchart of a method for monitoring network performance, according to a further exemplifying embodiment.

[00018] Figure 1c is a flowchart of a method for monitoring network performance, according to yet an exemplifying embodiment.

[00019] Figure 1d is a flowchart of a method for monitoring network performance, according to still an exemplifying embodiment.

[00020] Figure 2a is a block diagram of a first and second host of a data centre.

[00021] Figure 2b is a flowchart of a method for monitoring network performance, according to a further exemplifying embodiment.

[00022] Figure 2c is a flowchart of a method performed by a monitoring controller on a receiving side for monitoring network performance, according to an exemplifying embodiment.

[00023] Figure 2d is a signalling diagram or sequence illustrating signalling and communication within a sender host, e.g. a first host, between the sender host and a receiver host, e.g. a second host, and within the second host.

[00024] Figure 3 is a block diagram of a monitoring controller for monitoring network performance, according to an exemplifying embodiment.

[00025] Figure 4 is a block diagram of a monitoring controller for monitoring network performance, according to another exemplifying embodiment.

[00026] Figure 5 is a block diagram of an arrangement in a monitoring controller for monitoring network performance, according to an exemplifying embodiment.

Detailed description

[00027] Briefly described, a monitoring controller and a method are provided for monitoring network performance. The monitoring controller is running on, or is executed by a first host. The first host also comprises an application instance which may communicate with other application instances on other hosts. The monitoring controller on the first host may detect a change in communication between an application on the first host and an application on a second host by observing application traffic exchanged between the application instance on the first host and an application instance on a second host. Once such a change in communication is detected, the monitoring controller may initiate a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00028] Embodiments herein relate to a method for monitoring network performance. Embodiments will now be described with reference to **figures 1a-1d**. Figure 1a illustrates the method 100 comprising determining 110, by a monitoring controller of a first host, a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The method further comprises initiating 130, by the monitoring

controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00029] The monitoring controller is comprised in, or running on, the first host. The first host may be a physical host or a virtual host. The first host may further comprise one or more application instances, i.e. at least the first application instance, wherein any application instance of the first host may communicate dynamically with other application instances on other host, e.g. the second application instance on the second host. The second host may also be a physical host or a virtual host. By the first host comprising one or more application instances means that the first host is executing the one or more application instances of the first host, or that the one or more application instances is/are running on the first host.

[00030] The monitoring controller of the first host may tap in on communications, e.g. packets from the second application instance on the second host being received by the first host destined for the first application instance and/or packets being transmitted from the first host destined for the second application instance on the second host. By tapping in on communication, also referred to as tapping traffic, received by the first host and sent from the first host, the monitoring controller may determine any change in communication between the first and the second application instance. The tapping in is an example of observing said communication. There may be other examples of observing the traffic as will be described below.

[00031] Once the monitoring controller determines that there is a change in the communication between the first and the second application instance, the monitoring controller may initiate the monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance. Exchanging test packets means that test packets are sent between the first or the second host, e.g. the monitoring instance of the first host may transmit test packets to a corresponding monitoring instance of the second host and/or vice versa. The monitoring controller of the first host may thereby observe the performance of the network between the first and the second host.

[00032] The method may have a plurality of possible advantages. One possible advantage is that the monitoring function may be distributed within local hosts, wherein no centralised monitoring function may be needed. Another possible advantage is that the solution provides distributed zero-touch and dynamic instantiation and configuration of active monitoring functions when required based on passively observed application instance communication. Still a possible advantage is that the monitoring controllers run in a distributed fashion, wherein passive communication observation and possible associated data collection may be performed locally, so the monitoring controller may obtain timely and accurate information about which local application instances are communicating with which remote application instances. This provides another possible advantage that the monitoring function may be deployed without introducing an interface towards a service orchestrator. Yet another possible advantage is that IP address changes could be detected locally (for example, by inspecting Internet Protocol IP, packets, open and close connection packets, or by rapidly aging out an IP address that saw no traffic addressed to it in a short time interval) and therefore a local traffic matrix better reflects the actual situation in the first host or the network connecting the first and the second host. This also reduces the need for integration of the orchestrator or network management system with third-party IP address management tools, thus providing both OPEX and potentially CAPEX savings. Still a possible advantage is that the monitoring of exchanging test packets may be initiated and stopped dynamically and when needed. It is also possible to check if there is already active monitoring ongoing between two physical hosts. If a new pair of applications needs to be monitored between these hosts a new monitoring session is not needed and instead the measurements from the ongoing session may be used. This reduces the amount of test packets that are sent into the network for active measurements. This also means that the method may scale well and may also be used in cloud environments consisting of multiple data centres.

[00033] In an example, the determining 110 that there is a change in the communication between two application instances may comprise obtaining a

notification that there is a change in the communication between two application instances.

[00034] When there is a change in communication, the monitoring controller may determine that by obtaining the notification of the change. For example, there may be a monitoring controller in the second host that has detected the change in traffic and sends a notification to the monitoring controller of the first host.

[00035] In this manner, the monitoring controller of the first host is made aware that there is a change in traffic between the two application instances, i.e. the first application instance being executed on the first host and the second application instance being executed on the second host.

[00036] In another example, the method further comprises receiving traffic exchanged between the first and a second virtual application instance, wherein the traffic is tapped 105 from a virtual switch comprised in the first host.

[00037] The first host may comprise a virtual switch, which may send a copy of all traffic, e.g. packets being switched within the first host, to a passive monitoring container which may detect the change and then inform the monitoring controller. By packets being switched within the first host means packets being received by the first host destined for the first application instance and/or packets being outputted from the first application instance and destined for any application instance on any other host. Tapping the traffic from the virtual switch may have additional advantages. For example, it is a resource-optimised way of obtaining network information. Further, the tapped data may be analysed and will hence not affect the performance of the data flow. Still further, tapping the traffic from the virtual switch is a lightweight solution when performed within one host.

[00038] **Figure 1b** is a flowchart of the method 100 for monitoring network performance, according to an exemplifying embodiment. The method 100 for monitoring network performance is performed in a first host system, the first host system comprising a processing device providing a virtual application instance platform, a virtual switch and a monitoring controller. In this embodiment,

illustrated in figure 1b, the method comprises tapping 105, by the virtual switch, traffic exchanged between a first and a second virtual application instance, the first virtual application instance being executed on the first host system and the second virtual application instance being executed on a second host system. The method is also illustrated comprising determining 110' , by the monitoring controller, the change in the communication between the first and the second virtual application instance and then initiating 130, by the monitoring controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00039] In this exemplifying embodiment, the first host is referred to as the first host system and likewise the second host is referred to as the second host system. As will be explained in more detail below, the first and second host system may each comprise a processing device. The processing device provides the virtual application instance platform, a virtual switch and a monitoring controller. The virtual switch may receive traffic, e.g. in the form of packet(s), received from the second host and destined to the first application instance. The virtual switch may also receive traffic, e.g. in the form of packet(s), received from the second application instance and destined to the second application instance on the second host system. The virtual switch may be tapped 105, either by the monitoring controller or by e.g. a passive monitoring instance in order to "listen to" traffic involving the first application instance and thereby detecting any change in the traffic between the first and the second application instance. The monitoring controller may thus determine 110' (either by itself tapping the virtual switch or being notified from the passive monitoring instance) that there is a change in the communication between the first and the second virtual application instance and initiate 130 the monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance. The monitoring controller may instruct a virtual switch to tap traffic exchanged between the two application instances.

[00040] The change may comprise any of a new communication between the two application instances; that an existing, previously active communication having

expired; a change in traffic intensity; a change in traffic pattern; and a change of communication protocol.

[00041] There are many different examples of what a change might be. One example is the occurrence of a new communication between the first application instance on the first host and the second application instance on the second host. A new communication may comprise a traffic flow of packets being sent between the first application instance on the first host and the second application instance on the second host.

[00042] Another example is that an ongoing communication, i.e. an active communication, between the first application instance on the first host and the second application instance on the second host may change in one or more aspects. E.g. it may increase or decrease in intensity, meaning that e.g. the number and/or size of packets of the communication may change, the traffic flow may change to/from being relatively regular to being more erratic possibly indicating a problem in the network between the first host and the second host.

[00043] Still other examples of a change are that the communication expires or that the communication protocol that is being used is changed. Merely as an example, the communication protocol may change from TCP to user Datagram Protocol, UDP. Any change of communication protocol or above described changes may require the communication to be monitored, wherein the monitoring controller may initiate the monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00044] The method 100 may further comprise determining 120 that no previous monitoring is currently ongoing between the hosts with regard to the detected change, at least not with regard to a metric associated with the detected change in the communication between the first and the second application instance.

[00045] It may be the case that there is an ongoing monitoring instance, especially with respect to a specific metric, between the first host and the second

host. If there already is an ongoing monitoring between the first and the second host, there may be no need to initiate another monitoring instance. However, if no previous monitoring is currently ongoing between the hosts with regard to the detected change, then the monitoring controller may initiate the monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[00046] The detected change may be associated with a specific metric, e.g. capacity, latency, end-to-end delay, or end-to-end packet loss, wherein there may be an ongoing monitoring instance between the hosts with regard to one metric. If the detected change is related to another metric than the metric being monitored by the currently ongoing monitoring instance, then the ongoing monitoring instance may either be changed to include monitoring the communication also with regard to the metric associated with the detected change; or alternatively the monitoring controller may initiate a new monitoring instance with regard to the metric for which the change has been detected.

[00047] In an example, illustrated in **figure 1c**, the initiating 130 of the monitoring instance in the first host for transmitting test packets from the first host to the second host comprises obtaining 131 a software image required for starting one or more monitoring instances in the first host.

[00048] The software image is representative of the type of monitoring that is to be performed. In some cases the active monitoring software is not available at the host, hence a software image may be retrieved from a software image repository (such as a Docker repository). The software image is downloaded and installed where needed. The container is then started and stopped according to the method explained in the application.

[00049] In a further example, illustrated in **figure 1c**, the method 100 further comprises requesting 132 a monitoring controller of the second host to initiate a corresponding monitoring instance.

[00050] In order for the monitoring controller to perform monitoring, it may be necessary to have a monitoring instance in the second host. Then the monitoring instance of the first host and the monitoring instance of the second host may send test packets between themselves thereby evaluating network performance with regard to one or more metrics. The test packets may be time stamped in respective monitoring instance, they may be reflected back to the monitoring instance first transmitting it, the monitoring instances may exchange information between each other etc.

[00051] If no monitoring instance is running on, or is comprised in the second host, the monitoring controller of the first host may request the monitoring controller of the second host to initiate a monitoring instance on the second host.

[00052] In case there already is a monitoring instance running in the second host, the same monitoring instance may be used. The monitoring instance in the second host may e.g. receive test packets transmitted from the monitoring instance in the first host, time stamp the test packets and/or reflect the test packets back to the monitoring instance in the first host. By reflecting a test packet means that the monitoring instance in the second host transmits a corresponding test packet back to the monitoring instance in the first host from which the monitoring instance in the second host received them. Examples of packet reflecting tools and methods are ping and TWAMP.

[00053] The method 100 may further comprise executing 140 the monitoring between the two hosts.

[00054] Once the change is detected and the monitoring instances are initiated, the monitoring between the two hosts may be executed. The execution of the monitoring may comprise various actions as will be described below.

[00055] The executing 140 of the monitoring may comprise starting the monitoring instance.

[00056] The monitoring controller has initiated the monitoring instance in the first host for exchanging test packets between the first host and the second host for

monitoring network performance. Once this is done, the monitoring controller may start the monitoring instance in the first host, wherein the monitoring instance in the first host may transmit and/or receive test packets e.g. to/from the monitoring instance in the second host.

[00057] By analysing different characteristics associated with the transmission and/or reception of the test packet(s), the monitoring instance may evaluate different metrics of the network associated with the transmission and/or reception of the test packet(s). Either each test packet may generate monitoring information or two or more test packets may be analysed and/or evaluated to obtain monitoring information pertaining to network performance.

[00058] The method 100 may further comprise providing 145 monitoring information to a manager.

[00059] The monitoring may result in various monitoring information and/or results. The information may be provided continuously, regularly or at random/specific points in time. For example, as soon as monitoring information is obtain, e.g. after a test packets has been sent and reflected back or a test packet has been received such that there is information available to obtain monitoring information, the monitoring information may be transmitted to the manager. In another example, a number of packets have to be exchanged between the two monitoring instances in the first and second host respectively before any analysis or evaluation is performed to obtain monitoring information.

[00060] The monitoring information may be transmitted as soon as it is obtained, at regular time intervals, after a certain criterion is fulfilled etc.

[00061] The manager may then use the received monitoring information to take any appropriate action(s) that may be needed, or refrain to take any action if the received monitoring information so indicates.

[00062] The method 100 may yet further comprise stopping 150 the monitoring between the two hosts and providing 160 a result of the monitoring to the manager.

[00063] Once the monitoring between the two hosts is determined to be done, the monitoring controller may stop the monitoring. The monitoring information may have been reported continuously, randomly, regularly or not at all before the stopping of the monitoring. In case there is monitoring information which has not yet been provided to the manager, the monitoring controller may provide the not reported monitoring information to the manager, thereby providing the manager with a result of the monitoring.

[00064] The monitoring controller may also process the obtained monitoring information in order to produce a monitoring result to be provided to the manager.

[00065] In an example, application traffic is exchanged between virtualised applications, or wherein the application instances are virtualised applications.

[00066] Application traffic may be transmitted between the first application instance and the second application instance. The first and/or the second application instance may be virtualised applications.

[00067] A physical device may run one or more virtualised machines or platforms upon which the application instance(s) is/are run, being virtualised applications. Also the monitoring controller and the monitoring instance(s) may be run in the virtualised machine(s) or platform(s), wherein the monitoring controller and the monitoring instance(s) are also being virtualised.

[00068] The monitoring controller, the two application instances and the monitoring instance(s) may correspond to individual containers, individual virtual machines or uni-kernels or similar technology.

[00069] These are examples of virtual implementations. Generally, a container-based implementation, also referred to as virtualisation engines, implies that the kernel of an operating system allows for multiple isolated user-space instances instead of just one. From “the outside”, they may look like a real server or physical device.

[00070] Uni-kernels are generally referred to as specialised single address space machine images implemented using library operating systems. Uni-kernels may run directly on e.g. a hypervisor or hardware without an intervening operating system.

[00071] A virtual machine is generally referred to as an emulation of a particular computer system. Virtual machines may operate based on computer architecture and functions of a real or hypothetical computer. Implementations of virtual machines may involve specialised hardware, software or a combination of both.

[00072] The method 100 may further comprise requesting resources to be reserved on the first host for a monitoring instance from a manager.

[00073] In order for the monitoring instance on the first host to be created, resources are needed on the host. Thus the monitoring controller may request the manager to reserve resources for the monitoring instance on the first host. The resources may be in terms of CPU, memory, disk utilisation etc.

[00074] The manager may then reserve the necessary resources on the first host, wherein the monitoring instance may be executed or run on the first host for performing various measurements as described above.

[00075] In case there is a need to create a monitoring instance on the second host, then the monitoring controller on the second host may request resources to be reserved on the second host for the monitoring instance from the manager.

[00076] **Figure 2a** illustrates the first host 210, and also other hosts such as the second host 220, comprising a virtual switch 216, 226, herein also referred to as a container bridge or a virtual machine bridge. In general, the virtual switch 216, 226 may receive communication, e.g. in the form of packets of a traffic flow, being received by the host 210, 220 destined for the application instance 211, 221 and forward the communication (e.g. packet(s)) to the application instance 211, 220. Further, the virtual switch 216, 226 may receive communication outputted from the application instance 211, 221 and forward the communication according to a destination address, e.g. to a Network Interface Controller, NIC, to be outputted

from the host 210, 220. Any type of communication to and/or from the application instance received by the virtual switch may be copied and forwarded to a passive monitoring instance, e.g. the first passive monitoring instance 212 in the first host 210 or the second passive monitoring instance 222 in the second host 220. In this manner, the passive monitoring instance 212, 222 may tap into the communication flowing through the virtual switch 216, 226.

[00077] The first and the second host 210, 220 also comprises a respective monitoring controller 214 and 224, which may control the first and the second passive monitoring instance 212, 222 as well as a first active monitoring instance 213 and a second active monitoring instance 223 respectively.

[00078] In the example illustrated in figure 2a, the first host 210 and the second host 220 are part of a data centre, wherein the network connecting the first host 210 and the second host 220 is referred to as a data centre network.

[00079] In other words, passive monitoring may run in a container or virtual machine and passively receive mirrored traffic to and from the application instance on that host. An example of an application instance is a Virtualised network Function, VNF. The VNF is defined by European Telecommunications Standards Institute, ETSI, see ETSI GS NFV 003 V1.1.1 (2013-10). Alternatively the passive monitoring may share the same network stack with the application instance containers or virtual machines, also referred to as VNF containers or VNF virtual machines. Another alternative is that the passive monitoring may run on the host machine, e.g. as part of the monitoring controller, listening to the network namespace of the application instances.

[00080] The passive monitoring instance 212, 222 may detect a change in traffic to/from the application instance 211, 221 and notify the monitoring controller 214, 224. In this manner, the monitoring controller 214, 224 may receive an event/message from the passive monitoring instance 212, 222 when there is a change in the user traffic that is passively being observed, thereby the monitoring controller 214 may determine 110 a change in a communication between a first and a second application instance by observing application traffic exchanged

between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. A change may e.g. correspond to a new flow or expiration of a previously active flow. Flow expiration may happen when the application instance 211, 212 has not sent or received any packets from a remote application instance for a pre-defined amount of time.

[00081] The monitoring controller 214, 224 may decide, for example based on operator policies, if a new flow needs to be actively monitored or not. Alternatively, if the passive monitoring instance 212, 222 already monitors some metrics e.g. packet loss, the monitoring controller 214, 224 may decide to initiate active measurement sessions to complement the passive measurements and obtain more troubleshooting data.

[00082] For example, if the monitoring controller 214 decides to actively monitoring the flow associated with the detected change, the monitoring controller 214 initiates130 a monitoring instance, i.e. the first active monitoring instance 213 in figure 2a, in the first host 210 for exchanging test packets between the first host 210 and the second host 220 for monitoring network performance.

[00083] For many types of active monitoring, both a sender monitoring instance and a receiver/reflector monitoring instance may be required. If a reflector monitoring instance is not present on the receiver side e.g. on the second host 220, the monitoring controller 214 of the sender side, the first host 210, may request the remote the monitoring controller 224 to instantiate a reflector monitoring instance 221 for the active monitoring session. The monitoring controller 214 may also stop measurement sessions after a pre-determined time, or when network performance problem is resolved, or the path does not require monitoring anymore since the traffic flows have expired meaning that the traffic between the first application instance 211 and the second application instance 221 have expired or ended.

[00084] The passive monitoring, e.g. passive data collection, may be performed all the time, and active measurements are started only for monitoring the network

between application instances that communicate at each point in time. Therefore, while providing continuous monitoring of the system, the active test, or probe, packets are only sent on network paths on which monitoring have been requested. This may be particularly important for example when using high speed measurements with Data Plane Development Kit, DPDK, which takes full control of an interface and uses a full Central Processing Unit, CPU, core to receive measurements packets. Therefore, it may be important to be able to automatically start and stop senders and reflectors to avoid wasting compute and network resources.

[00085] During the monitoring, also referred to as measurement sessions, or at the end of the measurement sessions, the monitoring controller 214, 224 may communicate with higher management levels, a manager or an orchestrator sending monitored data/alarms based on passive and active measurements, as well as compute resource utilisation of containers or virtual machines which may be obtained locally, for example from Docker application Interface, API. The manager or orchestrator may then make decisions such as migration or scaling of the application, or VNF, instances experiencing network performance problems.

[00086] **Figure 2a** illustrates two hosts 210 and 220, datacentre network switches 231 and 232, a monitoring function software image repository 250, and a container/virtual machine management framework 240. In each host 210, 220, according to this example, a VNF or application instance 211 ,221, a monitor controller 214, 224, a passive monitoring instance 212, 222, an active monitoring instance 213, 223, a container bridge or virtual switch 216, 226, a container management system 215, 225 and a Network Interface Controller, NIC, exist. The container management 215, 225 may alternatively be located outside the hosts 210, 220. The VNF and monitoring containers run inside containers or virtual machines and are connected to the same bridge or virtual switch. The software image required for starting the monitoring containers may be obtained from the monitoring Function Software image repository by the container management system.

[00087] In this example, the first application instance 211 which is running in a container or virtual machine in the first host 210 is communicating with the second application instance container or virtual machine 221 in the second host 220. The passive monitoring containers or virtual machines 212 and 222 are already established in both hosts and passively observe copies of packets sent from or received by the corresponding application (VNF) instance which is being monitored, for example using port mirroring on the virtual switch or the container bridge 216, 226.

[00088] A monitor controller 214, 224 is running in each host as a process on the host machine or as a privileged container or virtual machine. The monitor controller 214, 224 may be responsible for automatic instantiation, configuration, and stopping of the passive and active monitoring instances 212, 213, 222 and 223. The monitor controller 214, 224 may get information about new flows, i.e. traffic to/from a new application instance, from the local passive monitoring 212, 222. If active monitoring is required for the new flow and the path that the flow traffic traverses is not already being monitored, the monitor controller 214, 224 may request from the container management system to instantiate an active monitoring container or virtual machine 213, 223. The monitor controller 214, 224 may also be responsible for providing the configuration required for performing active measurements for the active monitoring instances 213, 223 being instantiated.

[00089] Each monitoring controller 214, 224 may also communicate with monitoring controllers on the other hosts or servers in the datacentre to request for a new receiver/reflector active monitoring instance if required for the measurements or monitoring. Moreover, each monitoring controller 214, 224 may inform a remote monitoring controller 214, 224 to stop the receiver active monitoring container (or virtual machine) when it is not needed anymore in order to avoid waste of server resources. Monitoring controllers may potentially exchange messages with each other to identify which paths are being monitored, which monitor functions are associated with which application instances (e.g. VNFs) etc.

[00090] **Figure 2b** is a flowchart of a method performed by the monitoring controller according to an exemplifying embodiment. The monitoring controller waits for an event or message from the passive monitor, PM, about a new flow for the VNF/application which is being monitored.

[00091] Based on the information about the source and destination addresses of the VNFs that are part of the flow, the monitoring controller, MC, should identify on which server the remote VNF instance is located. The mapping information about where the receiver VNF is running may be obtained in different ways. One alternative is that MCs populate a distributed database (e.g. a key/value storage) with mapping of each VNF Internet Protocol, IP, address and/or each host machine identifier. The mapping may also include information about the IP address of the monitoring functions that are associated with the VNF. Another alternative is that local MC is connected to other MCs in a peer-to-peer fashion or in a hierarchical scheme and may query from its peers. One more alternative is to query a logically centralised monitor controller.

[00092] The MC checks if the path between the local and remote host/server is already being actively monitored or not. If there is already an active monitor, monitoring the required metric on the network between the sender and receiver host, there is no need for instantiating a new AM container.

[00093] If an AM container is not running on the local server, the MC starts an AM container by requesting the container management system.

[00094] If the required resources for starting the new AM container are not available, the MC receives a feedback and monitoring is stopped waiting for container(s) to be started.

[00095] If the active monitoring requires a receiver side reflector, the MC queries the remote MC to request for a remote reflector AM to be started.

[00096] After receiving the feedback about successful instantiation of the remote AM, the MC configures the AM to start the monitoring session. If the remote

container has failed to start, the MC also receives a feedback and can stop the sender AM container if required.

[00097] Optionally, the MC or another entity or container can perform some processing of the monitored data obtained from both passive and active monitors, for example in order to generate alarms if the network performance has degraded.

[00098] **Figure 2c** is a flowchart of a method performed by the monitoring controller on the second host, the receiver side, according to an exemplifying embodiment.

[00099] The MC waits for requests from remote MCs for monitoring functions. If a local AM does not exist, the MC creates an AM container by requesting the container management system.

[000100] The MC configures the requested monitoring function is the AM container (e.g. a reflector for Two-Way Active Measurement Protocol, TWAMP, which listens to a port).

[000101] The MC sends feedback with data about the AM's successful or failed start. In case of success, the IP and port addresses etc. are sent to the sender MC.

[000102] **Figure 2d** is a signalling diagram or sequence illustrating signalling and communication within a sender host, e.g. a first host, between the sender host and a receiver host, e.g. a second host, and within the second host.

[000103] If a new flow is observed by the PM of the sender host, the MC of the sender host receives an event from the PM and starts active monitoring if required. The PM should keep track of all active flows and should remove flows that have not been active, i.e. have not exchanged any packets within a pre-defined amount of time, to provide an up-to-date view of the application instance communications to the MC. When a flow expires, i.e. PM of the sender host has not seen any packets for that flow for the pre-defined amount of time, the PM sends an event to

MC of the sender host. MC then stops the active monitoring function to avoid waste of resources

[000104] Embodiments herein also relate to a monitoring controller of a first host for monitoring network performance. The monitoring controller has the same technical features, objects and advantages as the method described above. The monitoring controller will therefore be described only in brief in order to avoid unnecessary repetition. The monitoring controller will be described with reference to **figures 3** and **4**.

[000105] **Figure 3** illustrates the monitoring controller 300, 400 being configured for determining a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host; and for initiating a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[000106] The monitoring controller 300, 400 may be realised or implemented in various different ways. A first exemplifying implementation or realisation is illustrated in **figure 3**. **Figure 3** illustrates the monitoring controller 300 comprising a processor 321 and memory 322, the memory comprising instructions, e.g. by means of a computer program 323, which when executed by the processor 321 causes the monitoring controller 300 to determine a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The memory further comprises instructions, which when executed by the processor 321 causes the monitoring controller 300 to initiate a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[000107] **Figure 3** also illustrates the monitoring controller 300 comprising a memory 310. It shall be pointed out that figure 3 is merely an exemplifying illustration and memory 310 may be optional, be a part of the memory 322 or be a further memory of the monitoring controller 300. The memory may for example comprise information relating to the monitoring controller 300, to statistics of operation of the monitoring controller 300, just to give a couple of illustrating examples. Figure 3 further illustrates the monitoring controller 300 comprising processing means 320, which comprises the memory 322 and the processor 321. Still further, figure 3 illustrates the monitoring controller 300 comprising a communication unit 330. The communication unit 330 may comprise an interface through which the monitoring controller 300 communicates with other nodes or entities of the communication network as well as other communication units. Figure 3 also illustrates the monitoring controller 300 comprising further functionality 340. The further functionality 340 may comprise hardware or software necessary for the monitoring controller 300 to perform different tasks that are not disclosed herein.

[000108] An alternative exemplifying implementation of the monitoring controller 300, 400 is illustrated in **figure 4**. **Figure 4** illustrates the monitoring controller 400 comprising a determining unit 403 for determining a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The monitoring controller 400 further comprises an initiating unit 404 for initiating a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[000109] In **figure 4**, the monitoring controller 400 is also illustrated comprising a communication unit 401. Through this unit, the monitoring controller 400 is adapted to communicate with other nodes and/or entities in the wireless communication network. The communication unit 401 may comprise more than one receiving arrangement. For example, the communication unit 401 may be

connected to both a wire and an antenna, by means of which monitoring controller 400 is enabled to communicate with other nodes and/or entities in the wireless communication network. Similarly, the communication unit 401 may comprise more than one transmitting arrangement, which in turn is connected to both a wire and an antenna, by means of which the monitoring controller 400 is enabled to communicate with other nodes and/or entities in the wireless communication network. The monitoring controller 400 is further illustrated comprising a memory 402 for storing data. Further, the monitoring controller 400 may comprise a control or processing unit (not shown) which in turn is connected to the different units 403-404. It shall be pointed out that this is merely an illustrative example and the monitoring controller 400 may comprise more, less or other units or modules which execute the functions of the monitoring controller 400 in the same manner as the units illustrated in **figure 4**.

[000110] It should be noted that **figure 4** merely illustrates various functional units in the monitoring controller 400 in a logical sense. The functions in practice may be implemented using any suitable software and hardware means/circuits etc. Thus, the embodiments are generally not limited to the shown structures of the monitoring controller 400 and the functional units. Hence, the previously described exemplary embodiments may be realised in many ways. For example, one embodiment includes a computer-readable medium having instructions stored thereon that are executable by the control or processing unit for executing the method steps in the monitoring controller 400. The instructions executable by the computing system and stored on the computer-readable medium perform the method steps of the monitoring controller 400 as set forth in the claims.

[000111] In yet a further implementation, the first host comprises a processing device providing a virtual application instance platform, a virtual switch and a monitoring controller. Then the virtual controller is configured for tapping traffic exchanged between a first and a second virtual application instance, the first virtual application instance being executed on the first host and the second virtual application instance being executed on a second host. The monitoring controller is configured for determining a change in communication between the first and the

second virtual application instance. The monitoring controller is further configured for initiating a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[000112] The monitoring controller has the same possible advantages as the method performed by the monitoring controller. One possible advantage is that the monitoring function may be distributed within local hosts, wherein no centralised monitoring function may be needed. Another possible advantage is that the solution provides distributed zero-touch and dynamic instantiation and configuration of active monitoring functions when required based on passively observed application instance communication. Still a possible advantage is that the monitoring controllers run in a distributed fashion, wherein passive communication observation and possible associated data collection may be performed locally, so the monitoring controller may obtain timely and accurate information about which local application instances are communicating with which remote application instances. This provides another possible advantage that the monitoring function may be deployed without introducing an interface towards a service orchestrator. Yet another possible advantage is that IP address changes could be detected locally (for example, by inspecting the Internet Protocol packet, IP, open and close connection packets, or by rapidly aging out an IP address that saw no traffic addressed to it in a short time interval) and therefore a local traffic matrix better reflects the actual situation in the first host or the network connecting the first and the second host. This also reduces the need for integration of the orchestrator or network management system with third-party IP address management tools, thus providing both OPEX and potentially CAPEX savings. Still a possible advantage is that the monitoring of exchanging test packets may be initiated and stopped dynamically and when needed. It is also possible to check if there is already active monitoring ongoing between two physical hosts. If a new pair of applications needs to be monitored between these hosts a new monitoring session is not needed and instead the measurements from the ongoing session may be used. This reduces the amount of test packets that are sent into the network for active measurements. This also

means that the method may scale well and may also be used in cloud environments consisting of multiple data centres.

[000113] According to an embodiment, the monitoring controller is further configured to determine that there is a change in the communication between two application instances by obtaining a notification that there is a change in the communication between two application instances.

[000114] According to yet an embodiment, the monitoring controller is further configured to receive traffic between the two application instances, wherein the traffic has been tapped from a virtual switch comprised in the first host.

[000115] According to an alternative embodiment, the monitoring controller is further configured to instruct the virtual switch to tap traffic exchanged between the two application instances.

[000116] According to still an embodiment, the change comprises any of a new communication between the two application instances; that an existing, previously active communication having expired; a change in traffic intensity; a change in traffic pattern; and a change of communication protocol.

[000117] According to another embodiment, the monitoring controller is further configured for determining that no previous monitoring is currently ongoing between the hosts with regard to the detected change, at least not with regard to a metric associated with the detected change in the communication between the first and the second application instance.

[000118] According to a further embodiment, the monitoring controller is further configured for initiating the monitoring instance in the first host for transmitting test packets from the first host to the second host by obtaining a software image required for starting one or more monitoring instances in the first host.

[000119] According to an embodiment, the monitoring controller is further configured for requesting a monitoring controller of the second host to initiate a corresponding monitoring instance.

[000120] According to yet an embodiment, the monitoring controller is further configured for executing the monitoring between the two hosts.

[000121] According to still an embodiment, the monitoring controller is further configured for executing the monitoring by starting the monitoring instance.

[000122] According to another embodiment, the monitoring controller is further configured for providing monitoring information to a manager.

[000123] According to a further embodiment, the monitoring controller is further configured for stopping the monitoring between the two hosts and for providing a result of the monitoring to the manager.

[000124] According to an embodiment, the application traffic is exchanged between virtualised applications, or wherein the application instances are virtualised applications.

[000125] According to yet an embodiment, the monitoring controller, the two application instances and the monitoring instance(s) correspond to individual containers, individual virtual machines or uni-kernels.

[000126] According to another embodiment, the monitoring controller is further configured for requesting resources to be reserved on the first host for a monitoring instance from a manager.

[000127] **Figure 5** schematically shows an embodiment of an arrangement 500 in a monitoring controller 400. Comprised in the arrangement 500 in the monitoring controller 400 are here a processing unit 506, e.g. with a Digital Signal Processor, DSP. The processing unit 506 may be a single unit or a plurality of units to perform different actions of procedures described herein. The arrangement 500 of the monitoring controller 400 may also comprise an input unit 502 for receiving signals from other entities, and an output unit 504 for providing signal(s) to other entities. The input unit and the output unit may be arranged as an integrated entity or as illustrated in the example of figure 4, as one or more interfaces 401.

[000128] Furthermore, the arrangement 500 in the monitoring controller 400 comprises at least one computer program product 508 in the form of a non-volatile memory, e.g. an Electrically Erasable Programmable Read-Only Memory, EEPROM, a flash memory and a hard drive. The computer program product 508 comprises a computer program 510, which comprises code means, which when executed in the processing unit 506 in the arrangement 500 in the monitoring controller 400 causes the monitoring controller to perform the actions e.g. of the procedure described earlier in conjunction with **figures 1a-1d**.

[000129] The computer program 510 may be configured as a computer program code structured in computer program modules 510a-510e. Hence, in an exemplifying embodiment, the code means in the computer program of the arrangement 500 in the monitoring controller 400 comprises a determining unit, or module, for determining a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host. The computer program further comprises an initiating unit, or module, for initiating a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

[000130] The computer program modules could essentially perform the actions of the flow illustrated in **figures 1a-1d**, to emulate the monitoring controller 400. In other words, when the different computer program modules are executed in the processing unit 506, they may correspond to the units 403-404 of **figure 4**.

[000131] Although the code means in the embodiments disclosed above in conjunction with **figure 4** is implemented as computer program modules which when executed in the respective processing unit causes the monitoring controller to perform the actions described above in the conjunction with figures mentioned above, at least one of the code means may in alternative embodiments be implemented at least partly as hardware circuits.

[000132] The processor may be a single Central Processing Unit, CPU, but could also comprise two or more processing units. For example, the processor may include general purpose microprocessors; instruction set processors and/or related chips sets and/or special purpose microprocessors such as Application Specific Integrated Circuits, ASICs. The processor may also comprise board memory for caching purposes. The computer program may be carried by a computer program product connected to the processor. The computer program product may comprise a computer readable medium on which the computer program is stored. For example, the computer program product may be a flash memory, a Random-Access Memory RAM, Read-Only Memory, ROM, or an EEPROM, and the computer program modules described above could in alternative embodiments be distributed on different computer program products in the form of memories within the first host.

[000133] It is to be understood that the choice of interacting units, as well as the naming of the units within this disclosure are only for exemplifying purpose, and nodes suitable to execute any of the methods described above may be configured in a plurality of alternative ways in order to be able to execute the suggested procedure actions.

[000134] It should also be noted that the units described in this disclosure are to be regarded as logical entities and not with necessity as separate physical entities.

[000135] While the embodiments have been described in terms of several embodiments, it is contemplated that alternatives, modifications, permutations and equivalents thereof will become apparent upon reading of the specifications and study of the drawings. It is therefore intended that the following appended claims include such alternatives, modifications, permutations and equivalents as fall within the scope of the embodiments and defined by the pending claims.

- - -

CLAIMS

1. A method (100) for monitoring network performance, the method comprising

- determining (110), by a monitoring controller of a first host, a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host, and
- initiating (130), by the monitoring controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

2. The method (100) according to claim 1, wherein the determining (110) that there is a change in the communication between two application instances comprises obtaining a notification that there is a change in the communication between two application instances.

3. The method (100) according to claim 1, further comprising receiving traffic exchanged between the first and a second virtual application instance, wherein the traffic is tapped (105) from a virtual switch comprised in the first host.

4. The method (100) according to any of claims 1-3, wherein the change comprises any of a new communication between the two application instances; that an existing, previously active communication having expired; a change in traffic intensity; a change in traffic pattern; and a change of communication protocol.

5. The method (100) according to any of claims 1-4, further comprising determining (120) that no previous monitoring is currently ongoing between the hosts with regard to the detected change, at least not with regard to a metric

associated with the detected change in the communication between the first and the second application instance.

6. The method (100) according to any of claims 1-5, wherein the initiating (130) of the monitoring instance in the first host for transmitting test packets from the first host to the second host comprises obtaining (131) a software image required for starting one or more monitoring instances in the first host.

7. The method (100) according to any of claims 1-6, further comprising requesting (132) a monitoring controller of the second host to initiate a corresponding monitoring instance.

8. The method (100) according to any of claims 1-7, further comprising executing (140) the monitoring between the two hosts.

9. The method (100) according to claim 8, wherein the executing (140) of the monitoring comprises starting the monitoring instance.

10. The method (100) according to claim 8 or 9, further comprising providing (145) monitoring information to a manager.

11. The method (100) according to any of claims 8-10, further comprising stopping (150) the monitoring between the two hosts and providing (160) a result of the monitoring to the manager.

12. The method (100) according to any of claims 1-11, wherein the application traffic is exchanged between virtualised applications, or wherein the application instances are virtualised applications.

13. The method (100) according to claim 12, wherein the monitoring controller, the two application instances and the monitoring instance(s) correspond to individual containers, individual virtual machines or uni-kernels.

14. The method (100) according to any of claims 1-13, further comprising requesting resources to be reserved on the first host for a monitoring instance from a manager.

15. A monitoring controller (300, 400) of a first host for monitoring network performance, the monitoring controller (300, 400) being configured for:

- determining a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host, and
- initiating a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

16. The monitoring controller (300, 400) according to claim 15, wherein the monitoring controller is further configured to determine that there is a change in the communication between two application instances by obtaining a notification that there is a change in the communication between two application instances.

17. The monitoring controller (300, 400) according to claim 15, wherein the monitoring controller is further configured to receive traffic between the two application instances, wherein the traffic has been tapped from a virtual switch comprised in the first host.

18. The monitoring controller (300, 400) according to any of claims 15-17, wherein the change comprises any of a new communication between the two application instances; that an existing, previously active communication having expired; a change in traffic intensity; a change in traffic pattern; and a change of communication protocol.

19. The monitoring controller (300, 400) according to any of claims 15-18, further being configured for determining that no previous monitoring is currently ongoing between the hosts with regard to the detected change, at least not with regard to a metric associated with the detected change in the communication between the first and the second application instance.

20. The monitoring controller (300, 400) according to any of claims 15-19, further being configured for initiating the monitoring instance in the first host for transmitting test packets from the first host to the second host by obtaining a software image required for starting one or more monitoring instances in the first host.

21. The monitoring controller (300, 400) according to any of claims 15-20, further being configured for requesting a monitoring controller of the second host to initiate a corresponding monitoring instance.

22. The monitoring controller (300, 400) according to any of claims 15-21, further being configured for executing the monitoring between the two hosts.

23. The monitoring controller (300, 400) according to claim 22, further being configured for executing the monitoring by starting the monitoring instance.

24. The monitoring controller (300, 400) according to claim 22 or 23, further being configured for providing monitoring information to a manager.

25. The monitoring controller (300, 400) according to any of claims 8-10, further being configured for stopping the monitoring between the two hosts and for providing a result of the monitoring to the manager.

26. The monitoring controller (300, 400) according to any of claims 15-25, wherein the application traffic is exchanged between virtualised applications, or wherein the application instances are virtualised applications.

27. The monitoring controller (300, 400) according to claim 26, wherein the monitoring controller, the two application instances and the monitoring instance(s) correspond to individual containers, individual virtual machines or uni-kernels.

28. The monitoring controller (300, 400) according to any of claims 15-27, further being configured for requesting resources to be reserved on the first host for a monitoring instance from a manager.

29. A Computer program (510), comprising computer readable code means, which when run in a processing unit (506) comprised in an arrangement (500) in a monitoring controller (400) according to claims 15-28 causes the monitoring controller (400) to perform the corresponding method according to any of claims 1-14.

30. A Computer program product (508) comprising the computer program (510) according to claim 29.

31. A first host system comprising processing circuits and memory, the processing circuits and memory configured to instantiate a monitoring controller configured to:
- determining (110) a change in a communication between the first and the second virtual application instance, and
- initiating (130) a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance.

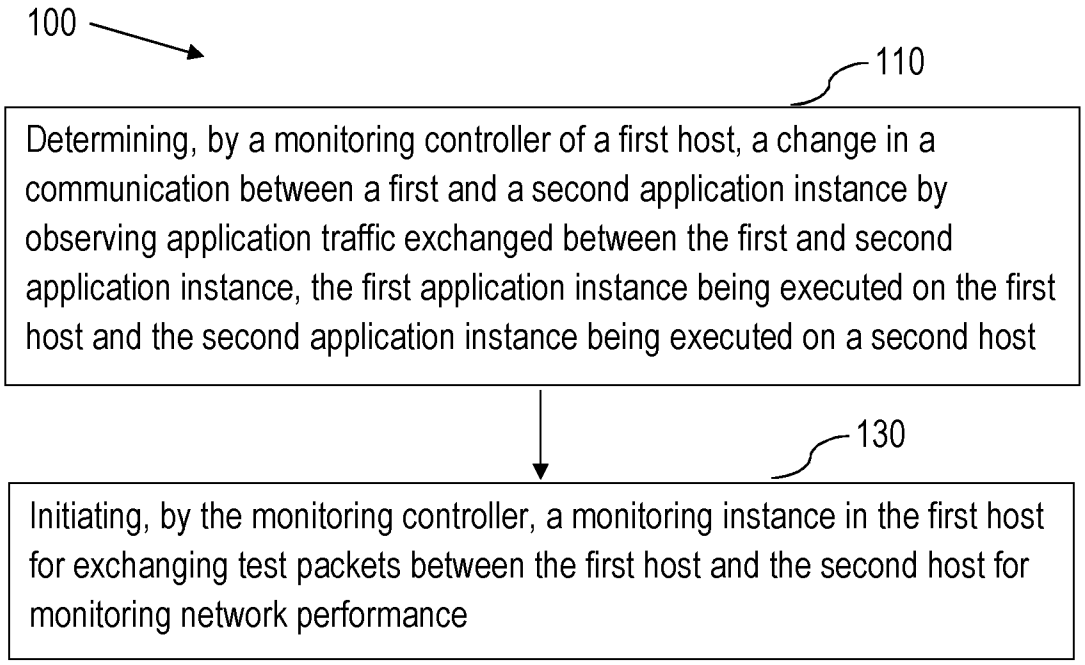
100
110
Determining, by a monitoring controller of a first host, a change in a communication between a first and a second application instance by observing application traffic exchanged between the first and second application instance, the first application instance being executed on the first host and the second application instance being executed on a second host
130
Initiating, by the monitoring controller, a monitoring instance in the first host for exchanging test packets between the first host and the second host for monitoring network performance

Fig. 1a

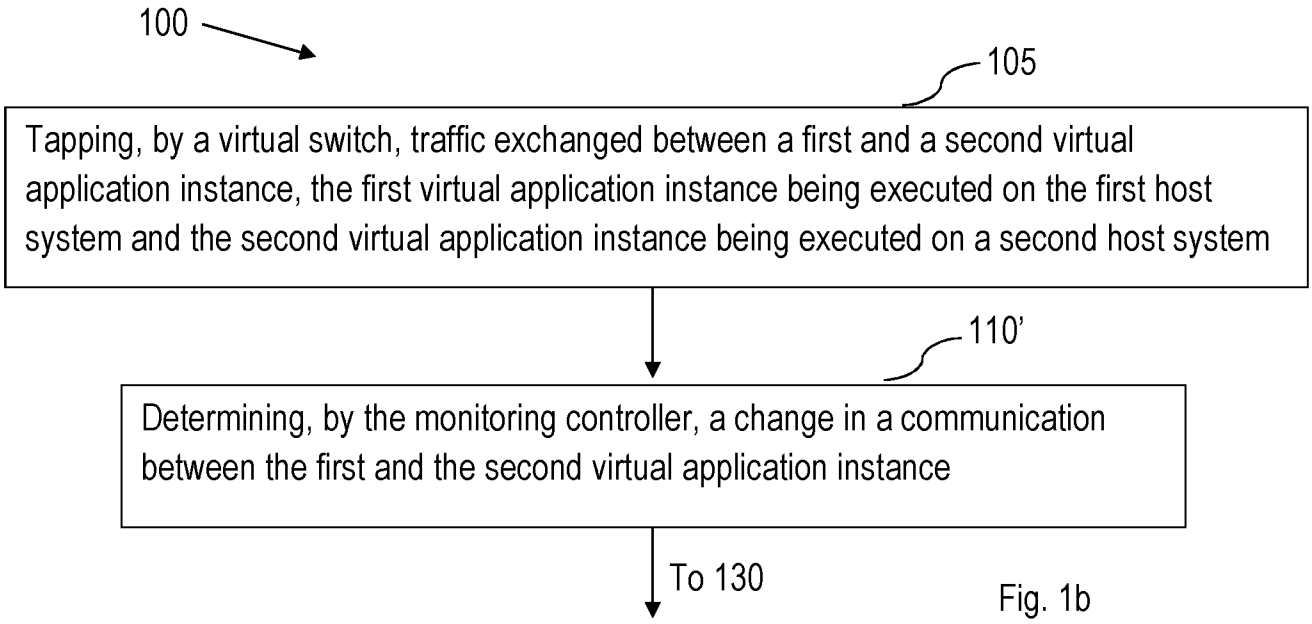
100
105
Tapping, by a virtual switch, traffic exchanged between a first and a second virtual application instance, the first virtual application instance being executed on the first host system and the second virtual application instance being executed on a second host system
110'
Determining, by the monitoring controller, a change in a communication between the first and the second virtual application instance
To 130

Fig. 1b

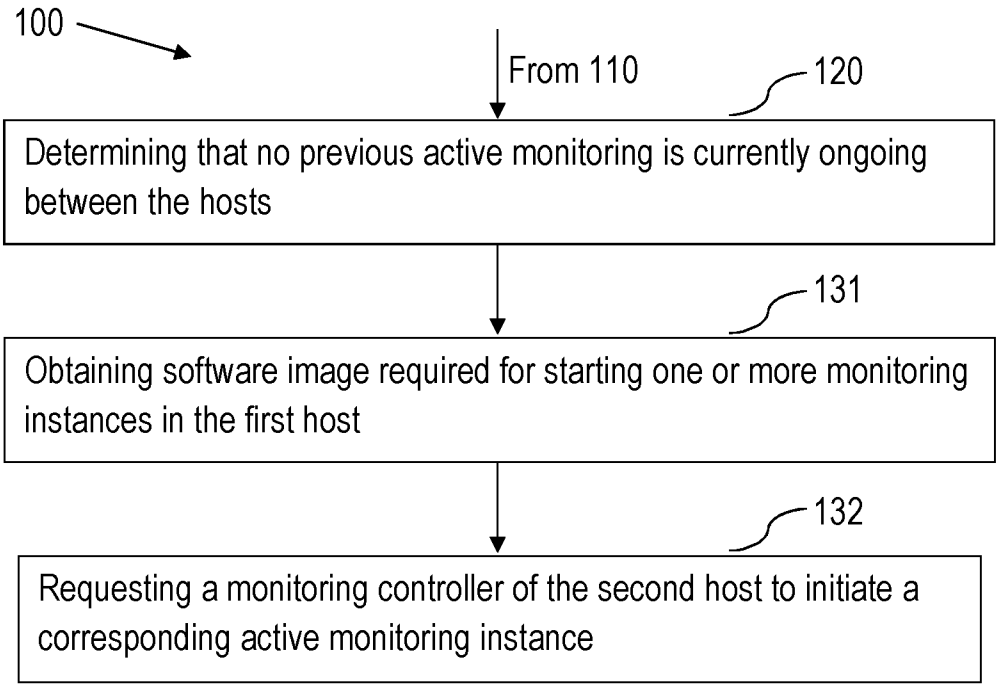
100
From 110
120
Determining that no previous active monitoring is currently ongoing between the hosts
131
Obtaining software image required for starting one or more monitoring instances in the first host
132
Requesting a monitoring controller of the second host to initiate a corresponding active monitoring instance

Fig. 1c

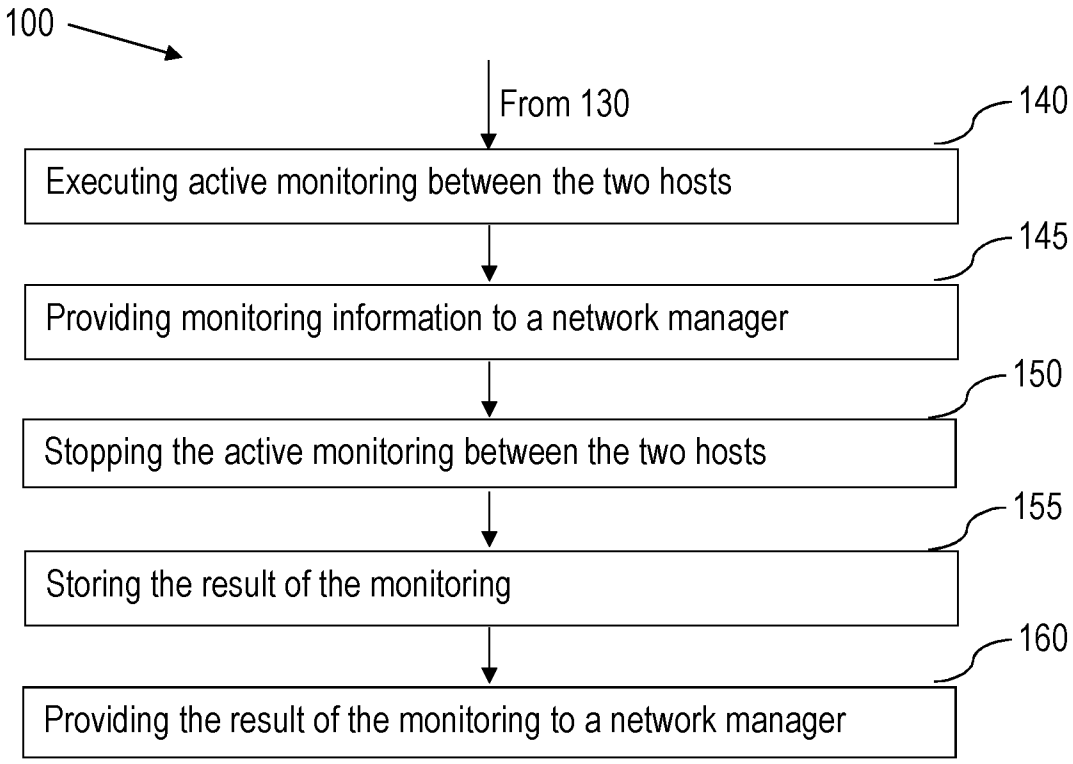
100
From 130
140
Executing active monitoring between the two hosts
145
Providing monitoring information to a network manager
150
Stopping the active monitoring between the two hosts
155
Storing the result of the monitoring
160
Providing the result of the monitoring to a network manager

Fig. 1d

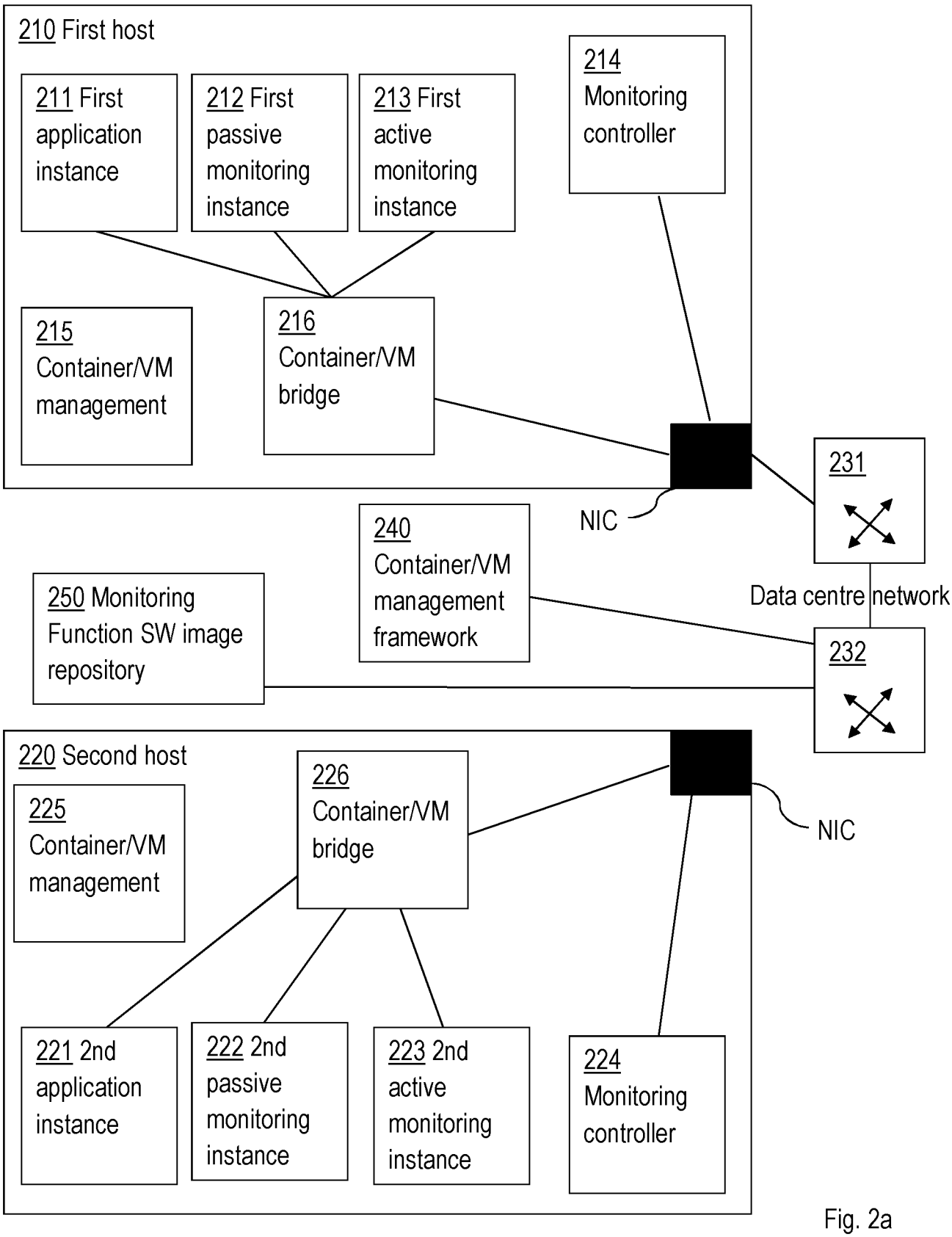
210 First host
211 First application instance
212 First passive monitoring instance
213 First active monitoring instance
214 Monitoring controller
215 Container/VM management
216 Container/VM bridge
NIC
231
240 Container/VM management framework
Data centre network
250 Monitoring Function SW image repository
232
220 Second host
225 Container/VM management
226 Container/VM bridge
NIC
221 2nd application instance
222 2nd passive monitoring instance
223 2nd active monitoring instance
224 Monitoring controller

Fig. 2a

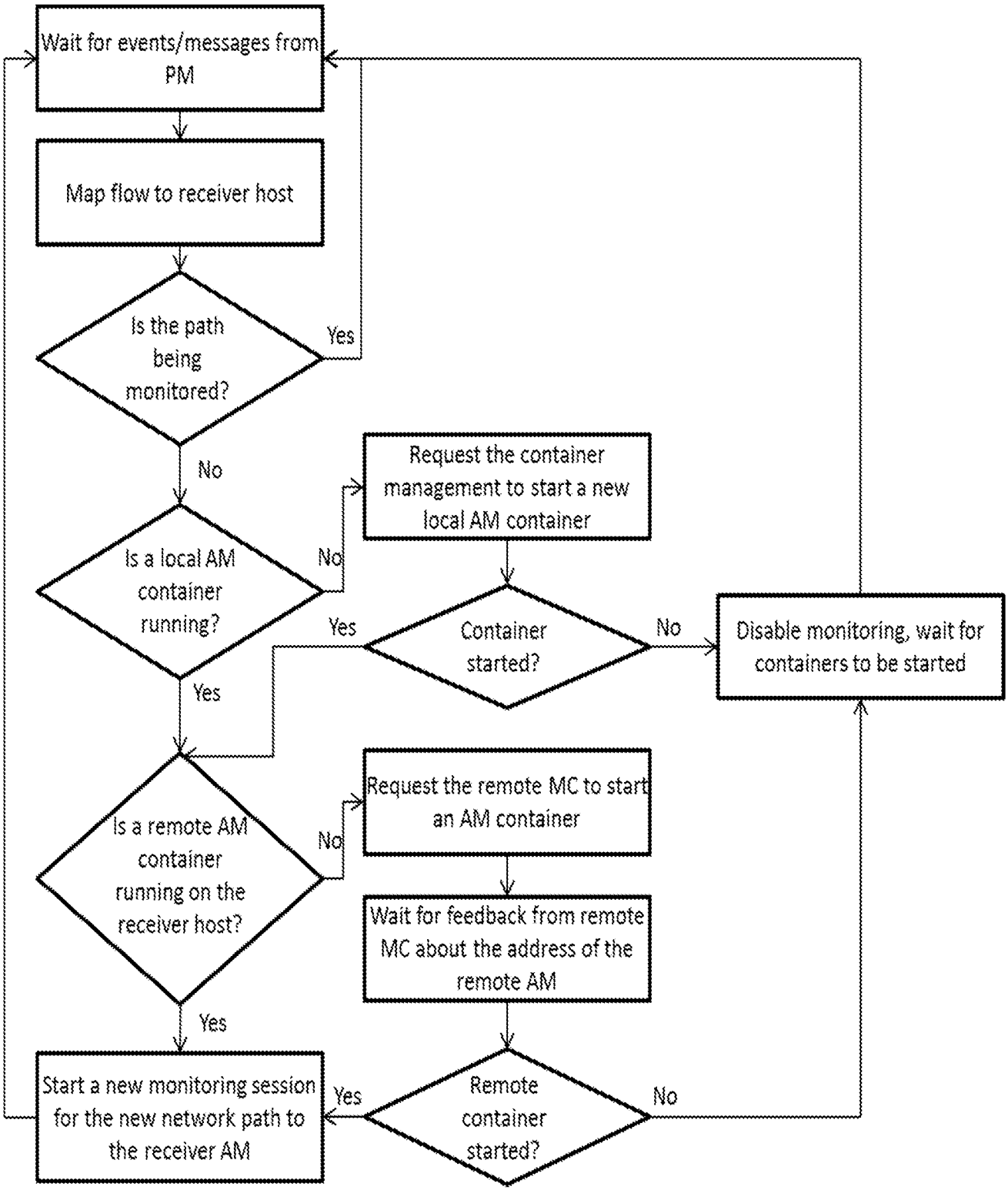
Wait for events/messages from PM
Map flow to receiver host
Is the path being monitored?
Yes
No
Is a local AM container running?
No
Request the container management to start a new local AM container
Container started?
Yes
No
Disable monitoring, wait for containers to be started
Yes
Is a remote AM container running on the receiver host?
No
Request the remote MC to start an AM container
Wait for feedback from remote MC about the address of the remote AM
Yes
Start a new monitoring session for the new network path to the receiver AM
Yes
Remote container started?
No

Fig. 2b

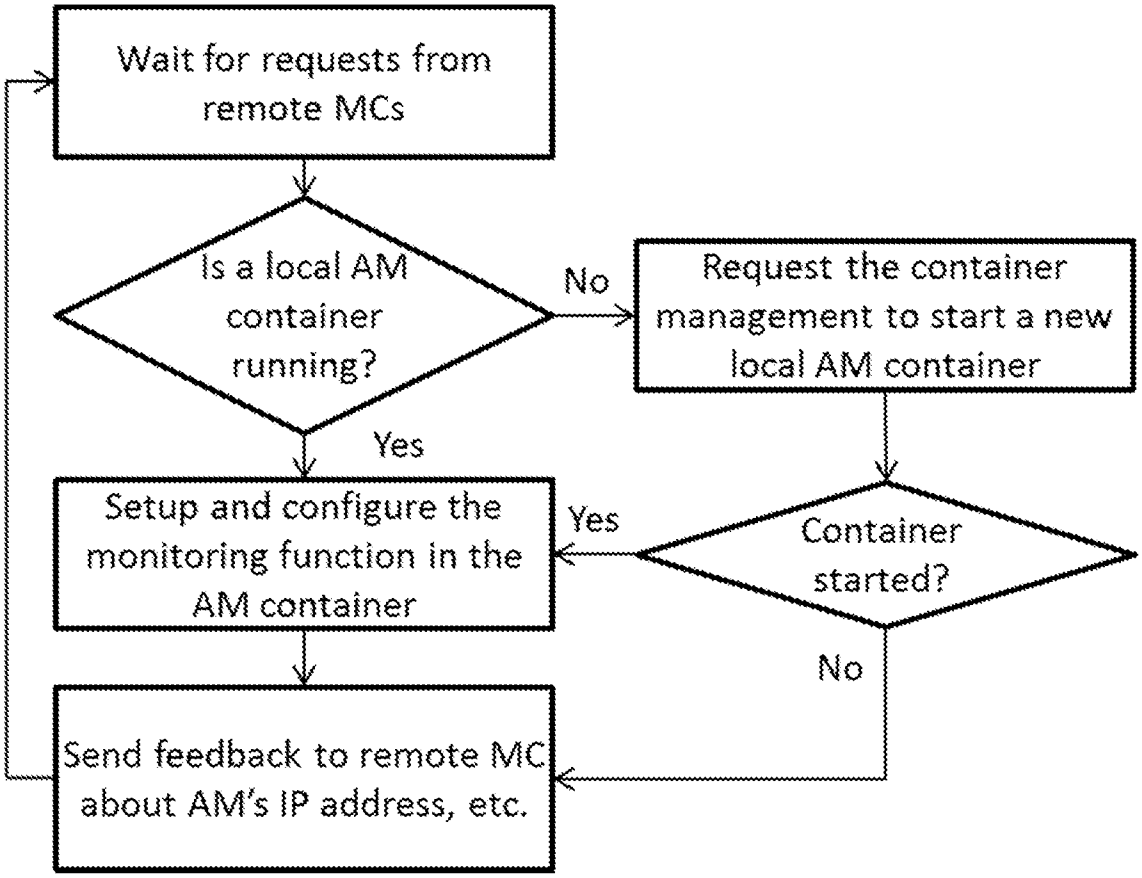
Wait for requests from remote MCs
Is a local AM container running?
No
Request the container management to start a new local AM container
Yes
Setup and configure the monitoring function in the AM container
Yes
Container started?
No
Send feedback to remote MC about AM's IP address, etc.

Fig. 2c

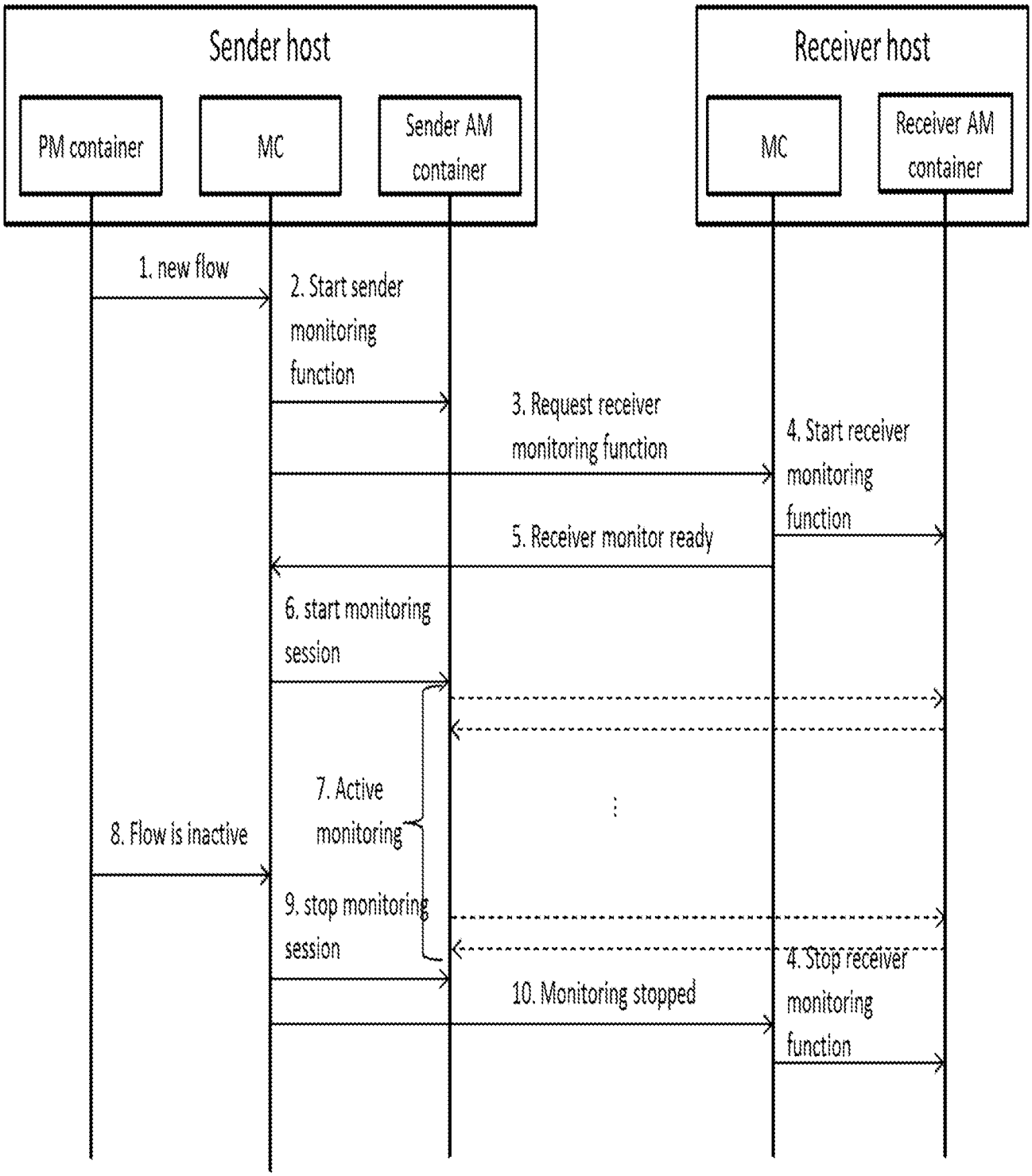
Sender host
PM container
MC
Sender AM container
Receiver host
MC
Receiver AM container
1. new flow
2. Start sender monitoring function
3. Request receiver monitoring function
4. Start receiver monitoring function
5. Receiver monitor ready
6. start monitoring session
7. Active monitoring
8. Flow is inactive
9. stop monitoring session
4. Stop receiver monitoring function
10. Monitoring stopped

Fig. 2d

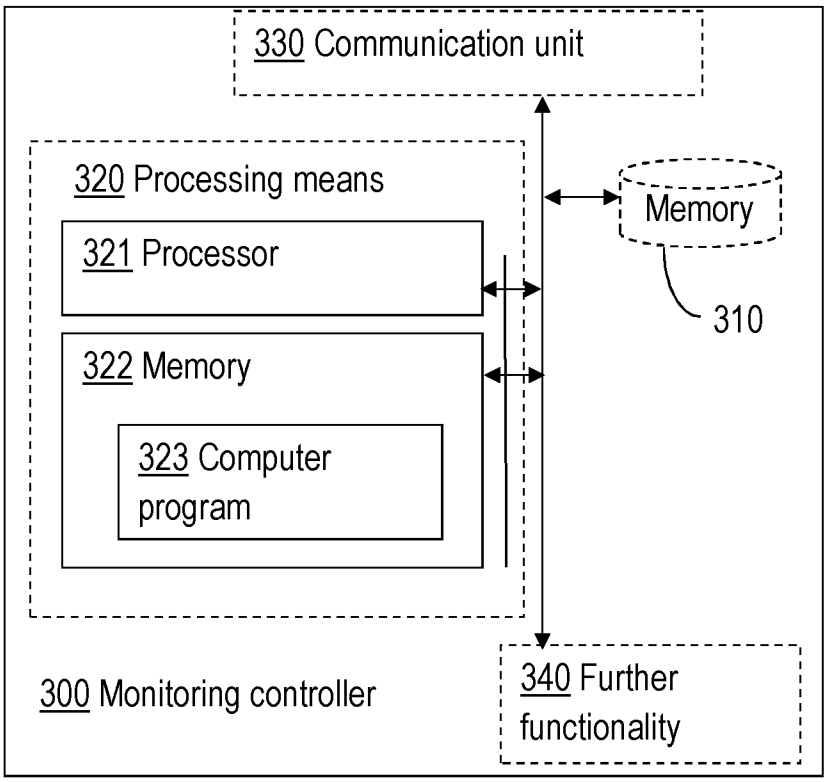
330 Communication unit
320 Processing means
321 Processor
322 Memory
323 Computer program
Memory
310
300 Monitoring controller
340 Further functionality

Fig. 3

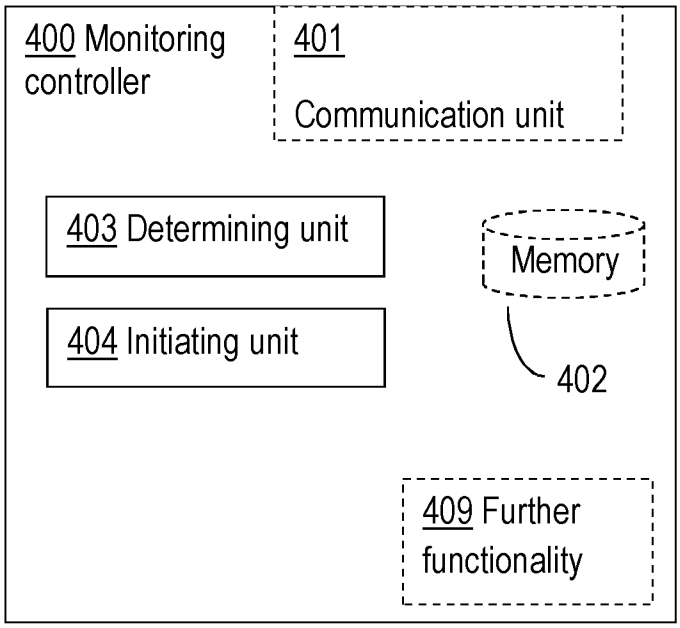
400 Monitoring controller
401
Communication unit
403 Determining unit
404 Initiating unit
Memory
402
409 Further functionality

Fig. 4

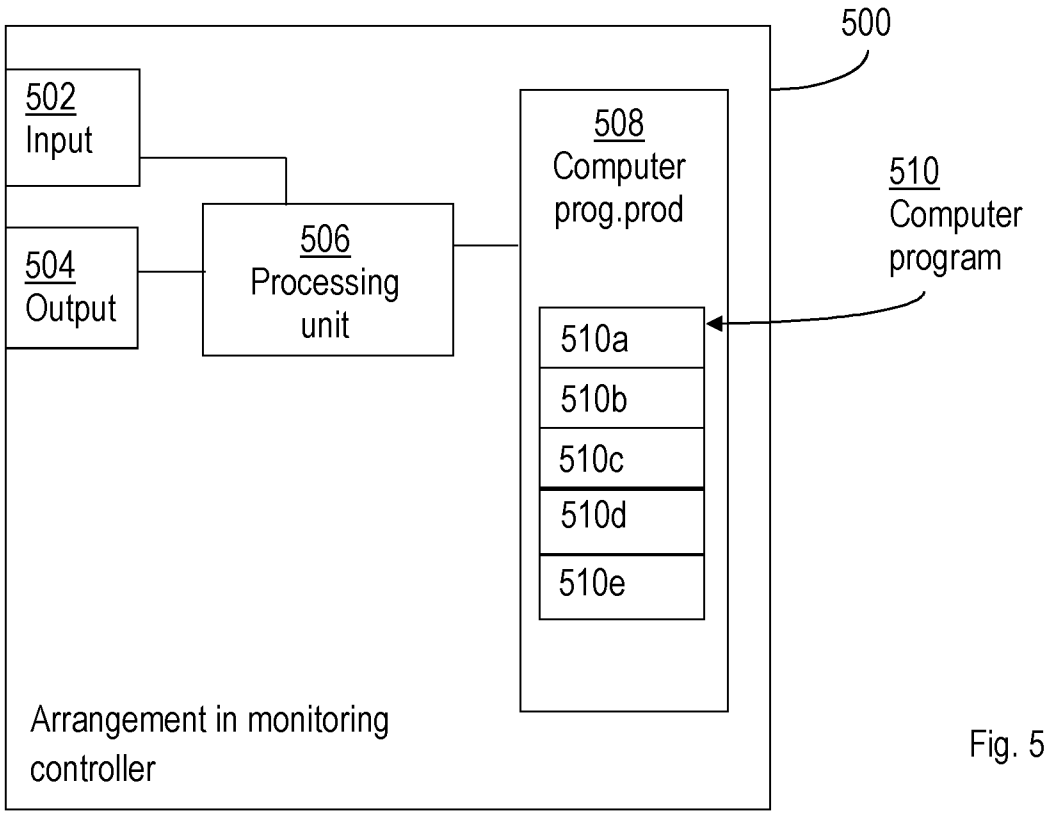
500
502
Input
504
Output
506
Processing
unit
508
Computer
prog.prod
510
Computer
program
510a
510b
510c
510d
510e
Arrangement in monitoring
controller

Fig. 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/SE2016/050430

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/26
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, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 6 363 056 B1 (BEIGI MANDIS SADR MOHAMMAD [US] ET AL) 26 March 2002 (2002-03-26) column 2, line 31 - line 58 column 5, line 50 - column 6, line 3 column 7, line 55 - column 8, line 14 figure 1 -----	1-31
X	US 6 763 380 B1 (MAYTON KIM IRVIN [US] ET AL) 13 July 2004 (2004-07-13) column 3, line 16 - line 39 column 4, line 10 - line 20 column 6, line 52 - column 7, line 15 column 12, line 7 - line 31 column 12, line 60 - column 14, line 48 ----- -/--	1-31

[X] Further documents are listed in the continuation of Box C.

[X] 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
14 September 2016	22/09/2016

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

1

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2002/120727 A1 (CURLEY ROBERT [US] ET AL) 29 August 2002 (2002-08-29) paragraphs [0058] - [0061], [0137] - [0142], [0237]	1-31
X	US 2016/099853 A1 (NEDELTCHEV PLAMEN NEDELTCHEV [US] ET AL) 7 April 2016 (2016-04-07) paragraphs [0013], [0017], [0033], [0060] - [0065]	1-31
A	HEDAYAT BRIX NETWORKS R KRZANOWSKI VERIZON A MORTON AT&T LABS K YUM JUNIPER NETWORKS J BABIARZ NORTEL NETWORKS K: "A Two-Way Active Measurement Protocol (TWAMP); rfc5357.txt", A TWO-WAY ACTIVE MEASUREMENT PROTOCOL (TWAMP); RFC5357.TXT, INTERNET ENGINEERING TASK FORCE, IETF; STANDARD, INTERNET SOCIETY (ISOC) 4, RUE DES FALAISES CH-1205 GENEVA, SWITZERLAND, 1 October 2008 (2008-10-01), XP015060332, the whole document	1-31
A	US 2002/131369 A1 (HASEGAWA TOORU [JP] ET AL) 19 September 2002 (2002-09-19) paragraphs [0002], [0021] - [0023], [0056] - [0061]	1-31
A	US 6 560 648 B1 (DUNN JAMES M [US] ET AL) 6 May 2003 (2003-05-06) the whole document	1-31

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/SE2016/050430

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 6363056 B1	26-03-2002	NONE	
US 6763380 B1	13-07-2004	NONE	
US 2002120727 A1	29-08-2002	AU 2001297721 A1	15-10-2002
		US 2002120727 A1	29-08-2002
		WO 02079903 A2	10-10-2002
US 2016099853 A1	07-04-2016	NONE	
US 2002131369 A1	19-09-2002	JP 2002281086 A	27-09-2002
		US 2002131369 A1	19-09-2002
US 6560648 B1	06-05-2003	NONE	