(19) DANMARK

Patent- og Varemærkestyrelsen

(10) **DK/EP 2869495 T3**

(12) # Oversættelse af europæisk patentskrift

(51) Int.Cl.: *H 04 L 12/24 (2006.01)* *H 04 L 29/12 (2006.01)*

(45) Oversættelsen bekendtgjort den: **2018-02-19**

(80) Dato for Den Europæiske Patentmyndigheds bekendtgørelse om meddelelse af patentet: **2018-01-10**

(86) Europæisk ansøgning nr.: **14191409.3**

(86) Europæisk indleveringsdag: **2014-11-03**

(87) Den europæiske ansøgnings publiceringsdag: **2015-05-06**

(30) Prioritet: **2013-11-05 US 201314072150**

(84) Designerede stater: **AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR**

(73) Patenthaver: **Solarwinds Worldwide, LLC, 7171 Southwest Parkway , Building 400, Austin, TX 78735, USA**

(72) Opfinder: **Zizlavsky, Marek, 29. augusta, 477/17, 03901 Turcianske Teplice, Slovakiet**
Pospisil, Tomas, Nemcice 106, 76843 Kostelec u Holesova, Tjekkiet
Mrkvicka, Tomas, Sportovni 251, 58281 Habry, Tjekkiet

(74) Fuldmægtig i Danmark: **Zacco Denmark A/S, Arne Jacobsens Allé 15, 2300 København S, Danmark**

(54) Benævnelse: **Knudeafduplikering i et netværksovervågningssystem**

(56) Fremdragne publikationer:
EP-A1- 2 051 446
US-A1- 2002 156 883
HEWLETT-PACKARD COMPANY ET AL: "Unique network node identification algorithm", RESEARCH DISCLOSURE, MASON PUBLICATIONS, HAMPSHIRE, GB, vol. 582, no. 62, 31 October 2012 (2012-10-31), page 838, XP007141640, ISSN: 0374-4353
MATTHIAS GEEL ET AL: "Mix-n-Match: Building Personal Libraries from Web Content", 23 September 2012 (2012-09-23), THEORY AND PRACTICE OF DIGITAL LIBRARIES, SPRINGER BERLIN HEIDELBERG, BERLIN, HEIDELBERG, PAGE(S) 345 - 356, XP047018044, ISBN: 978-3-642-33289-0 * chapter 3.5 , pages 350-351 *

DESCRIPTION

BACKGROUND:

Field:

[0001] Embodiments of the invention generally relate to network traffic monitoring, analysis, and/or reporting. More particularly, some embodiments are directed to methods, systems, and computer programs for node de-duplication of physical nodes monitored by a network monitoring system, for example.

Description of the Related Art:

[0002] Network management includes activities, methods, procedures, and tools related to the operation, administration, maintenance, and/or provisioning of networked systems. Functions that can be performed as part of network management may include, for example, planning, controlling, deploying, allocating, coordinating, and monitoring the resources of a network. Further functions may be related to network planning, frequency allocation, load balancing, configuration management, fault management, security management, performance management, bandwidth management, route analytics, and accounting management.

[0003] As mentioned above, a subset of network management includes network monitoring of network traffic. Network traffic data is of interest to network administrators for a number of reasons, including analyzing the impact of a new application on the network, troubleshooting network pain points, detecting slow or failing network devices, detecting heavy users of bandwidth, and securing networks. Various protocols for network traffic flow data have been developed. These protocols can contain numerous types of information, such as source internet protocol (IP) address, destination IP address, source port, destination port, IP protocol, ingress interface, IP Type of Service, start and finish times, number of bytes, and next hop.

[0004] As networks become larger and more complex, systems that monitor, analyze, and report on traffic flow data must become more efficient at handling the increasing number of network devices and amount of information generated about

[0005] The article entitled "Unique network node identification algorithm" by Hewlett-Packard Company et al., Research disclosure, vol. 582. no. 62, 1 October 2012, pages 838 to 840, ISSN 0374-4353, describes a unique network node identification algorithm, in which network devices are uniquely identified using a combination of MAC addresses on the device, IP addresses assigned to the device, the SNMP system object ID of the device, the DNS name of the device, and the tenant the device is assigned to.

[0006] The article entitled "Mix-n-Match: Building Personal Libraries from Web Content" by Matthias Geel et al., Theory and Practice of Digital Libraries, 23 September 2012, pages 345 to 365, ISBN 978-3-642-33289-0, describes an approach to web content aggregation that allows information to be harvested from web pages, independent of specific markup languages. It is described that a content aggregation engine is realized as an extensive framework in a way that end users as well as developers can use the associated tools to crate personal libraries of content extracted from the web.

SUMMARY:

[0007] According to the present invention, there are disclosed methods, apparatuses, and computer program products for node de-duplication, as defined in the appended claims.

[0008] Certain embodiments are directed to methods, apparatuses, and computer program products for node de-duplication, which comprise, are configured to execute, and are configured to control a processor to execute a process comprising: discovering, by a network monitoring apparatus, nodes in a network, collecting a list of internet protocol (IP) addresses, media access control (MAC) addresses, domain name system (DNS) names, and sysnames for each of the nodes discovered in the network, comparing the IP addresses of each of the discovered nodes with IP addresses of current (monitored) nodes and other (previously) discovered nodes, comparing the MAC addresses of each of the discovered nodes with MAC addresses of the current (monitored) nodes and the other (previously) discovered nodes, comparing the DNS names of each of the discovered nodes with DNS names of the current (monitored) nodes and the other (previously) discovered nodes, comparing the sysnames of each of the discovered nodes with sysnames of the current (monitored) nodes and the other (previously) discovered nodes, and determining duplicate nodes that are duplicates of the other (previously) discovered nodes and/or the current (monitored) nodes based on the comparison of the IP addresses, MAC addresses, DNS names, and sysnames, wherein, for the determining, each of the discovered nodes is assigned a node ID, and a MatchIndex is assigned to each node ID, the MatchIndex indicating a likelihood of a match between the discovered node and any of the current (monitored) nodes and the other (previously) discovered nodes.

BRIEF DESCRIPTION OF THE DRAWINGS:

[0009] For proper understanding of the invention, reference should be made to the accompanying drawings, wherein:

Fig. 1 illustrates a block diagram of a system according to an embodiment;

Fig. 2 illustrates a block diagram of a system according to one embodiment;

Fig. 3 illustrates a flow diagram of a method according to one embodiment;

Fig. 4 illustrates a flow diagram of a method according to another embodiment;

Fig. 5 illustrates a block diagram of an apparatus according to an embodiment; and

Fig. 6 illustrates a flow diagram of a method according to another embodiment.

DETAILED DESCRIPTION:

[0010] It will be readily understood that the components of the invention, as generally described and illustrated in the figures herein, may be arranged and designed in a wide variety of different configurations. Thus, the following detailed description of the embodiments of systems, methods, apparatuses, and computer program products for node de-duplication, as represented in the attached figures, is not intended to limit the scope of the invention, but is merely representative of selected embodiments of the invention.

[0011] The features, structures, or characteristics of the invention described throughout this specification may be combined in any suitable manner in one or more embodiments. For example, the usage of the phrases "certain embodiments," "some embodiments," or other similar language, throughout this specification refers to the fact that a particular feature, structure, or characteristic described in connection with the embodiment may be included in at least one embodiment of the present invention. Thus, appearances of the phrases "in certain embodiments," "in some embodiments," "in other embodiments," or other similar language, throughout this specification do not necessarily all refer to the same group of embodiments, and the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments. Additionally, if desired, the different functions discussed below may be performed in a different order and/or concurrently with each other. Furthermore, if desired, one or more of the described functions may be optional or may be combined. As such, the following description should be considered as merely illustrative of the principles, teachings and embodiments of this invention, and not in limitation thereof.

[0012] It should be noted that throughout this specification the terms network devices and network nodes, or simply devices or nodes, may be used interchangeably to refer to any physical device that is capable of connecting to and/or communicating on a network. Examples of such devices or nodes may include, but is not limited to, routers, switches, servers, computers, laptops, tablets, telephones, printers, mobile devices, and any other current or future component capable of sending, receiving, or forwarding information over a communications channel.

[0013] Fig. 1 illustrates an example of a system according to one embodiment. The system includes network monitoring system 100, network monitoring system storage 110, switch device 130, and one or more network devices 120. Network monitoring system storage 110 can store network monitoring data. Network monitoring system storage 110 can be a database or any other appropriate storage device. Network devices 120 may be nodes in the network that are monitored by network traffic monitor 100. It should be noted that any number and type of network devices 120 can be supported in the system. Accordingly, embodiments are not limited to the number and type of network devices illustrated in Fig. 1. In an embodiment, network monitoring system storage 110 may store discovery result database tables that store and organize information about discovered network devices in the network.

[0014] In some cases, network devices can be accessible on multiple internet protocol (IP) addresses. For example, some devices can simultaneously have more than one IP address. Also, some nodes are dynamic in that they have IP addresses that vary in time. Such dynamic IP addresses cause problems for de-duplication logic because it might be matching an outdated primary IP address from discovery result with the current IP address of a dynamic node.

[0015] Currently, systems generally apply logic where one IP address equals one network node. As a result, nodes with multiple IP address bindings may not be recognized as a single node. This behavior can result in a state where one physical device is being monitored more than once, which can obviously cause additional overhead and inefficiencies in the network monitoring system.

[0016] Therefore, users usually want to have devices which respond on multiple IP addresses monitored as a single node in the network. Embodiments of the invention implement an automatic network discovery that is able to detect such situation and avoid processing the same physical node multiple times (e.g., each time using a different IP address). One embodiment includes a de-duplication logic configured to automatically identify duplicate nodes such that a single node is not monitored multiple times.

[0017] Certain embodiments identify one or more pieces of information that function as a node identifier needed to uniquely identify the network node. Having this unique node identifier, certain embodiments can proceed with defining logic, which detects duplicates within nodes found during the discovery process (e.g., same physical network node under different IP addresses) and/or within nodes already being monitored under a different IP address.

[0018] According to certain embodiments, the information used to uniquely identify nodes includes information which can be polled easily, is available for a majority of devices (i.e., vendor independent), is available for Internet Control Message Protocol (ICMP) nodes as well as other types of nodes (e.g., ICMP nodes can be considered to be nodes not reachable over SNMP or WMI), and is a minimal set of information that can still provide accurate results while being efficient.

[0019] Certain embodiments can handle at least two typical uses cases. For example, one use case may include running an automated network discovery over a subnetwork, which contains devices accessible on multiple IP addresses. As is known, devices that belong to a subnet are addressed with a common, identical, most-significant bit-group in their IP address. Another use case may include running an automated network discovery over a subnetwork, which is already being monitored.

[0020] Certain embodiments are able to identify network nodes as network duplicates, even if one set of information was collected in a different time frame than the other. For instance, this may occur when a user decides to import discovery results for a scheduled discovery profile. In such a case, some embodiments may need to work with possibly outdated information collected during discovery and compare it to new information, which is being continuously collected for all monitored nodes. Accordingly, de-duplication according to certain embodiments can take place during a discovery job where discovered nodes are compared to each other such that duplicates are removed, and/or can take place during a discovery result importing where discovered nodes are imported and compared against existing nodes (e.g., nodes stored in network monitoring system storage or database) such that duplicate nodes are identified and removed.

[0021] According to an embodiment, a data collection set made up of DNS name, Sysname, IP address, and MAC address for each network node is used to help identify duplicate nodes. One embodiment includes logic which compares pieces of information from the data collection set against each other, for example DNS to DNS, MACs to MACs, etc. The logic may be implemented in a Duplicate Detector component. A result from the detection is the match index, which indicates whether there is a match, no match, or unkown.. Certain embodiments also provide logic for aggregating partial results from detectors and calculating a final verdict as to whether a node is a duplicate or not. For example, in an embodiment, the node de-duplication may include several sub-iterations that are each responsible for discovering selected IP-range. At the end of every iteration, de-duplication is performed to omit nodes (e.g., endpoints) as soon as they are thought to be duplicates. In case of an unknown or non-duplicate result for one iteration, the node is passed to the next step or iteration for processing.

[0022] In an embodiment, two sets of duplicate detectors may be provided. One set of detectors can be used during the automated discovery process to filter out newly found devices and remove duplicates. Another set of detectors can be used during discovery result import to avoid adding duplicate nodes into set of monitored nodes.

[0023] In one embodiment, the system is configured to collect list of all DNS names, Sysnames, IP addresses, and MAC addresses for all discovered nodes, and to store them, for example, as part of a discovery job result. This information may be stored in the network system monitoring storage or database 110. As mentioned above, this information can be used during at least two phases of discovery. For instance, the DNS names, Sysnames, IP addresses, and MAC addresses can be used when running discovery to check whether currently discovered node(s) are a duplicate of any other already found, and/or can be used during discovery result importing to compare discovery result(s) with existing nodes monitored by the system. In one embodiment, the MAC information is stored to a persistent storage, for example, as part of discovery result.

[0024] Fig. 2 illustrates a system 200, according to one embodiment, which may include an IP address duplicate detector 201, a DNS duplicate detector 203, a MAC address duplicate detector 202, and a Sysname duplicate detector 204. In an embodiment, a database 210 may be in communication with the IP address duplicate detector 201, DNS duplicate detector 203, MAC address duplicate detector 202, and Sysname duplicate detector 204 such that each of the detectors can collect the relevant addresses stored in the database 210. In an embodiment, the database 210 may be stored in the network traffic data storage 110 illustrated in Fig. 1. It should be noted that system 200 can be implemented completely in hardware, or in a combination of hardware and software.

[0025] Each of the duplicate detectors 201, 202, 203, 204 may have a defined priority of order of execution (e.g., lower number indicates earlier execution), a weight which indicates the reliability of the result provided by the duplicate detector (e.g., a weight of 0 will have no impact on final result), and a veto that is used as a top priority to determine if a node is a duplicate or not.

[0026] DNS duplicate detector 203 may be configured to compare a DNS of a discovered node with all other discovered nodes and current nodes being monitored. In an embodiment, DNS duplicate detector 203 may conclude that a node is a duplicate if the DNS of a discovered node is the same as a DNS used by any of the monitored nodes or any of the other discovered nodes.

[0027] MAC addresses are generally unique by design (although there are situations where the same MAC address is used on two different devices, for example cloned virtual machines hosted in two separated virtual hosts). MAC address duplicate detector 202 may be configured to compare a MAC address of a discovered node against all previously collected MAC addresses of nodes, which may be stored in a NodeMACAddresses database table in database 210. MAC address duplicate detector 202 may be configured to conclude that a node is a duplicate if a set of discovered MAC addresses is a sub-set of currently monitored MAC addresses for a node or if a set of monitored MAC addresses is sub-set of discovered MAC addresses.

[0028] Table 1 below illustrates an example where the nodes are considered equal according to the MAC address duplicate detector 202, where A,B,C,... represent MAC addresses. Meanwhile, Table 2 below illustrates an example where the nodes are not considered equal (e.g., based on MAC addresses).

TABLE 1

Node A	**Node B**	**Node C**	**Node D**
A	A	A	A
B	B	B	B
C	C		C

Node A	Node B	Node C	Node D
			D

TABLE 2

Node A	Node B	Node C	Node D
A	A	D	A
B	B	E	D
C	D	F	

[0029] Two nodes, which both have just two MAC addresses, for instance 0000000000000000 and 00000000000000E0 should be considered as equal. For instance, *Node A* and *Node B* are equal when and only when *Node A* 's list of MAC addresses is subset of *Node B'* s list of MAC addresses or *Node B* 's list of MACs is subset of *Node A 's* list of MAC.In that case, the system may look to a different duplicate detector or different de-duplication method (e.g., sysname matching) since, according to MAC addresses, the two nodes are equal.

[0030] Sysname duplicate detector 204 may be configured to compare a sysname of a discovered node with all other discovered nodes and current nodes being monitored. In an embodiment, Sysname duplicate detector 204 may conclude that a node is a duplicate if the Sysname of a discovered node is the same as a Sysname used by any of the monitored nodes or any of the other discovered nodes. In an embodiment, the Sysname may be the Sysname for Simple Network Management Protocol (SNMP) nodes or may be the full computer name for window management instrumentation (WMI) nodes, for example. It should be noted that embodiments do not limit data sources to SNMP and/or WMI, and other types of data sources are equally applicable according to certain embodiments (e.g., CLI over SSH on routers/switches or telnet, etc.).

[0031] As mentioned above, each duplicate detector may load an associated weight from a settings database table. The weight represents the reliability of the result provided by the associated duplicate detector. It is possible to set the weight to -1 to disable the associated duplicate detector. According to an embodiment, weight values may range from 0 to 100 where 0 represents the least reliable and 100 represents the most reliable.

[0032] All of the duplicate detectors (d_1, ..., d_n) may be executed sequentially with order defined by priority, as discussed above. Each of the duplicate detectors may set an 'IsAuthoritative' flag to be true, which may then terminate execution of following duplicate detectors. In such a case, the vote of the duplicate detector with the 'IsAuthoritative' flag set to true is considered as final, ignoring all other votes. According to one embodiment, if there is no 'IsAuthoritative' flag set to true, a final result as to whether a node is a duplicate is computed as a sum of all duplicate detector vote result values as follows:

Final Decision =

d IsDuplicate()*d Priority+ +d IsDuplicate()*d Priority

d_1.IsDuplicate()·d_1.Priority+...+d_n.IsDuplicate()·d_n.Priority,

where d_1 is a first duplicate detector, d_2 is a second duplicate detector, ...and d_n is an nth duplicate detector. Accordingly, d_n.IsDuplicate() is a function representing the nth duplicate detector's conclusion as to whether a node is a duplicate.

[0033] As illustrated in Table 3 below, Each duplicate detector may return a list of node IDs for all duplicate nodes it finds. Each node ID may have an associated MatchIndex assigned, which indicates the likelihood of the match. In an embodiment, the range of MatchIndex values is 0 to 100, where 0 indicates the least likelihood of a match and 100 indicates the greatest likelihood of a match. According to an embodiment, system 200 may be configured to group the duplicate node information depicted in Table 3 by node ID, and to sum the MatchIndexes for the same node ID. Then, system 200 may be configured to select the node ID with the highest summed total MatchIndex for discarding.

TABLE 3

Duplicate Detector	**DuplicateNodeID**	**MatchIndex**
Dns	5	90
Mac	1	60
Mac	5	80
Sysname	5	85
Final Decision:	**5**	**90**

[0034] Table 4 below illustrates an example results table, according to an embodiment. In this example, each row of the table may represent a node. The 'DnsDuplicateDetector' column shows the conclusion of the DNS duplicate detector as to whether the node is a duplicate or not. Similarly, the 'MacAddressDuplicateDetector' column shows the conclusion of the MAC address duplicate detector as to whether the node is a duplicate or not, and the 'NameDuplicateDetector' column shows the conclusion of the Sysname duplicate detector as to whether the node is a duplicate or not. Then, the final 'Expected Result' column shows the expected result for the node.

DnsDuplicateDetector	**MacAddressDuplicateDetector**	**NameDuplicateDetector**	**Expected Result**
Is Duplicate	Is Duplicate	Is Duplicate	**Is Duplicate**
Is Duplicate	Is Duplicate	Don't know	**Is Duplicate**
Is Duplicate	Is Duplicate	Is Not Duplicate	**Is Duplicate**
Is Duplicate	Don't know	Is Duplicate	**Is Duplicate**
Is Duplicate	Don't know	Don't know	**Is Duplicate**
Is Duplicate	Don't know	Is Not Duplicate	**Is Duplicate**
Is Duplicate	Is Not Duplicate	Is Duplicate	**Is Duplicate**
Is Duplicate	Is Not Duplicate	Don't know	**Is Duplicate**
Is Duplicate	Is Not Duplicate	Is Not Duplicate	**Is Not Duplicate**

Don't know	Is Duplicate	Is Duplicate	**Is Duplicate**
Don't know	Is Duplicate	Don't know	**Is Duplicate**
Don't know	Is Duplicate	Is Not Duplicate	**Is Duplicate**
Don't know	Don't know	Is Duplicate	**Is Duplicate**
Don't know	Don't know	Don't know	**Is Not Duplicate**
Don't know	Don't know	Is Not Duplicate	**Is Not Duplicate**
Don't know	Is Not Duplicate	Is Duplicate	**Is Not Duplicate**
Don't know	Is Not Duplicate	Don't know	**Is Not Duplicate**
Don't know	Is Not Duplicate	Is Not Duplicate	**Is Not Duplicate**
Is Not Duplicate	Is Duplicate	Is Duplicate	**Is Duplicate**
Is Not Duplicate	Is Duplicate	Don't know	**Is Not Duplicate**
Is Not Duplicate	Is Duplicate	Is Not Duplicate	**Is Not Duplicate**
Is Not Duplicate	Don't know	Is Duplicate	**Is Not Duplicate**
Is Not Duplicate	Don't know	Don't know	**Is Not Duplicate**
Is Not Duplicate	Don't know	Is Not Duplicate	**Is Not Duplicate**
Is Not Duplicate	Is Not Duplicate	Is Duplicate	**Is Not Duplicate**
Is Not Duplicate	Is Not Duplicate	Don't know	**Is Not Duplicate**
Is Not Duplicate	Is Not Duplicate	Is Not Duplicate	**Is Not Duplicate**

TABLE 4

[0035] Fig. 3 illustrates a flow diagram of a method for node de-duplication according to one embodiment. In the example of Fig. 3, at 300, one or more nodes are discovered. At 310, the associated IP address of the node(s) is discovered. At 320, the supported technology used for obtaining device information, such as SNMP, WMI, etc. is detected. At 330, information about the node(s) is obtained via the (detected) supported technology. At 340, de-duplication of the discovered node(s) is performed. It is determined whether the node(s) is a duplicate at 350. If it is, the duplicate node(s) is omitted from the discovered result set at 360. If the node(s) is not a duplicate, at 370, discovered node(s) are saved to the database. At 380, data associated with discovered node(s) are prepared for discovery import..

[0036] Fig. 4 illustrates a flow diagram of a method for node de-duplication according to another embodiment. In the example of Fig. 4, at 400, a result of a discovery job is saved. At 405, the discovery result is loaded and deserialized from the database. At 410, detection of duplicate is primarily done with primary IP address associated with the node. At 415, it is detected whether node with same IP is already monitored by any engine. If node with same IP is already monitored, the existing node information is updated at 420. If node with same IP is not already monitored, duplicate detection against ignored nodes is performed at 425. At 430, it is determined whether node is a duplicate. If so, the conditions are logged and the duplicate node is discarded at 435. If it is not determined as a duplicate, a duplicate check is performed against all monitored nodes at 440. At 445, it is determined whether node is a duplicate. If it is determined as duplicate, the conditions are logged and the duplicate node is discarded at 450. If it is not determined as a duplicate, node(s) information is saved to persistent storage at 455.

At 460, result of discovery import is a list of nodes with associated IP addresses and MAC addresses.

[0037] Fig. 5 illustrates a block diagram of an apparatus 10 that may implement one embodiment of the invention. Apparatus 10 may include a bus 12 or other communications mechanism for communicating information between components of apparatus 10. Apparatus 10 also includes a processor 22, coupled to bus 12, for processing information and executing instructions or operations. Processor 22 may be any type of general or specific purpose processor. Apparatus 10 further includes a memory 14 for storing information and instructions to be executed by processor 22. Memory 14 can be comprised of any combination of random access memory ("RAM"), read only memory ("ROM"), static storage such as a magnetic or optical disk, or any other type of machine or computer readable media. Apparatus 10 further includes a communication device 20, such as a network interface card or other communications interface, to provide access to a network. As a result, a user may interface with apparatus 10 directly or remotely through a network or any other method.

[0038] Computer readable media may be any available media that can be accessed by processor 22 and includes both volatile and nonvolatile media, removable and non-removable media, and communication media. Communication media may include computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media.

[0039] Processor 22 is further coupled via bus 12 to a presentation device 24, such as a display, monitor, screen, or web browser , for displaying information to a user, such as network traffic information. A user input component 26 , such as a keyboard, computer mouse, or web browser, are further coupled to bus 12 to enable a user to interface with apparatus 10. Processor 22 and memory 14 may also be coupled via bus 12 to a database system 30 and, thus, may be able to access and retrieve information stored in database system 30. In one embodiment, database system 30 is the network monitoring system storage 110 illustrated in Fig. 1. Although only a single database is illustrated in Fig. 5, any number of databases may be used in accordance with certain embodiments.

[0040] In one embodiment, memory 14 stores software modules that provide functionality when executed by processor 22. The modules may include an operating system 15 that provides operating system functionality for apparatus 10. The memory may also store one or more duplicate detector(s) 16, which supports a node deduplication functionality, as discussed above. The one or more duplicate detector(s) 16 may include, for example, IP address duplicate detector 201, DNS duplicate detector 203, MAC address duplicate detector 202, and Sysname duplicate detector 204, as depicted in Fig. 2 discussed above. Apparatus 10 may also include one or more other functional modules 18 to provide additional functionality.

[0041] Database system 30 may include a database server and any type of database, such as a relational or flat file database. Database system 30 may store data related to network traffic flow of each of the entities in the network, and/or any data associated with apparatus 10 or its associated modules and components.

[0042] In certain embodiments, processor 22, duplicate detector(s) 16, and other functional modules 18 may be implemented as separate physical and logical units or may be implemented in a single physical and logical unit. Furthermore, in some embodiments, processor 22, duplicate detector(s) 16, and other functional modules 18 may be implemented in hardware, or as any suitable combination of hardware and software.

[0043] In some embodiments, processor 22 is configured to control apparatus 10 to discover nodes in a network. According to an embodiment, information identifying the discovered nodes may be stored in database 110, for example. Processor 22 may be configured to control apparatus 10 to collect a list of IP addresses, MAC addresses, DNS names, and sysnames for each of the nodes discovered in the network.

[0044] According to one embodiment, processor 22 may be configured to control apparatus 10 to execute an IP duplicate detector configured to compare the IP addresses of each of the discovered nodes with IP addresses of current nodes and other discovered nodes, a MAC duplicate detector configured to compare the MAC addresses of each of the discovered nodes with MAC addresses of the current nodes and the other discovered nodes, a DNS duplicate detector configured to compare the DNS names of each of the discovered nodes with DNS names of the current nodes and the other discovered nodes, and a name duplicate detector configured to compare the sysnames of each of the discovered nodes with sysnames of the current nodes and the other discovered nodes. Processor 22 may then be configured to control apparatus 10 to determine duplicate nodes that are duplicates of the other discovered nodes and/or the current nodes based on the result of comparison of the IP duplicate detector, the MAC duplicate detector, the DNS duplicate detector, and the name duplicate detector.

[0045] In an embodiment, processor 22 may be configured to control apparatus 10 to discard the duplicate nodes. According to one embodiment, processor 22 may be configured to control apparatus 10 to assign a priority to each of the IP duplicate detector, the MAC duplicate detector, the DNS duplicate detector, and the name duplicate detector that determines an order of execution. Apparatus 10 may be controlled to determine the duplicate nodes, for example, by executing the following formula:
$d_1.\mathrm{IsDuplicate}()*d_1.\mathrm{Priority}+\ldots+d_n.\mathrm{IsDuplicate}()*d_n.\mathrm{Priority}.$

[0046] In an embodiment, each of the discovered nodes may be assigned a node ID. Processor 22 may be configured to control apparatus 10 to assign a MatchIndex to each node ID, where the MatchIndex indicates a likelihood of a match between the discovered node and any of the current nodes and the other discovered nodes. According to one embodiment, processor 22 may be configured to control apparatus 10 to group the duplicate nodes by node ID and to sum the MatchIndexes for the same node ID. Additionally, a weight is assigned to each of the IP duplicate detector, the MAC duplicate detector, the DNS duplicate detector, and the name duplicate detector. The weight indicates the reliability of the result provided by the respective duplicate detectors.

[0047] Fig. 6 illustrates an example flow chart of a method, according to one embodiment. The method includes, at 600, discovering, for example by a network monitoring apparatus, nodes in a network. The method may then include, at 610, collecting a list of internet protocol (IP) addresses, media access control (MAC) addresses, domain name system (DNS) names, and sysnames for each of the nodes discovered in the network. At 620, the method includes comparing the IP addresses of each of the discovered nodes with IP addresses of current nodes and other discovered nodes, At 630, the method includes comparing the MAC addresses of each of the discovered nodes with MAC addresses of the current nodes and the other discovered nodes. At 640, the method includes comparing the DNS names of each of the discovered nodes with DNS names of the current nodes and the other discovered nodes. At 650, the method includes comparing the sysnames of each of the discovered nodes with sysnames of the current nodes and the other discovered nodes. The method may further include, at 660, determining duplicate nodes that are duplicates of the other discovered nodes and/or the current nodes based on the comparison of the IP addresses, MAC addresses, DNS names, and sysnames.

[0048] In some embodiments, the functionality of any of the methods described herein, such as those of Figs. 3, 4, and 6, may be implemented by software and/or computer program code stored in memory or other computer readable or tangible media, and executed by a processor. In other embodiments, the functionality may be performed by hardware, for example through the use of an application specific integrated circuit (ASIC), a programmable gate array (PGA), a field programmable gate array (FPGA), or any other combination of hardware and software.

[0049] One having ordinary skill in the art will readily understand that the invention as discussed above may be practiced with steps in a different order, and/or with hardware elements in configurations which are different than those which are disclosed. Therefore, although the invention has been described based upon these preferred embodiments, it would be apparent to those of skill in the art that certain modifications, variations, and alternative constructions would be apparent, without departing from the present invention as defined in the appended claims.

[0050] As evident from the above, there are disclosed systems, methods, apparatuses, and computer program products for node de-duplication. One method includes discovering, by a network monitoring apparatus, nodes in a network, and collecting a list of internet protocol (IP) addresses, media access control (MAC) addresses, domain name system (DNS) names, and sysnames for each of the nodes discovered in the network. The method may also include comparing the collected list of information for each of the discovered nodes with corresponding information for current nodes and other discovered nodes. The method may then includes determining duplicate nodes that are duplicates of the other discovered nodes and/or the current nodes based on the comparison of the IP addresses, MAC addresses, DNS names, and sysnames.

REFERENCES CITED IN THE DESCRIPTION

Non-patent literature cited in the description

- **HEWLETT-PACKARD COMPANY et al.**Unique network node identification algorithmResearch disclosure, 2012, vol. 582, 620374-4353838-840 [0005]
- **MATTHIAS GEEL et al.**Mix-n-Match: Building Personal Libraries from Web ContentTheory and Practice of Digital Libraries, 2012, 978-3-642-33289-0345-365 [0006]

Patentkrav

1. Fremgangsmåde omfattende:
at opdage (600), ved hjælp af en netværksovervågningsindretning, knuder i et netværk;
at indsamle (610) en liste af internetprotokol, IP, -adresser, MAC, media access control, -adresser, domænenavnsystem, DNS, -navne og sysnames for hver af knuderne, der opdages i netværket;
at sammenligne (620) IP-adresserne af hver af de opdagede knuder med IP-adresser af overvågede knuder og tidligere opdagede knuder;
at sammenligne (630) MAC-adresserne af hver af de opdagede knuder med MAC-adresser af de overvågede knuder og de tidligere opdagede knuder;
at sammenligne (640) DNS-navnene af hver af de opdagede knuder med DNS-navne af de overvågede knuder og de tidligere opdagede knuder;
at sammenligne (650) sysnames af hver af de opdagede knuder med sysnames af de overvågede knuder og de tidligere opdagede knuder;
at bestemme (660) duplikatknuder, der er duplikater af de tidligere opdagede knuder og/eller de overvågede knuder baseret på sammenligningen af IP-adresserne, MAC-adresserne, DNS-navnene og sysnames,
hvor, til bestemmelsen, hver af de opdagede knuder er tildelt et knude-ID, og fremgangsmåden endvidere omfatter at tildele et MatchIndex til hvert knude-ID, hvor MatchIndexet angiver sandsynligheden for et match mellem den opdagede knude og en af de overvågede knuder og de tidligere opdagede knuder.

2. Fremgangsmåde ifølge krav 1, endvidere omfattende at kassere (435) duplikatknuderne.

3. Fremgangsmåde ifølge krav 1, endvidere omfattende at tildele en prioritet til hvert af sammenligningstrinnene, der bestemmer en udførelsesrækkefølge.

4. Fremgangsmåde ifølge krav 1, hvor bestemmelsen udføres ved anvendelse af en flerhed af duplikatdetektorer, hvor hver duplikatdetektor er konfigureret til at udføre en af sammenligningerne, og omfatter at udføre den følgende formel til beregning af et slutresultat med hensyn til, om en opdaget knude er et duplikat:

$d_1.\text{IsDuplicate}()*d_1.\text{Priority}+\ldots+d_n.\text{IsDuplicate}()*d_n.\text{Priority}$

hvor d_1 er en første duplikatdetektor, d_2 er en anden duplikatdetektor, ..., og d_n er en n^te duplikatdetektor, d_n.IsDuplicate() er en funktion, der repræsenterer den n^te duplikatdetektors konklusion af, om den opdagede knude er et duplikat, og d_n.Priority er en prioritet, der er tildelt til hver af den n^te duplikatdetektor.

5. Fremgangsmåde ifølge et af kravene 1 til 4, endvidere omfattende at gruppere duplikatknuderne ved knude-ID og summere MatchIndexerne for det samme knude-ID.

6. Indretning omfattende:
mindst en processor (22) og mindst en hukommelse (14) med computerprogramkode,
hvor den mindst ene hukommelse og computerprogramkoden er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til
at opdage (600) knuder i netværket;
at indsamle (610) en liste af internetprotokol, IP, -adresser, MAC, media access control, -adresser, domænenavnsystem, DNS, -navne og sysnames for hver af knuderne, der opdages i netværket;
hvor den mindst ene processor endvidere er konfigureret til at udføre:
en IP-duplikatdetektor (16, 201), der er konfigureret til at sammenligne (620) IP-adresserne af hver af de opdagede knuder med IP-adresser af overvågede knuder og tidligere opdagede knuder;
en MAC-duplikatdetektor (16, 202), der er konfigureret til at sammenligne (630) MAC-adresserne af hver af de opdagede knuder med MAC-adresser af de overvågede knuder og de tidligere opdagede knuder;
en DNS-duplikatdetektor (16, 203), der er konfigureret til at sammenligne (640) DNS-navnene af hver af de opdagede knuder med DNS-navne af de overvågede knuder og de tidligere opdagede knuder;
en navneduplikatdetektor (16, 204), der er konfigureret til at sammenligne (650) sysnames af hver af de opdagede knuder med sysnames af de overvågede knuder og de tidligere opdagede knuder;

hvor den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til
at bestemme (660) duplikatknuder, der er duplikater af de tidligere opdagede knuder og/eller de overvågede knuder baseret på resultatet af sammenligningen af IP-duplikatdetektoren, MAC-duplikatdetektoren, DNS-duplikatdetektoren og navneduplikatdetektoren,
hvor, til bestemmelsen, hver af de opdagede knuder er tildelt et knude-ID, og den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til at tildele et MatchIndex til hvert knude-ID, hvor MatchIndexet angiver sandsynligheden for et match mellem den opdagede knude og en af de overvågede knuder og de tidligere opdagede knuder.

7. Indretning ifølge krav 6, hvor den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til at kassere (435) duplikatknuderne.

8. Indretning ifølge krav 6, hvor den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til at tildele en prioritet til hver af IP-duplikatdetektoren, MAC-duplikatdetektoren, DNS-duplikatdetektoren og navneduplikatdetektoren, der bestemmer en udførelsesrækkefølge.

9. Indretning ifølge krav 6, hvor den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processer, til i det mindste at få indretningen til at bestemme duplikatknuderne ved at udføre den følgende formel til beregning af et slutresultat med hensyn til, om en opdaget knude er et duplikat:

$$d_1.\mathrm{IsDuplicate}()*d_1.\mathrm{Priority}+\ldots+d_n.\mathrm{IsDuplicate}()*d_n.\mathrm{Priority}$$

hvor d_1 er en første duplikatdetektor, d_2 er en anden duplikatdetektor, ..., og d_n er en n^{te} duplikatdetektor, d_n.IsDuplicate() er en funktion, der repræsenterer den n^{te} duplikatdetektors konklusion af, om den opdagede knude er et duplikat, og d_n.Priority er en prioritet, der er tildelt til hver af den n^{te} duplikatdetektor.

10. Indretning ifølge et af kravene 6 til 9, hvor den mindst ene hukommelse og computerprogramkoden endvidere er konfigureret, med den mindst ene processor, til i det mindste at få indretningen til at gruppere duplikatknuderne ved knude-ID og summere MatchIndexerne for det samme knude-ID.

11. Indretning ifølge krav 6, hvor en vægt er tildelt til hver af IP-duplikatdetektoren, MAC-duplikatdetektoren, DNS-duplikatdetektoren og navneduplikatdetektoren, hvor vægten angiver pålideligheden af resultatet, der tilvejebringes af de respektive duplikatdetektorer.

12. Computerprogram, der er nedlagt på et computerlæsbart medium, der omfatter instruktioner, som, når computerprogrammet udføres af en processor, får processoren til at udføre trinnene med:
at opdage (600), ved hjælp af en netværksovervågningsindretning, knuder i et netværk;
at indsamle (610) en liste af internetprotokol, IP, -adresser, MAC, media access control, -adresser, domænenavnsystem, DNS, -navne og sysnames for hver af knuderne, der opdages i netværket;
at sammenligne (620) IP-adresserne af hver af de opdagede knuder med IP-adresser af overvågede knuder og tidligere opdagede knuder;
at sammenligne (630) MAC-adresserne af hver af de opdagede knuder med MAC-adresser af de overvågede knuder og de tidligere opdagede knuder;
at sammenligne (640) DNS-navnene af hver af de opdagede knuder med DNS-navne af de overvågede knuder og de tidligere opdagede knuder;
at sammenligne (650) sysnames af hver af de opdagede knuder med sysnames af de overvågede knuder og de tidligere opdagede knuder; og
at bestemme (660) duplikatknuder, der er duplikater af de tidligere opdagede knuder og/eller de overvågede knuder baseret på sammenligningen af IP-adresserne, MAC-adresserne, DNS-navnene og sysnames,
hvor, til bestemmelsen, hver af de opdagede knuder er tildelt et knude-ID, og fremgangsmåden endvidere omfatter at tildele et MatchIndex til hvert knude-ID, hvor MatchIndexet angiver sandsynligheden for et match mellem den opdagede knude og en af de overvågede knuder og de tidligere opdagede knuder.

DRAWINGS

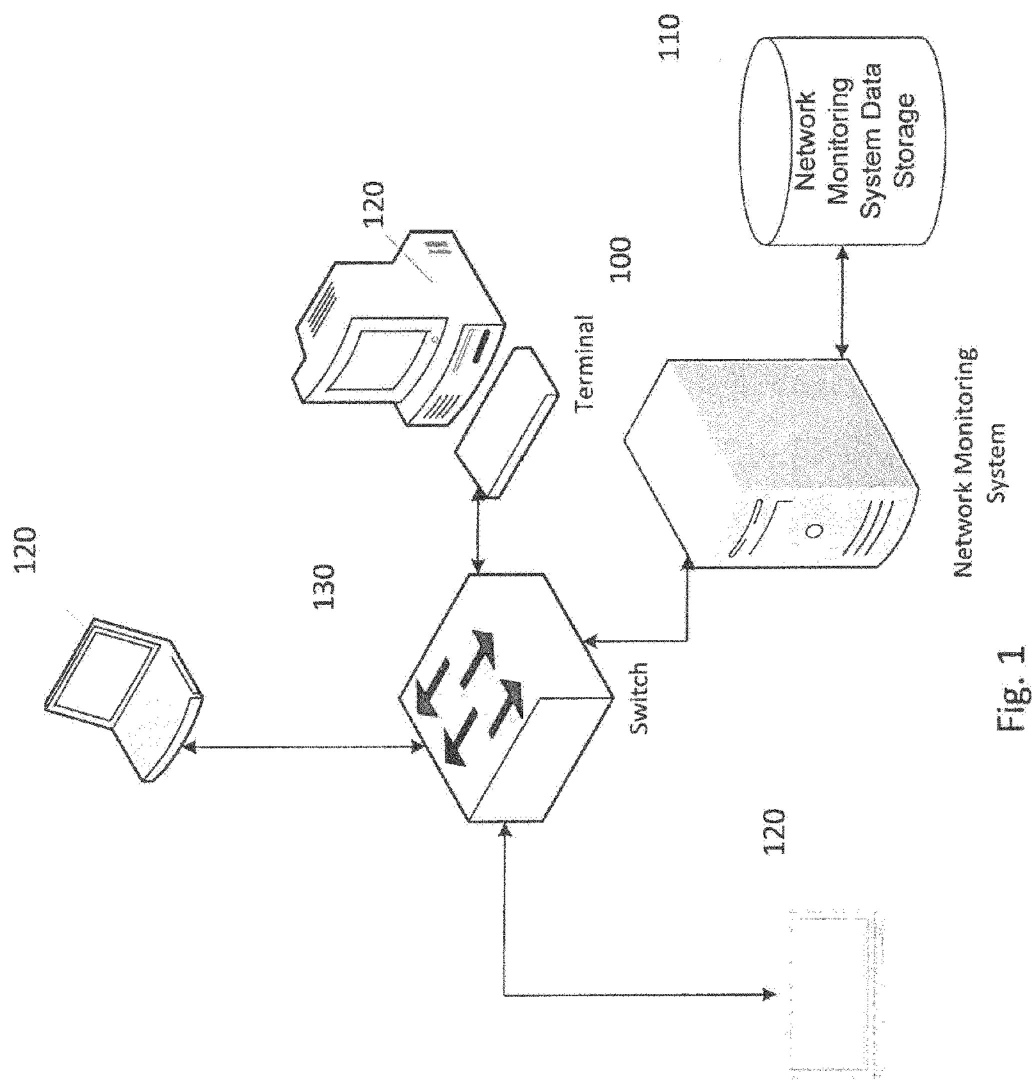
110
Network Monitoring System Data Storage
120
100
Terminal
Network Monitoring System
120
130
Switch
120

Fig. 1

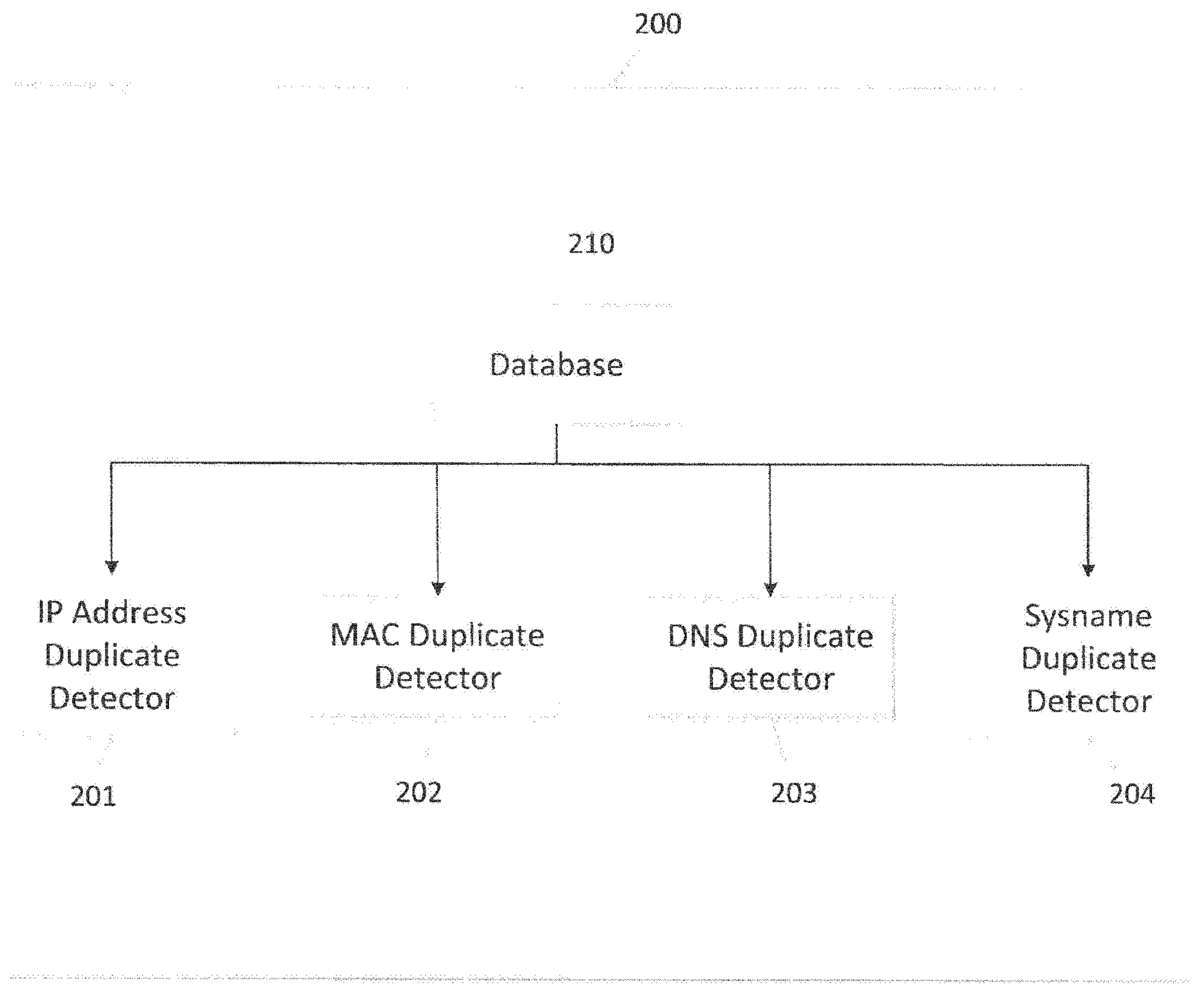
200
210
Database
IP Address Duplicate Detector
MAC Duplicate Detector
DNS Duplicate Detector
Sysname Duplicate Detector
201
202
203
204

Fig. 2

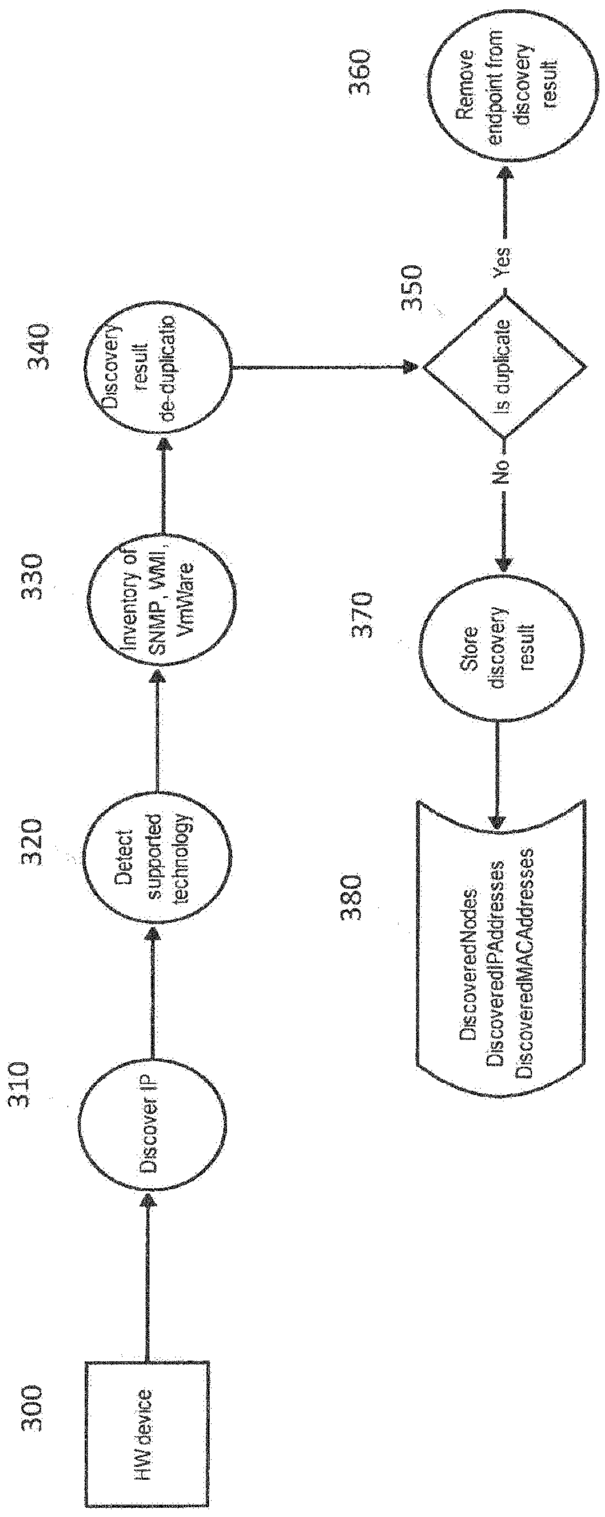
300
HW device
310
Discover IP
320
Detect supported technology
330
Inventory of SNMP, WMI, VmWare
340
Discovery result de-duplicatio
350
Is duplicate
Yes
No
360
Remove endpoint from discovery result
370
Store discovery result
380
DiscoveredNodes
DiscoveredIPAddresses
DiscoveredMACAddresses

Fig. 3

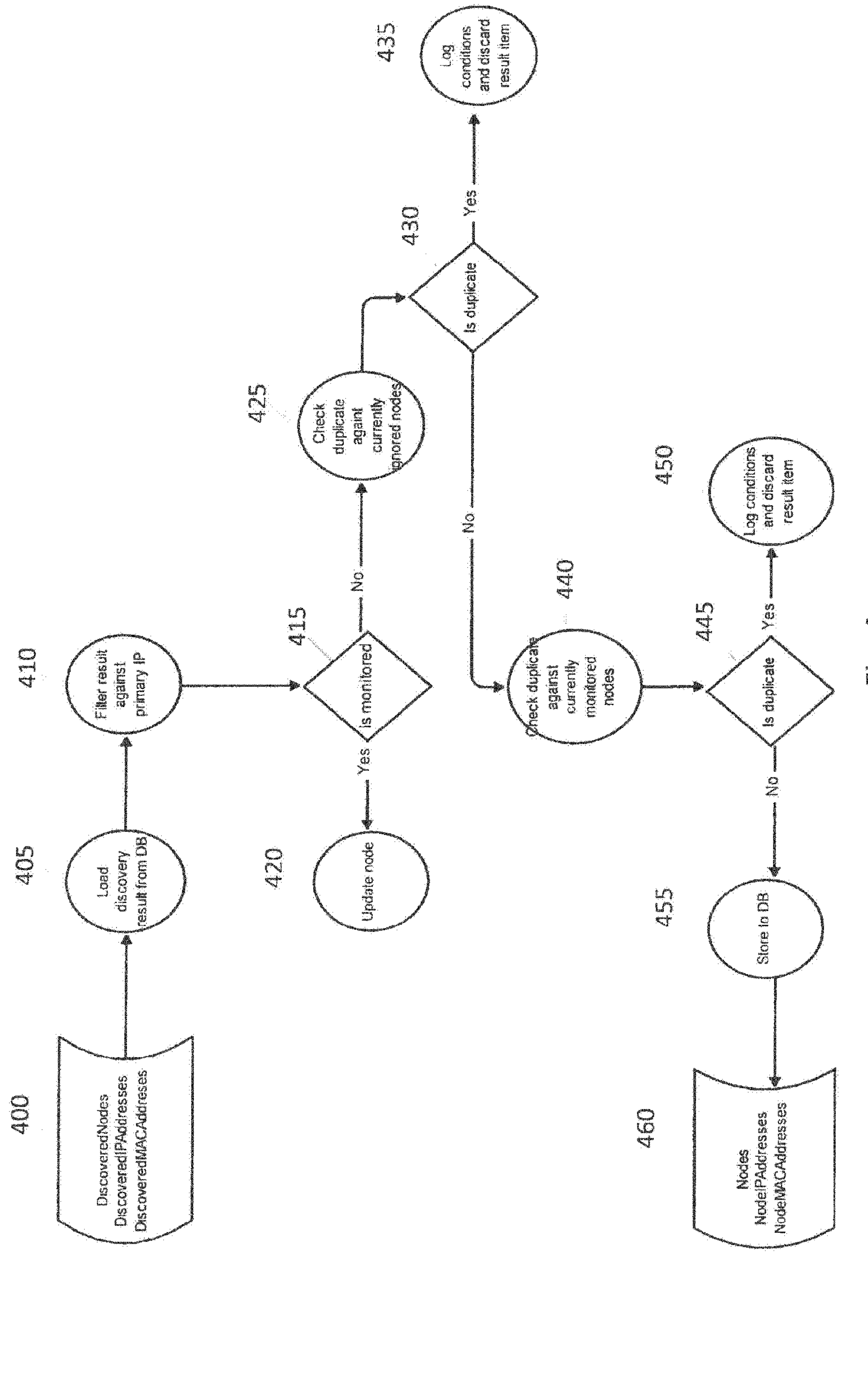
400
DiscoveredNodes
DiscoveredIPAddresses
DiscoveredMACAddresses
405
Load discovery result from DB
410
Filter result against primary IP
415
Is monitored
Yes
No
420
Update node
425
Check duplicate against currently ignored nodes
430
Is duplicate
Yes
No
435
Log conditions and discard result item
440
Check duplicate against currently monitored nodes
445
Is duplicate
Yes
No
450
Log conditions and discard result item
455
Store In DB
460
Nodes
NodeIPAddresses
NodeMACAddresses

Fig. 4

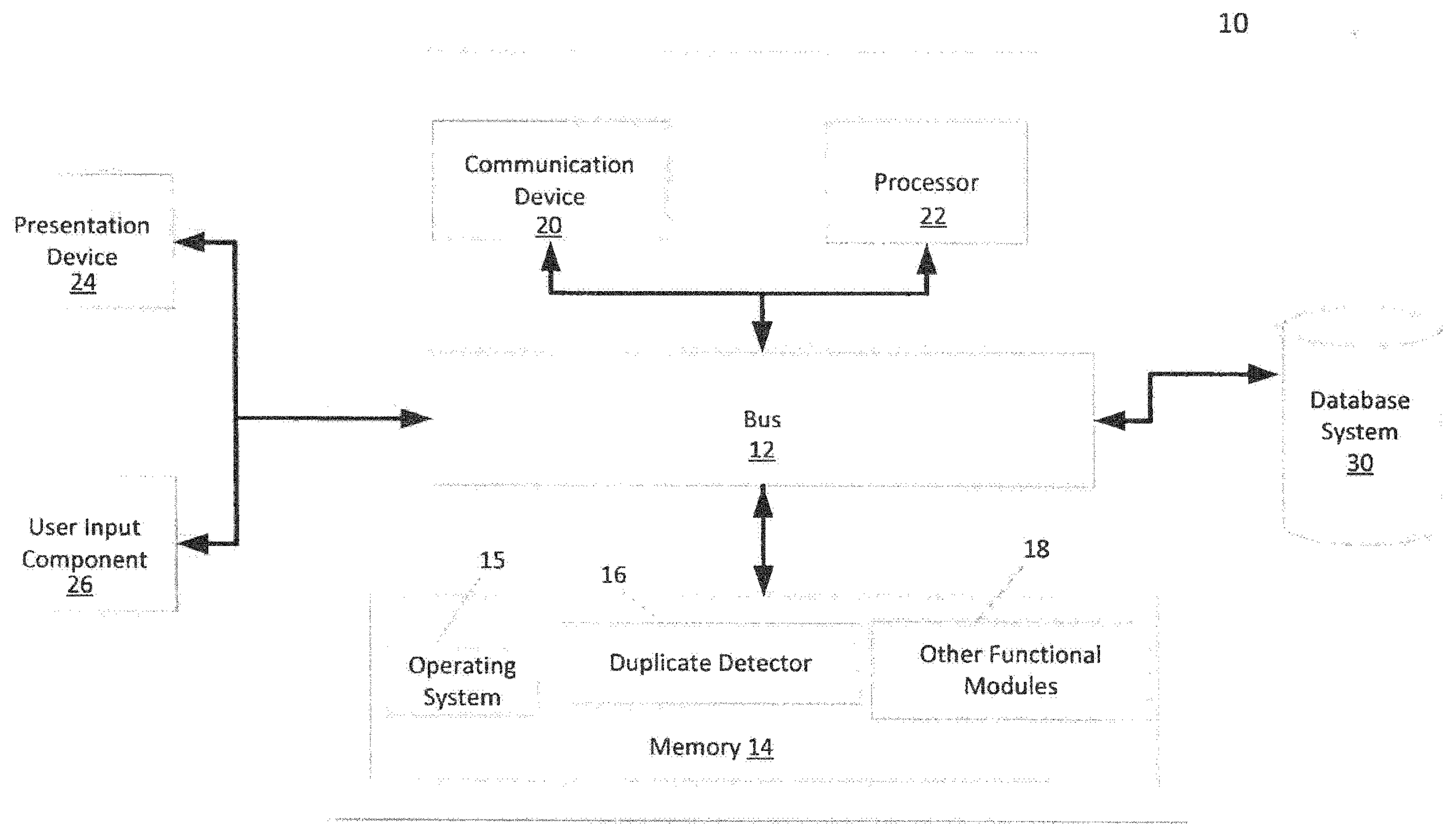
10
Presentation Device 24
User Input Component 26
Communication Device 20
Processor 22
Bus 12
Database System 30
15
16
18
Operating System
Duplicate Detector
Other Functional Modules
Memory 14

Fig. 5

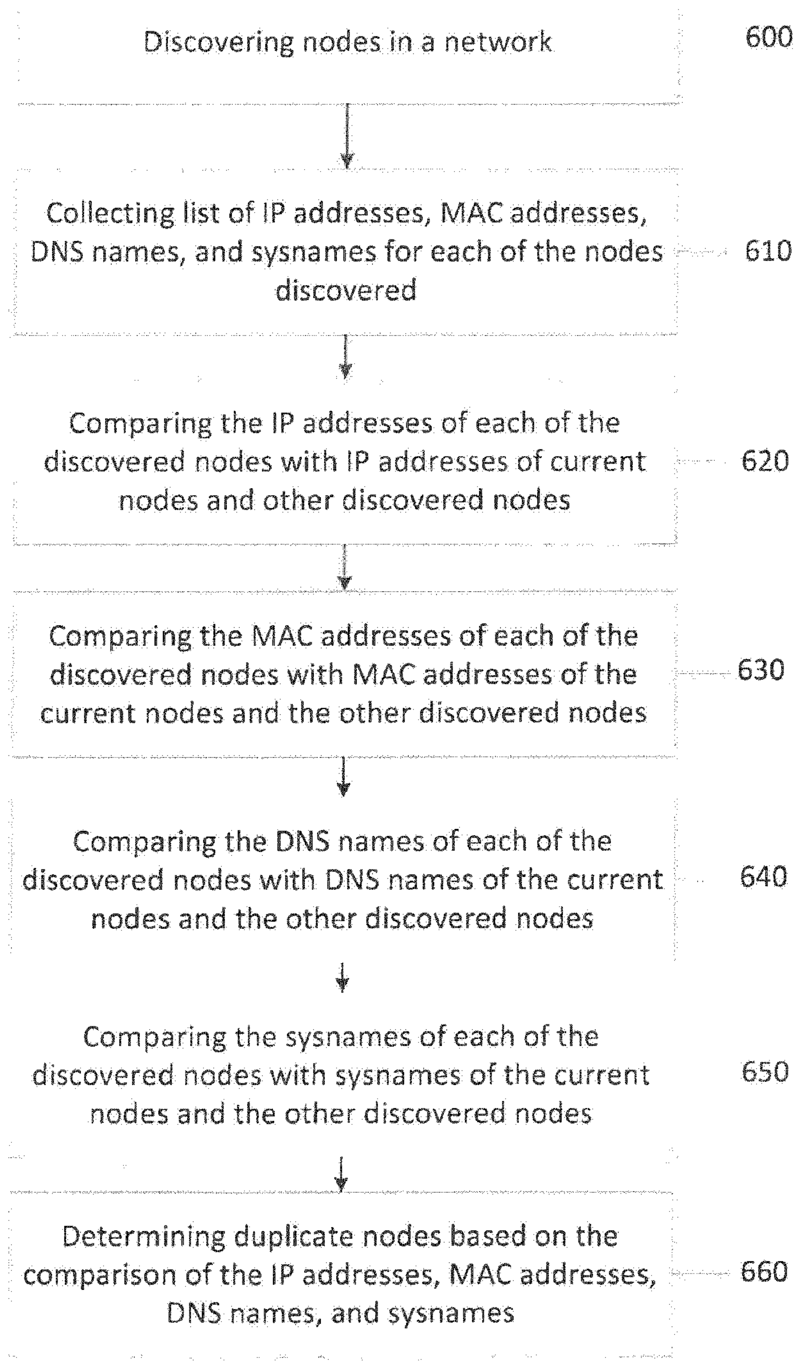
Discovering nodes in a network
600
Collecting list of IP addresses, MAC addresses, DNS names, and sysnames for each of the nodes discovered
610
Comparing the IP addresses of each of the discovered nodes with IP addresses of current nodes and other discovered nodes
620
Comparing the MAC addresses of each of the discovered nodes with MAC addresses of the current nodes and the other discovered nodes
630
Comparing the DNS names of each of the discovered nodes with DNS names of the current nodes and the other discovered nodes
640
Comparing the sysnames of each of the discovered nodes with sysnames of the current nodes and the other discovered nodes
650
Determining duplicate nodes based on the comparison of the IP addresses, MAC addresses, DNS names, and sysnames
660

Fig. 6