



(51) International Patent Classification:

H04L 29/06 (2006.01) *H04L 29/08* (2006.01)

(21) International Application Number:

PCT/US2017/034137

(22) International Filing Date:

24 May 2017 (24.05.2017)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

15/181,041 13 June 2016 (13.06.2016) US

(71) Applicant: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).

(72) Inventors: **BODAS, Devadatta**; 35224 7th Avenue SW, Federal Way, Washington 98023-8144 (US). **SONG, Justin J.**; 6923 47th Loop NE, Olympia, Washington 98516 (US). **RAJAPPA, Muralidhar**; 3195 W. Tyson Place,

Chandler, Arizona 85226 (US). **HOFFMAN, Andy**; 231 Clay Court SE, Olympia, Washington 98513 (US).

(74) Agent: **BARKER, Aaron D.**; Stoel Rives LLP, 201 South Main, Suite 1100, Salt Lake City, Utah 84111 (US).

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

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK,

(54) Title: SYSTEMS AND METHODS FOR DETECTING A ZOMBIE SERVER

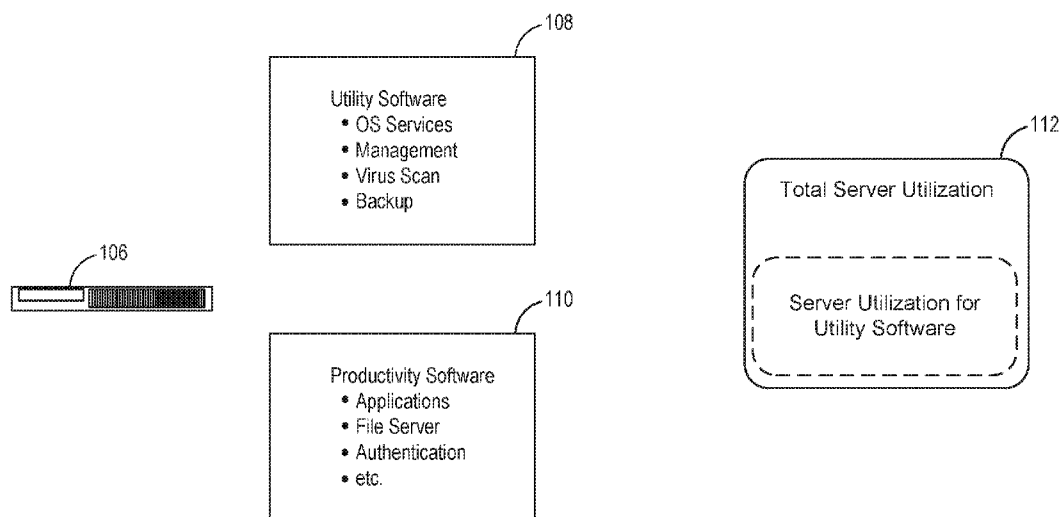


FIG. 1

(57) Abstract: A zombie server can be detected. Detecting a zombie server can include receiving, at a server, network traffic and calculating a percentage of the network traffic as being productivity software layer 7 protocols every first time interval. Detecting a zombie server can also include marking the server as a zombie server based on the percentage every second time interval and processing the network traffic at the server to perform a number of actions by the productivity software.

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).

Declarations under Rule 4.17:

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

Published:

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

SYSTEMS AND METHODS FOR DETECTING A ZOMBIE SERVER

Technical Field

[0001] This application relates to systems and methods for detecting a zombie or comatose server.

Background

[0002] In some organizations, users purchase servers that are maintained by information technology (IT) departments. The servers may be located in facilities such as data centers. The physical hardware and the security of the servers are maintained by facility managers. Servers may be purchased for specific applications or projects. Over time and as the projects are completed, need for the applications may diminish. As such, racks and data centers may host unused servers. These servers are referred to as zombie servers. The terms “zombie server” and “comatose server” can be used interchangeably. Zombie servers may be regularly maintained by IT specialists even though the zombie servers are no longer used by customers. Besides IT resources, zombie servers occupy rack space, consume energy (e.g., power), and burden cooling resources. Expenses on zombie servers can be considered as wasted resources because the zombie servers do not contribute to productivity. In some environments, zombie servers may occupy 10% to 20% of servers in a data center and/or rack. A similar situation exists even in virtual environments where resources get allocated to an application and remain unused. For example, a virtual server may be created without being used. The virtual server may be considered a virtual zombie server. As used herein, the term “zombie server” also includes virtual zombie servers.

[0003] In a number of previous approaches, zombie servers are identified by tracking processor utilization. However, tracking processor utilization to identify zombie servers may include a number of limitations. For example, a processor of a server may process services that are not consumed by customers (e.g., users). For example, management software executed on the server may contribute to processor utilization without a customer utilizing the server. Management software and/or services that may occur in zombie servers that may be carried out without a consumer's request may include backup services, security services, and virtualization services, among others. That is, management services may contribute

to processor utilization such that a server receiving the management services may appear to be utilized by a customer when the server is not actually utilized by the customer.

[0004] Identifying a zombie server may be difficult because non-zombie servers (e.g., servers that are utilized by customers) that are used for productivity applications may also have low utilization. That is, there may be little difference in processor utilization as it corresponds to zombie servers and non-zombie servers.

Brief Description of the Drawings

[0005] FIG. 1 is a system diagram of a server according to one embodiment.

[0006] FIG. 2 is a system diagram of a data center according to one embodiment.

[0007] FIG. 3 is a system diagram of a data center according to one embodiment.

[0008] FIG. 4 is a flowchart of an approach to identifying a zombie server according to one embodiment.

[0009] FIG. 5 is a flowchart of an approach to identifying a zombie server according to one embodiment.

[0010] FIG. 6 is a flowchart of an approach to identifying a zombie server according to one embodiment.

[0011] FIG. 7 is a flowchart of an approach to identifying a zombie server according to one embodiment.

[0012] FIG. 8 is a flowchart for identifying a zombie server and taking an action based on the identification according to one embodiment.

Detailed Description of Preferred Embodiments

[0013] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. In the drawings, similar symbols typically identify similar components, unless context dictates otherwise. The illustrative embodiments described in the detailed description, drawings, and claims are not meant to be limiting. Other embodiments may be utilized, and other changes may be made, without departing from the spirit or scope of the subject matter presented here.

[0014] The present disclosure provides systems and methods for detecting a zombie or comatose server. Detecting a zombie server can include labeling a plurality of processes as utility software and calculating a utilization of utility software on the plurality of processes executed in one or more processing resources during an interval of time. Detecting a zombie server can also include calculating a server

utilization of the one or more processing resources during the interval of time, determining whether a difference between the utilization of utility software and the server utilization is greater than a threshold, and identifying a server that hosts the processing resource as a zombie server based on a determination that the difference is smaller than the threshold.

[0015] In some embodiments, a system for detecting a zombie server can include a first server comprising memory and at least one processing resource, the first server comprising instructions to sample a plurality of process identifications (IDs) of a plurality of processes executed on a processing resource of a second server, wherein each of the plurality of process IDs is sampled during a first time interval and calculate a plurality of central processing unit (CPU) run times for the plurality of process IDs based on the first time interval, each of the plurality of CPU run times corresponding to one of the plurality of process IDs. The first server also comprises instructions to calculate a utilization of utility software on the plurality of processes executed in the processing resource repetitively every second time interval, calculate a server utilization of the processing resource repetitively every second time interval, and determine whether the second server is a zombie server based on the utilization of utility software and the server utilization repetitively every third time interval.

[0016] Detecting a zombie server can also include instructions to query a server on active processes running on a processing resource of a server using a task management program repetitively every first time interval and calculate a utilization by utility software based on the active processes that are associated with utility software, repetitively every first time interval. The server can also include instructions to calculate a server utilization based on the active processes every first time interval, compare the utilization of the utility software and the server utilization every second time interval, and mark the server as a zombie server based on the comparison every second time interval.

[0017] In some embodiments, a system for detecting a zombie server can include a server comprising a processing resources and a computing device. The computing device can be configured to receive network traffic, calculate a percentage of the network traffic as being utility software layer 7 protocols every first time interval, mark the server as a zombie server based on the percentage every second time interval, and distribute the network traffic to the server.

[0018] Detecting a zombie server can further include receiving, at a server, network traffic, calculating a percentage of the network traffic as being productivity software layer 7 protocols every first time interval, marking the server as a zombie server based on the percentage every second time interval, and processing the network traffic at the server to perform a number of actions by the productivity software.

[0019] Detecting a zombie server can also include receiving, at a second computing device, network traffic sent to a first computing device, wherein the first computing device is configured to distribute the network traffic to a plurality of servers, calculating a percentage of a portion of the network traffic as being utility software layer 7 protocols every first time interval, for each of the plurality of servers, wherein the portion of the network traffic is network traffic to a respective server, marking each of the plurality of servers as zombie servers based on a respective percentage every second time interval, and forwarding the network traffic to the first computing device.

[0020] In a number of embodiments described herein, zombie servers may be identified by tracking utility software utilization and/or productivity software utilization. As used herein, productivity software utilization can describe user applications, file server applications, and/or authentication applications, among others. Utility software can describe operating system (OS) services, management services, security services, and/or backup services, among others.

[0021] In some embodiments, productivity software and/or utility software can be defined by customers and/or IT specialists. For example, processes, subprocesses, and/or threads associated with software executed on a server may be categorized as productivity software or utility software by customers and/or IT specialists.

[0022] Software utilization (e.g., utility software and/or productivity software) can be defined as a duration of time that the software is executed on at least one of a plurality of processors of a server. For example, utility software utilization is a duration of time that the utility software is executed on at least one of the plurality of processors of a server.

[0023] In some examples, zombie servers may be identified by comparing utility software utilization to server utilization and/or productivity software utilization. For example, a zombie server may be identified in response to determining that productivity software utilization is zero.

[0024] As used herein, the terms “processor utilization” and “server utilization” may be used interchangeably. The term “server utilization” may describe a duration of time that at least one of a plurality of processors is utilized. For example, a server utilization may describe the utilization of one, two, three, or more processors.

[0025] Embodiments may include various steps, which may be embodied in machine-executable instructions to be executed by a computer system. A computer system includes one or more general-purpose or special-purpose computers (or other electronic devices). The computer system may include hardware components that include specific logic for performing the steps or may include a combination of hardware, software, and/or firmware.

[0026] Embodiments may also be provided as a computer program product including a computer-readable medium having stored thereon instructions that may be used to program a computer system or other electronic device to perform the processes described herein. The computer-readable medium may include, but is not limited to: hard drives, floppy diskettes, optical disks, CD-ROMs, DVD-ROMs, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, solid-state memory devices, or other types of media/computer-readable media suitable for storing electronic instructions.

[0027] Computer systems and the computers in a computer system may be connected via a network. Suitable networks for configuration and/or use as described herein include one or more local area networks, wide area networks, metropolitan area networks, and/or “Internet” or IP networks, such as the World Wide Web, a private Internet, a secure Internet, a value-added network, a virtual private network, an extranet, an intranet, or even standalone machines which communicate with other machines by physical transport of media (a so-called “sneakernet”). In particular, a suitable network may be formed from parts or entireties of two or more other networks, including networks using disparate hardware and network communication technologies. The network may include a wireless network.

[0028] One suitable network includes a server and several clients; other suitable networks may contain other combinations of servers, clients, and/or peer-to-peer nodes, and a given computer system may function both as a client and as a server. Each network includes at least two computers or computer systems, such as the server and/or clients. A computer system may include a workstation, laptop computer, disconnectable mobile computer, server, mainframe, cluster, so-called

“network computer” or “thin client,” tablet, smart phone, personal digital assistant or other hand-held computing device, “smart” consumer electronics device or appliance, medical device, or a combination thereof.

[0029] The network may include communications or networking software, such as the software available from Novell, Microsoft, Artisoft, and other vendors, and may operate using TCP/IP, SPX, IPX, or other protocols over twisted pair, coaxial, or optical fiber cables, telephone lines, radio waves, satellites, microwave relays, modulated AC power lines, physical media transfer, and/or other data transmission “wires” and/or wireless protocols known to those of skill in the art. The network may encompass smaller networks and/or be connectable to other networks through a gateway or similar mechanism.

[0030] Each computer system includes at least a processor and a memory; computer systems may also include various input devices and/or output devices. The processor may include a general-purpose device, such as an Intel®, AMD®, or other “off-the-shelf” microprocessor. The processor may include a special-purpose processing device, such as an ASIC, a SoC, a SiP, an FPGA, a PAL, a PLA, an FPLA, a PLD, or other customized or programmable device. The memory may include static RAM, dynamic RAM, flash memory, one or more flip-flops, ROM, CD-ROM, disk, tape, or a magnetic, an optical, or other computer storage medium. The input device(s) may include a keyboard, mouse, touch screen, light pen, tablet, microphone, sensor, or other hardware with accompanying firmware and/or software. The output device(s) may include a monitor or other display, printer, speech or text synthesizer, switch, signal line, or other hardware with accompanying firmware and/or software.

[0031] The computer systems may be capable of using a floppy drive, a tape drive, an optical drive, a magneto-optical drive, or other means to read a storage medium. A suitable storage medium includes a magnetic, an optical, or other computer-readable storage device having a specific physical configuration. Suitable storage devices include floppy disks, hard disks, tape, CD-ROMs, DVDs, PROMs, random access memory, flash memory, and other computer system storage devices. The physical configuration represents data and instructions which cause the computer system to operate in a specific and predefined manner as described herein.

[0032] Suitable software to assist in implementing a number of examples described herein is readily provided by those of skill in the pertinent art(s) using the teachings presented here and programming languages and tools, such as Java, Pascal, C++, C, database languages, APIs, SDKs, assembly, firmware, microcode, and/or other languages and tools. Suitable signal formats may be embodied in analog or digital form, with or without error detection and/or correction bits, packet headers, network addresses in a specific format, and/or other supporting data readily provided by those of skill in the pertinent art(s).

[0033] Several aspects of the embodiments described will be illustrated as software modules or components. As used herein, a software module or component may include any type of computer instruction or computer executable code located within a memory device. A software module may, for instance, include one or more physical or logical blocks of computer instructions, which may be organized as a routine, program, object, component, data structure, etc., that perform one or more tasks or implement particular abstract data types.

[0034] In certain embodiments, a particular software module may include disparate instructions stored in different locations of a memory device, different memory devices, or different computers, which together implement the described functionality of the module. Indeed, a module may include a single instruction or many instructions, and may be distributed over several different code segments, among different programs, and across several memory devices. Some embodiments may be practiced in a distributed computing environment where tasks are performed by a remote processing device linked through a communications network. In a distributed computing environment, software modules may be located in local and/or remote memory storage devices. In addition, data being tied or rendered together in a database record may be resident in the same memory device, or across several memory devices, and may be linked together in fields of a record in a database across a network.

[0035] Much of the infrastructure that can be used according to a number of examples described herein is already available, such as: general-purpose computers, computer programming tools and techniques, computer networks and networking technologies, digital storage media, authentication, access control, and other security tools and techniques provided by public keys, encryption, firewalls, and/or other means.

[0036] FIG. 1 is a system diagram of a server according to one embodiment. FIG. 1 includes a server 106, utility software 108, and productivity software 110. The utility software 108 and/or the productivity software 110 can be executed on the server 106.

[0037] As previously described, the utility software 108 can include OS services, management services, virus scan services (e.g., security services), and/or backup services, among other types of services. The productivity software 110 can include user applications, file server applications, and/or authentication applications, among other types of applications.

[0038] The utilization of the utility software 108 and the utilization of the productivity software 110 can define a server utilization 112. In some embodiments, the server utilization 112 can be defined as the average utilization of a plurality of processors in the server 106. A utilization of each of the plurality of processors can be defined as the unhalted cycles of a processor in a time period divided by the total processor cycles of the time period. In some embodiments, a utilization of each of the plurality of processors can be defined as a first duration of time of unhalted cycles of the processor divided by a second duration of time of the total processor cycles.

[0039] FIG. 2 is a system diagram of a data center according to one embodiment. FIG. 2 includes servers 206-1, 206-2, ..., 206-N (referred to generally as servers 206), which are analogous to the server 106 in FIG. 1. FIG. 2 also includes a management unit 214. The servers 206 can be part of a rack and/or a data center. Each of the servers 206 can include a plurality of processing resources such as central processing units (CPUs) and/or graphical processing units (GPUs), among others. In some examples, the servers 206 can be dispersed among separate racks, data centers, and/or locations.

[0040] Each of the servers 206 can be monitored to determine whether any of the servers 206 are zombie servers. For example, each of the servers 206 can monitor itself to determine whether it is a zombie server. Once one of the servers 206 determines that it is a zombie server, any one of the servers 206 can report the finding to the management unit 214.

[0041] In some examples, each of the servers 206 can monitor a different one of the servers 206 to determine whether the monitored server is a zombie server. Once a monitoring server from the servers 206 determines that the monitored server

is a zombie server, the monitoring server can report the finding to the management unit 214.

[0042] In a different embodiment, the management unit 214 can monitor each of the servers 206 to determine whether the monitored server is a zombie server. As used herein, monitoring the servers 206 can include monitoring the software and/or hardware of the servers 206 and/or the messages coming to and/or from the servers 206.

[0043] In some examples, a zombie server can be identified based on the network traffic coming to and/or from the servers 206. For example, the management unit 214 can monitor a network device to identify network traffic directed to and/or from the servers 206. The network device can be a router, a switch, a bridge, and/or a gateway, among other network devices. In some examples, the network device can be a computing system that intercepts network traffic directed to and/or from a router, a switch, a bridge, and/or a gateway, among other network devices. FIG. 5 further describes identifying a zombie server based on network traffic.

[0044] FIG. 3 is a system diagram of a data center according to one embodiment. FIG. 3 includes servers 306-1, 306-2, ..., 306-N (e.g., referred to generally as servers 306). FIG. 3 also includes a switch 307, a network device 309, a management unit 314, and an Internet 315.

[0045] The switch 307 can distribute network traffic to the servers 306. As used herein, the switch 307 represents any type of device that can be configured to distribute network traffic to the servers 306. For example, the switch 307 can represent a switch and/or a router. The switch 307 can be an out-of-band switch.

[0046] The network device 309 can intercept network traffic directed to the switch 307. The network device 309 can include hardware and/or computer readable instructions configured to intercept the network traffic.

[0047] The management unit 314 can include hardware and/or computer readable instructions to determine whether any of the servers 306 are zombie servers. In some examples, the management unit 314, the network device 309, and/or the switch 307 can be configured as one or more computing devices. For example, the management unit 314, the network device 309, and/or the switch 307 can be integrated into a single computing device or multiple computing devices.

[0048] For example, a system for detecting a zombie server can include a server 306-1 comprising a processing resources and a computing device (e.g., the switch 307). The computing device can be configured to receive network traffic, calculate a percentage of the network traffic as being utility software layer 7 protocols every first time interval, mark the server as a zombie server based on the percentage every second time interval, and distribute the network traffic to the server 306-1.

[0049] The utility software layer 7 protocols can be received at least on a predetermined port. The utility software layer 7 protocols can be identified based on a header of a plurality of packets comprising the network traffic. The utility software layer 7 protocols can also be identified based on a payload of a plurality of packets comprising the network traffic.

[0050] The computing device can also be configured to decrypt at least one of a header and a payload of a packet. The computing device can also be configured to identify the utility software layer 7 protocols based on at least one of the header and the payload of the packet.

[0051] In a different embodiment, the server 306-1 can be configured to receive, at a server 306-1, network traffic and calculate a percentage of the network traffic as being productivity software layer 7 protocols every first time interval. The server 306-1 can also be configured to mark the server 306-1 as a zombie server based on the percentage every second time interval, and process the network traffic at the server to perform a number of actions by the productivity software.

[0052] The server 306-1 can also be configured to calculate the percentage of the network traffic, and the marking of the server as the zombie server occurs at a management unit. Calculating the percentage can further comprise determining that a portion of the plurality of packets comprising the network traffic is associated with productivity software layer 7 protocols. Calculating the percentage can further comprise dividing the portion of the plurality of packets by the plurality of packets.

[0053] Determining that the portion of the plurality of packets is associated with productivity software layer 7 protocols can further include identifying the layer 7 protocols from the network traffic and bucketizing the layer 7 protocols. Bucketizing the layer 7 protocols can further include identifying the portion of the plurality of packets that comprise the network traffic as corresponding to productivity software layer 7 protocols.

[0054] In some embodiments, the network device 309 can include a computer-readable medium having instructions stored therein, the instructions comprising receiving network traffic sent to a first computing device (e.g., the switch 307), wherein the first computing device is configured to distribute the network traffic to a plurality of servers 306 and calculating a percentage of a portion of the network traffic as being utility software layer 7 protocols every first time interval, for each of the plurality of servers 306, wherein the portion of the network traffic is network traffic to a respective server 306-1. The instructions can also comprise marking each of the plurality of servers 306 as zombie servers based on a respective percentage every second time interval and forwarding the network traffic to the first computing device (e.g., the switch 307).

[0055] Calculating the percentage of the portion of the network traffic can be performed at a third computing device. The third computing device can be the management unit 314. Calculating the percentage of the portion of the network traffic can also include forwarding information regarding layer 7 protocols from the portion of the network traffic and utility software layer 7 protocols from the portion of the network traffic to the third computing device. Calculating the percentage of the portion of the network traffic includes dividing the utility software layer 7 protocols by the layer 7 protocols.

[0056] FIG. 4 is a flowchart of an approach to identifying a zombie server according to one embodiment. FIGS. 4, 5, 6, and 7 include multiple flowcharts that can be processed separately and/or sequentially. For example, flowcharts 423, 425, and 427 are shown in FIG. 4. In some examples, the flowcharts 423, 425, and 427 can be processed independently from each other. For example, the flowcharts 423, 425, and 427 can be processed and/or performed in separate threads, processors, and/or computing devices.

[0057] The flowcharts 423, 425, and 427 can also be performed sequentially. For example, the flowchart 427 can be performed in a same processor such that only one of the flowcharts 423, 425, and 427 is performed at a given time.

[0058] At the flowchart 423, it can be determined whether a context switch 422 has been performed. The context switch 422 can relate to the switching out of a first process for a second process in a processing resource.

[0059] In a number of examples, each of the processes including the first process and the second process can be labeled as processes originating from utility software

or productivity software. That is, each of the processes can be labeled as utility software and/or productivity software. The processes can be labeled by an IT specialist and/or a customer in anticipation of identifying a zombie server. In some examples, the processes can be labeled at run time. For example, processes that are associated with software that is directly accessed by a customer for productivity activities can be labeled productivity software while processes that are associated with management services can be labeled utility software.

[0060] At each context switch 422, a process identification (ID) of an old process and/or a time stamp can be read 424. Each of the processes that were labeled can be identified using a process ID. As such, an old process can be identified using a first process ID and a new process can be identified using a second process ID. The old process can denote a process that was switched out of context at the context switch 422. The new process can denote a process that was switched into context at the context switch 422. The old process can also be referred to as a first process, and the new process can be referred to as a second process. The time stamp can be associated with the old process and/or the new process such that each context switch 422 can be associated with a unique time stamp.

[0061] Associating a time stamp with the old process and/or the new process can lead to use of the time stamp as a stop time stamp for the old process and a start time stamp for the new process. For example, the start time for the new process can be given as $t(n+1)$ which can also be the stop time for the old process. The start time for the old process can be given as tn . The duration for which the old process ran is given as $t(n+1)-tn$.

[0062] The old time stamp, the new time stamp, the old process ID, and/or the new process ID can be used to calculate 426 a CPU run time of the old process and/or the new process. The CPU run time of the old process refers to a duration of time that the old process was executed on a processing resource for the duration of time spanning context switches 422 in which the old process was executed on a processing resource.

[0063] At each context switch 422, the CPU run time of the old process can be logged 428 in a memory resource of a management unit (e.g., the management unit 214 in FIG. 2) and/or the server on which the old process was executed. The CPU run time of the old process can be logged 428 to provide a tally of the CPU run times that are associated with the process ID of the old process at a later time.

[0064] For example, at a first context switch, a first time stamp (e.g., tn) can be read and associated with a first process with a first process ID, the first process being switched into context. At the first context switch, a first CPU run time can be calculated for a process that was switched out of context at the first context switch.

[0065] At a second context switch, a second time stamp (e.g., $t(n+1)$) can be read and associated with a second process with a second process ID, the first process being switched out of context and the second process being switched into context. At the second context switch, a second CPU run time (e.g., $PCPU_x$) of the context switch can be calculated by subtracting the first time stamp from the second time stamp ($t(n+1) - tn$). The result of the second CPU run time of the context switch can be added to a tally of CPU run times of CPU run times associated with a respective process ID. For example, $PCPU_x = PCPU_x + (t(n+1) - tn)$. The second CPU run time can be stored (e.g., logged 428) in the server hosting the second CPU run time.

[0066] After logging 428 the CPU run time, the old process, and/or the old process ID, the flowchart 423 can be processed again at a next context switch.

[0067] The flowchart 425 can include determining 430 whether a time interval T has elapsed. Determining 430 whether the time interval T has elapsed can include repetitively determining 430 whether the time interval T has elapsed until the time interval T has actually elapsed.

[0068] The time intervals T and X can be selected by an IT specialist. The time intervals T and X can be predetermined. The time intervals T and X can include separate beginning times and end times. The time span between the beginning time and the end time can define the time interval T. The time interval T can be a time over which the CPU run times are used to calculate 432 a server utilization and to calculate 434 a utilization by utility software. The time interval X can be an evaluation time in which a server is evaluated to determine whether the server is a zombie server. A determination of whether a server is a zombie server can be performed every time interval X 438.

[0069] A time interval T can be a millisecond, a second, or any interval thereof. A time interval X can be a second, an hour, a day, a plurality of days, or any interval thereof, among other possible time intervals. The time interval X can be greater than the time interval T.

[0070] If the time interval T has elapsed, then a server utilization can be calculated 432. The server utilization can be calculated 432 using the logged 428

CPU run times. For example, a utilization for each process ID associated with the CPU run times can be calculated to calculate 432 the server utilization. The utilization (e.g., Upu) for each of the process IDs can be calculated by dividing the corresponding CPU run time by the time interval T (e.g., Te) such that $Upu = PCPU_x/Te$.

[0071] Calculating 432 the server utilization (e.g., Su) over the time interval T (e.g., Te) can include summing utilization for the process IDs. That is, the $Upus$ associated with each of the process ID is summed to calculate 432 the server utilization (Su).

[0072] Calculating 434 the utilization (e.g., $Sutu$) by utility software can include summing all the utilization (e.g., Upu) for processes labeled utility software. The flowchart 425 can also include logging 436 the server utilization and the utilization by utility software. Logging 436 the server utilization can include adding the server utilization calculated over the time interval T to previously calculated server utilizations over different iterations of the time interval T . Logging 436 the utilization of the utility software can include adding the utilization of the utility software over the time interval T to the previously calculated server utilizations over different iterations of the time interval T .

[0073] The flowchart 427 can include determining 438 whether the time interval X has elapsed. The determination 438 can be made until the time interval X has elapsed. If the time interval X has elapsed, then the utilization by utility software and the server utilization can be compared 440.

[0074] The server for which the server utilization and the utilization for the utility software are calculated can be marked based on the comparison 440. Comparing the utilization by the utility software and the server utilization can include dividing the utilization by the utility software by the server utilization. If the result of the division is smaller than a predetermined threshold, then the server can be marked as a zombie server 442.

[0075] In other embodiments, comparing the server utilization and the utilization by the utility software can include determining whether a difference between the server utilization and the utilization by the utility software is greater than a predetermined threshold. As such, the server that hosts the processing resource can be identified as a zombie server based on a determination that the difference is smaller than a threshold.

[0076] FIG. 5 is a flowchart of an approach to identifying a zombie server according to one embodiment. FIG. 5 includes flowcharts 525, 527, and 529. If the flowcharts are implemented sequentially, then the flowchart 529 can be implemented before the flowchart 527 which can be implemented before the flowchart 525. The flowcharts 525 and 527 are analogous to the flowcharts 425 and 427 in FIG. 4.

[0077] The flowchart 529 can include determining 522 whether a time interval Y has elapsed. The determination 522 can be performed until the time interval Y has elapsed.

[0078] In a number of examples, the time interval X can be greater than the time interval T which is greater than the time interval Y. For example, the time interval X can be measured in days. The time interval T can be measured in seconds. The time interval Y can be measured in milliseconds, although other durations of times can be selected for the time intervals X, T, and Y. That is, in some examples, the time interval Y can be provided in multiples of milliseconds and the time interval T can be provided in intervals of seconds.

[0079] If the time interval Y has elapsed, then a process ID of an active process can be read 524. Reading 524 the process ID of an active process can include sampling a process ID of a process being executed on a processing resource of a server repetitively over the time interval Y. That is, an active process can be a process that is being executed on the processing resource of the server.

[0080] A CPU run time of the active process can be calculated 526. Calculating 526 the CPU run time can include determining the duration of time between the last time interval Y and the current time interval Y. The duration of time can be given as m . M may be equal to the time interval Y if the determination that the time interval Y elapsed occurs at the exact interval Y. M may also be equal to the duration of times between consecutive determinations that the time interval Y has elapsed. The CPU run time (e.g., $PCPU_x$) can be calculated as $PCPU_x = PCPU_x + m$. The flowcharts 525 and 527 can be processed as described by the flowcharts 425 and 427 in FIG. 4.

[0081] In the examples provided in FIGS. 1-7, a processing resource can include one or more processing resources. For example, the processing resource of a server can include one or more CPUs. As such, reading a process ID of a process being executed on a processing resource can include reading a plurality of process IDs being executed on the processing resources. As such, at every context switch

as described in FIG. 4 or at each time interval Y, a plurality of process IDs can be read. That is, each CPU can host a process and, as such, a process ID can be read for each process being executed on each of the CPUs in a server.

[0082] FIG. 6 is a flowchart of an approach to identifying a zombie server according to one embodiment. FIG. 6 includes flowcharts 627 and 631.

[0083] In the example provided in FIG. 6, existing tools are used to determine whether a server is a zombie server. The existing tools can include software and/or hardware to monitor processes at the application level. For example, existing tools may include task management programs. Task management programs may include Top and/or Perfmon.

[0084] The examples provided in FIG. 6 and FIG. 5 may differ in that the example provided in FIG. 5 may differentiate between productivity software that is performing utility tasks and productivity software that is performing productivity tasks, and the example provided in FIG. 6 may not. For example, even when an application (e.g., productivity software) is loaded, it may run its own processes for maintenance or security. For example, the application may run processes to check for updates, check if there are instant messages, and/or report that the application is active (e.g., alive). Such processes, although initiated by productivity software, may not perform productivity tasks but instead may perform utility tasks and, as such, may be considered as utility software. The example provided in FIG. 5 may classify productivity software that performs utility tasks as utility software.

[0085] At the flowchart 631 a determination 630 can be made as to whether the time interval T has elapsed. If the time interval T has elapsed, then the operating system (OS) can be queried 662 on active processes. The OS can be queried 662 using a task management program. In some examples, the result of the query 662 can be utilization of an active process or a CPU run time of the active process. If the result of the query 662 is a CPU run time, then a utilization of the active process can be calculated from the CPU run time as described above.

[0086] A server utilization can be calculated 664 from the utilization of the active process. A utilization by the utility software can be calculated 664 from the utilization of active processes that are marked as utility software. The server utilization and the utilization by utility software can be logged 668.

[0087] The flowchart 627 is analogous to the flowcharts 427 and 527 in FIGS. 4 and 5, respectively. As such, the utilization by the utility software and the server

utilization can be compared 640 and a server can be marked 642 as previously described in FIGS. 4 and 5.

[0088] FIG. 7 is a flowchart of an approach to identifying a zombie server according to one embodiment. FIG. 7 includes flowcharts 741, 743, and 745. The time interval X can be greater than the time interval T. The time interval X can be x days while the time interval T can be t seconds or t milliseconds.

[0089] In some embodiments, network traffic 752 associated with a server can be used to identify zombie servers. For example, the network traffic 752 can include the network traffic 752 to/from applications executing on a server. The network traffic 752 can be identified to determine whether the network traffic 752 is associated with utility software and/or productivity software. Identifying the type of the network traffic 752 coming to/from the server can provide the ability to determine whether productivity software is being executed on the server, which can in turn be used to identify zombie servers.

[0090] The flowchart 743 can include receiving incoming network traffic 752 at a computing device. The computing device can be a switch that can be out-of-band, a router, and/or a sniffer, among other computing devices that can be configured to receive network traffic. In some examples, the computing device can be configured to distribute the network traffic 752 to a plurality of servers.

[0091] The computing device can be a sniffer. As used herein, a sniffer can be an instrument, tool, hardware, and/or software that collects statistics on network packets received and sent. For example, a sniffer can intercept network traffic before the network traffic is distributed to the plurality of servers.

[0092] If the computing device receives incoming network traffic 752, then the OSI layer 7 protocols (e.g., application layer protocols) can be identified 754 from the network traffic 752. An application layer protocol is often associated with a particular TCP/IP port. In an embodiment, an administrator can identify network layer 7 protocols/ports that can be termed utility software. For example, the computing device can store a pre-canned list of layer 7 protocols that are termed utility software that can be modified. The pre-canned list of layer 7 protocols can include packets received via a port numbered 5723 for a system center operations manager (SCOM), for example.

[0093] The layer 7 protocols can be bucketized 756. Bucketizing 756 the layer 7 protocols can include separating the layer 7 protocols into buckets, groups, and/or

categories. For example, the layer 7 protocols can be grouped by rate of arrival or frequency of occurrence of each type of protocol. In some examples, the network traffic can be bucketized 756 by bucketizing the packets comprising the network traffic.

[0094] The flowchart 741 can include determining 730 whether a time interval T has elapsed. If the time interval T has elapsed, then the bucketized layer 7 protocols can be examined to calculate 750 a percentage of the utility software layer 7 protocols in time interval T. One of the buckets in the bucketized layer 7 protocols can include layer 7 protocols associated with utility software. The bucketized layer 7 protocols can also include a grouping of productivity software.

[0095] The flowchart 745 can include determining 738 whether the time interval X has elapsed. If the time interval X has elapsed, then it can be determined 758 whether the percentage of the utility software layer 7 protocols is greater than a predetermined threshold. In FIG. 7, the utility software layer 7 is referenced as US L7. If the percentage is greater than the predetermined threshold, then the server can be marked 760 as a zombie server. If the percentage is not greater than the predetermined threshold, then it can be determined 738 whether the time interval X has elapsed.

[0096] As used in FIG. 7, a predetermined threshold can include any percentage that is selected before the determination is made. For example, an administrator can set the predetermined percentage. In some examples, the predetermined percentage can be 90%, 95%, or 99%, among other percentages.

[0097] Bucketizing the layer 7 protocols can include bucketizing the packets associated with the layer 7 protocols. Calculating the percentage 750 can include determining a percentage of the received packets that are associated with utility software.

[0098] In some embodiments, the computing device may receive network traffic for a plurality of servers each comprising processing resources. The computing device can be a sniffer. The computing device may be configured to distribute network traffic to the plurality of servers. For example, the computing device can be a switch or a router. The computing device may be referred to as a first computing device.

[0099] The first computing device can be configured to identify application traffic, from the network traffic, to the plurality of servers by identifying the layer 7 protocols.

The identified application traffic can be associated with each of the plurality of servers. The application traffic can be network traffic to/from productivity software being executed in the plurality of servers.

[0100] In some embodiments, a second computing device can be configured to access the application traffic from the first computing device. The second computing device can be a management unit (e.g., the management unit 214 in FIG. 2). The second computing device can identify as zombie servers a portion of the plurality of servers that did not receive application traffic based on the identified application traffic that is associated with the plurality of servers. That is, if a server does not receive application traffic associated with productivity software being executed on the server, then the server can be marked as a zombie server.

[0101] The second computing device can further perform a management action on the zombie servers based on the identification of the portion of the plurality of servers as zombie servers.

[0102] Identifying the application traffic can further comprise identifying network traffic received on a predetermined port. For example, the network traffic received on a predetermined port can be classified as application traffic.

[0103] The application traffic may further be identified based on a header of a packet received at the first computing device. That is, each packet in the network traffic may be reviewed by reviewing the header information. The header information may have data that identifies the packet as application traffic.

[0104] The application traffic may also be identified based on a payload of a packet. The payload of the packet may be the body of the packet. The body of the packet may include information, instructions, and/or data that may be characteristic of information, instructions, and/or data received by productivity software being executed on a server. As such, the payload of a packet may be compared to predefined payloads that define application traffic.

[0105] In some embodiments, the payload may be encrypted or portions of the payload may be encrypted. As such, at least one of a header and/or a payload of a packet can be decrypted to identify application traffic. For example, the application traffic can be identified based on at least one of the header and the payload of the packet being identified as corresponding to application traffic.

[0106] FIG. 8 is a flowchart for identifying a zombie server and taking an action based on the identification according to one embodiment. FIG. 8 includes a server with low utilization 870, a zombie server 872, and a rest in peace (RIP) server 874.

[0107] The low utilization server 870 can represent a server that either has a low utilization of productivity software, utility software, and/or total server utilization. In some embodiments, the different approaches described in FIGS. 4-7 can be performed on low utilization servers (e.g., low utilization server 870) to determine whether the low utilization servers are zombie servers.

[0108] For example, any one or more of the embodiments described in FIGS. 4-7 can be applied to the low utilization server 870 to determine whether the low utilization server 870 is the zombie server 872. The embodiments described in FIGS. 4-7 can be applied at time t1. That is, the time t1 is the time interval X as described above.

[0109] The management unit and/or system administrator can take a number of actions based upon the identification of the zombie server 872. For example, the management unit can quarantine the zombie server 872 for a duration of time to better evaluate the zombie server 872. The system administrator can provide a notice to the owner of the hardware or application that the server is not used and may be decommissioned or reassigned.

[0110] The management unit can retest the zombie server 872 at different time periods to determine whether the zombie server 872 is also marked as a zombie server at different times. The management unit can apply different embodiments from the embodiments described in FIGS. 4-7 to determine whether the different embodiments mark the zombie server 872 as a zombie server. For example, the embodiment described in FIG. 4 can be applied to the low utilization server 870 to determine whether the low utilization server 870 is the zombie server 872. If the embodiment described in FIG. 4 marks the low utilization server 870 as the zombie server 872, then a management unit can apply the embodiment described in FIG. 7 to determine whether the embodiment described in FIG. 7 also marks the now classified zombie server 872 as a zombie server. This process can be repeated a number of times using the embodiments described in FIGS. 4-7.

[0111] In some embodiments, the management unit can disable access to the zombie server 872 to incentivize a user to request access to the zombie server 872. That is, if a user is utilizing the zombie server 872, then the low utilization server 870

may have incorrectly been marked as the zombie server 872. If the user requests access to the zombie server 872, then the zombie server 872 can be classified as a non-zombie server, the low utilization server 870, and/or a different classification altogether.

[0112] If access to the zombie server 872 is not requested by time t2 after access to the zombie server 872 has been disabled, then a terminating action may be taken. A terminating action can include deactivating the zombie server 872 and/or repurposing the zombie server 872, among other possible actions that can be taken to conserve the resources expended on the zombie server 872.

[0113] Additional Examples

[0114] The following examples may include subject matter such as a method, means for performing acts of the method, at least one machine-readable medium including instructions that, when performed by a machine cause the machine to perform acts of the method, or of an apparatus or system, according to embodiments and examples described herein.

[0115] Example 1 is a system for detecting a zombie server. The system includes a server with processing resources and a computing device. The computing device is designed to receive network traffic, calculate a percentage of the network traffic as being utility software layer 7 protocols every first time interval, mark the server as a zombie server based on the percentage every second time interval, and distribute the network traffic to the server.

[0116] In Example 2, the subject matter of Example 1 or any of the Examples described herein may further include a system wherein the computing device is a switch.

[0117] In Example 3, the subject matter of Example 2 or any of the Examples described herein may further include a system wherein the switch is an out-of-band switch.

[0118] In Example 4, the subject matter of Example 1 or any of the Examples described herein may further include a system wherein the computing device is a router.

[0119] In Example 5, the subject matter of Example 1 or any of the Examples described herein may further include a system wherein the utility software layer 7 protocols are received on a predetermined port.

[0120] In Example 6, the subject matter of Example 1 or any of the Examples described herein may further include a system wherein the utility software layer 7 protocols are identified based on a header of a variety of packets including the network traffic.

[0121] In Example 7, the subject matter of Example 1 or any of the Examples described herein may further include a system wherein the utility software layer 7 protocols are identified based on a payload of a variety of packets including the network traffic.

[0122] In Example 8, the subject matter of Example 1 or any of the Examples described herein may further include a system of Example 1 wherein the computing device is further designed to decrypt at least one header and payload of a packet, and identify the software layer 7 protocols based on one of the header and the payload of the packet.

[0123] Example 9 is a method for detecting a zombie server. The method includes receiving, at a server, network traffic, calculating a percentage of the network traffic as being productivity software layer 7 protocols every first time interval, marking the server as a zombie server based on the percentage every second time interval, and processing the network traffic at the server to perform a number of actions by the productivity software.

[0124] In Example 10, the subject matter of Example 9 or any of the Examples described herein may further include the method wherein calculating the percentage of the network traffic and the marking of the server as the zombie server occurs at a management unit.

[0125] In Example 11, the subject matter of Example 10 or any of the Examples described herein may further include the method wherein calculating the percentage also includes determining that a portion of the variety of packets comprising the network traffic is associated with productivity software layer 7 protocols.

[0126] In Example 12, the subject matter of Example 11 or any of the Examples described herein may further include the method wherein calculating the percentage also includes dividing the portion of the variety of packets by the variety of packets.

[0127] In Example 13, the subject matter of Example 12 or any of the Examples described herein may further include the method wherein determining that the portion of the variety of packets is associated with productivity software layer 7

protocols which include identifying the layer 7 protocols from the network traffic, and bucketizing the layer 7 protocols.

[0128] In Example 14, the subject matter of Example 13 or any of the Examples described herein may further include the method wherein bucketizing the layer 7 protocols also includes identifying the portion of the variety of packets as corresponding to productivity software layer 7 protocols where the variety of packets comprise the network traffic.

[0129] Example 15 is a computer-readable medium having instructions stored therein. The instructions include receiving, at a second computing device, network traffic sent to a first computing device. The first computing device is designed to distribute the network traffic to a plurality of servers. The instructions also include calculating a percentage of a portion of the network traffic as being utility software layer 7 protocols every first time interval, for each of the variety of servers. The portion of the network traffic is network traffic to a respective server. The instructions further include marking each of the variety of servers as zombie servers based on a respective percentage every second time interval, and finally, forwarding the network traffic to the first computing device.

[0130] In Example 16, the subject matter of Example 15 or any of the Examples described herein may further include a computer-readable medium where calculating the percentage of the portion of the network traffic is performed at a third computing device.

[0131] In Example 17, the subject matter of Example 16 or any of the Examples described herein may further include a computer-readable medium where the third computing device is a management unit.

[0132] In Example 18, the subject matter of Example 15 or any of the Examples described herein may further include a computer-readable medium where calculating the percentage of the portion of the network traffic includes forwarding information regarding layer 7 protocols from the portion of the network traffic and utility software layer 7 protocols from the portion of the network traffic to the third computing device.

[0133] In Example 19, the subject matter of Example 15 or any of the Examples described herein may further include a computer-readable medium where calculating the percentage of the portion of the network traffic includes dividing the utility software layer 7 protocols by the layer 7 protocols.

[0134] Example 20 is a method which includes receiving network traffic, calculating a percentage of the network traffic as being utility software layer 7 protocols every first time interval, marking the server as a zombie server based on the percentage every second time interval, and distributing the network traffic to the server.

[0135] In Example 21, the subject matter of Example 20 or any of the Examples described herein may further include the method where the computing device is a switch.

[0136] In Example 22, the subject matter of Example 21 or any of the Examples described herein may further include the method where the switch is an out-of-band switch.

[0137] In Example 23, the subject matter of Example 20 or any of the Examples described herein may further include the method where the computing device is a router.

[0138] In Example 24, the subject matter of Example 20 or any of the Examples described herein may further include the method where the utility software layer 7 protocols are received on a predetermined port.

[0139] In Example 25, the subject matter of Example 20 or any of the Examples described herein may further include the method where the utility software layer 7 protocols are identified based on a header of a variety of packets including the network traffic.

[0140] In Example 26, the subject matter of Example 20 or any of the Examples described herein may further include the method where the utility software layer 7 protocols are identified based on a payload of a variety of packets including the network traffic.

[0141] In Example 27, the subject matter of Example 20 or any of the Examples described herein may further include the method which includes decrypting one of a header and a payload of a packet, and identifying the software layer 7 protocols based on one of the header and the payload of the packet.

[0142] Example 28 is a method which includes receiving, at a second computing device, network traffic sent to a first computing device. The first computing device is designed to distribute the network traffic to a variety of servers. The method also includes calculating a percentage of a portion of the network traffic as being utility software layer 7 protocols every first time interval, for each of the variety of servers,

where the portion of the network traffic is network traffic to a respective server. The method further includes marking each of the variety of servers as zombie servers based on a respective percentage every second time interval, and finally forwarding the network traffic to the first computing device.

[0143] In Example 29, the subject matter of Example 28 or any of the Examples described herein may further include the method where calculating the percentage of the portion of the network traffic is performed at a third computing device.

[0144] In Example 30, the subject matter of Example 29 or any of the Examples described herein may further include the method where the third computing device is a management unit.

[0145] In Example 31, the subject matter of Example 28 or any of the Examples described herein may further include the method where calculating the percentage of the portion of the network traffic includes forwarding information regarding layer 7 protocols from the portion of the network traffic and utility software layer 7 protocols from the portion of the network traffic to the third computing device.

[0146] In Example 32, the subject matter of Example 28 or any of the Examples described herein may further include the method where calculating the percentage of the portion of the network traffic includes dividing the utility software layer 7 protocols by the layer 7 protocols.

[0147] Example 33 is at least one computer-readable storage medium having stored thereon computer-readable instructions, which, when executed, implement a method as identified in any of Examples 19-14 and Examples 20-32.

[0148] Example 34 is an apparatus which includes a procedure to perform a method as identified in any of Examples 19-14 and Examples 20-32.

[0149] While various aspects and embodiments have been disclosed herein, other aspects and embodiments will be apparent to those skilled in the art. The various aspects and embodiments disclosed herein are for purposes of illustration and are not intended to be limiting, with the true scope and spirit being indicated by the following claims.

Claims

1. A system for detecting a zombie server, the system comprising:
a server comprising processing resources;
a computing device configured to:
 receive network traffic;
 calculate a percentage of the network traffic as being utility software layer 7 protocols every first time interval;
 mark the server as a zombie server based on the percentage every second time interval; and
 distribute the network traffic to the server.
2. The system of claim 1, wherein the computing device is a switch.
3. The system of claim 2, wherein the switch is an out-of-band switch.
4. The system of claim 1, wherein the computing device is a router.
5. The system as in claims 1, 2, or 4, wherein the utility software layer 7 protocols are received at least on a predetermined port.
6. The system as in claims 1, 2, or 4, wherein the utility software layer 7 protocols are identified based on a header of a plurality of packets comprising the network traffic.
7. The system as in claims 1, 2, or 4, wherein the utility software layer 7 protocols are identified based on a payload of a plurality of packets comprising the network traffic.
8. The system as in claims 1, 2, or 4, wherein the computing device is further configured to:
 decrypt at least one of a header and a payload of a packet; and
 identify the software layer 7 protocols based on at least one of the header and the payload of the packet.
9. A method for detecting a zombie server, the method comprising:
 receiving, at a server, network traffic;
 calculating a percentage of the network traffic as being productivity software layer 7 protocols every first time interval;
 marking the server as a zombie server based on the percentage every second time interval; and
 processing the network traffic at the server to perform a number of actions by the productivity software.

10. The method of claim 9, wherein calculating the percentage of the network traffic and the marking of the server as the zombie server occurs at a management unit.
11. The method of claim 10, wherein calculating the percentage further comprises determining that a portion of the plurality of packets comprising the network traffic is associated with productivity software layer 7 protocols.
12. The method of claim 11, wherein calculating the percentage further comprises dividing the portion of the plurality of packets by the plurality of packets.
13. The method as in claims 9, 10, 11, or 12, wherein determining that the portion of the plurality of packets is associated with productivity software layer 7 protocols includes:
 - identifying the layer 7 protocols from the network traffic; and
 - bucketizing the layer 7 protocols.
14. The method of claim 13, wherein bucketizing the layer 7 protocols includes identifying the portion of the plurality of packets as corresponding to productivity software layer 7 protocols, wherein the plurality of packets comprise the network traffic.
15. A computer-readable medium having instructions stored therein, the instructions comprising:
 - receiving, at a second computing device, network traffic sent to a first computing device, wherein the first computing device is configured to distribute the network traffic to a plurality of servers;
 - calculating a percentage of a portion of the network traffic as being utility software layer 7 protocols every first time interval, for each of the plurality of servers, wherein the portion of the network traffic is network traffic to a respective server;
 - marking each of the plurality of servers as zombie servers based on a respective percentage every second time interval; and
 - forwarding the network traffic to the first computing device.
16. The computer-readable medium of claim 15, wherein calculating the percentage of the portion of the network traffic is performed at a third computing device.
17. The computer-readable medium of claim 16, wherein the third computing device is a management unit.

18. The computer-readable medium as in claims 15, 16, or 17, wherein calculating the percentage of the portion of the network traffic includes forwarding information regarding layer 7 protocols from the portion of the network traffic and utility software layer 7 protocols from the portion of the network traffic to the third computing device.

19. The computer-readable medium as in claims 15, 16, or 17, wherein calculating the percentage of the portion of the network traffic includes dividing the utility software layer 7 protocols by the layer 7 protocols.

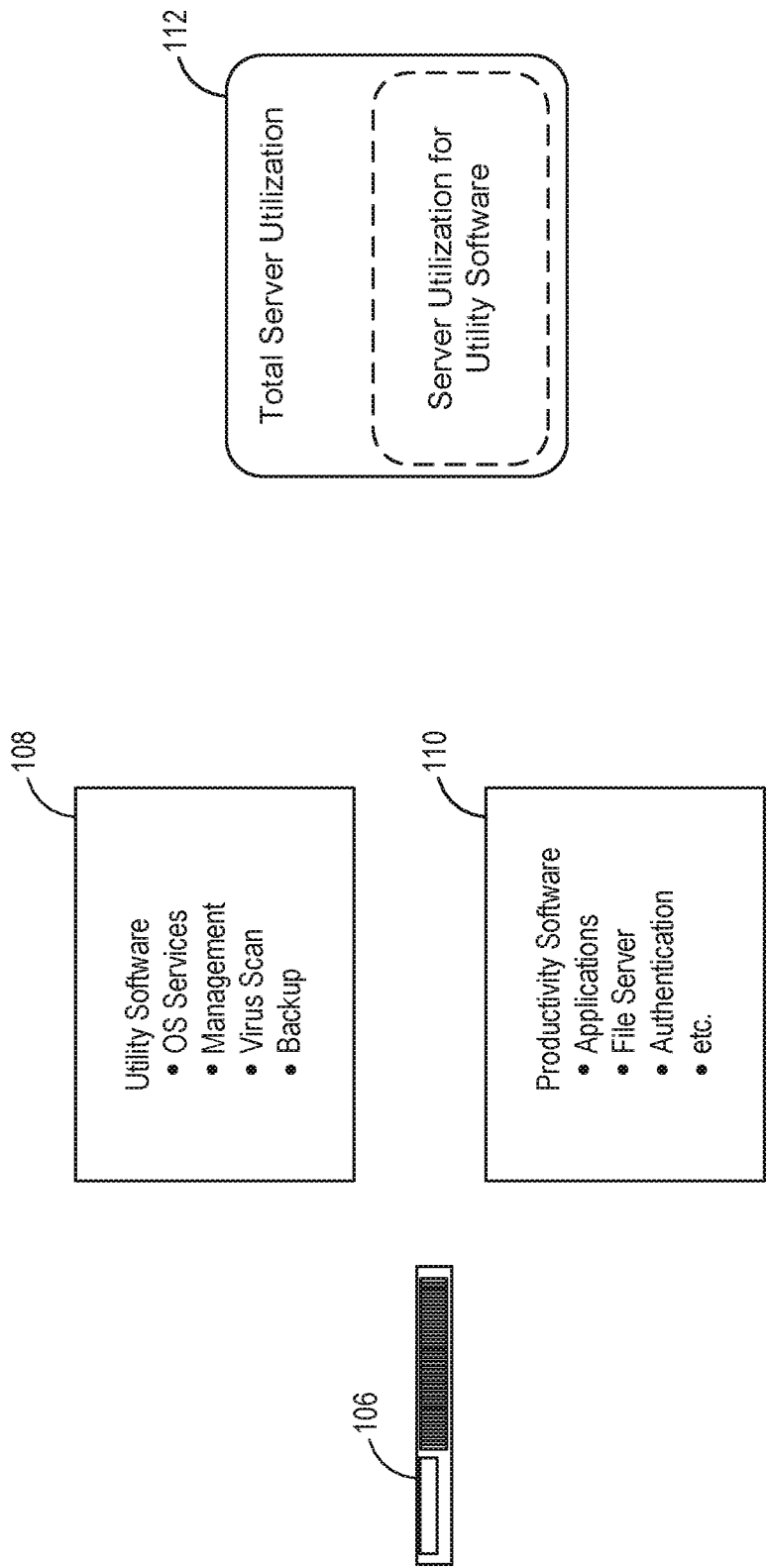


FIG. 1

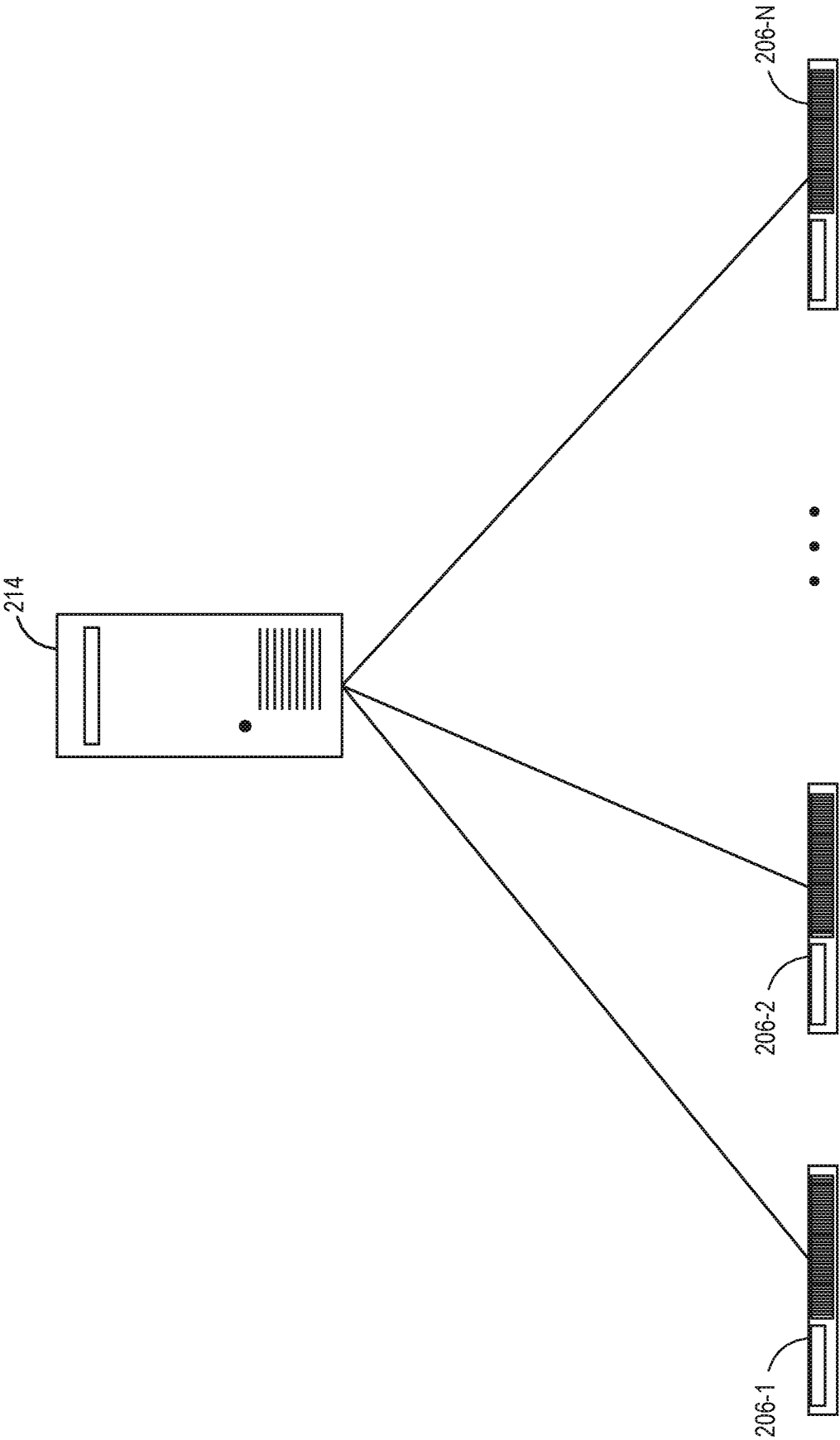


FIG. 2

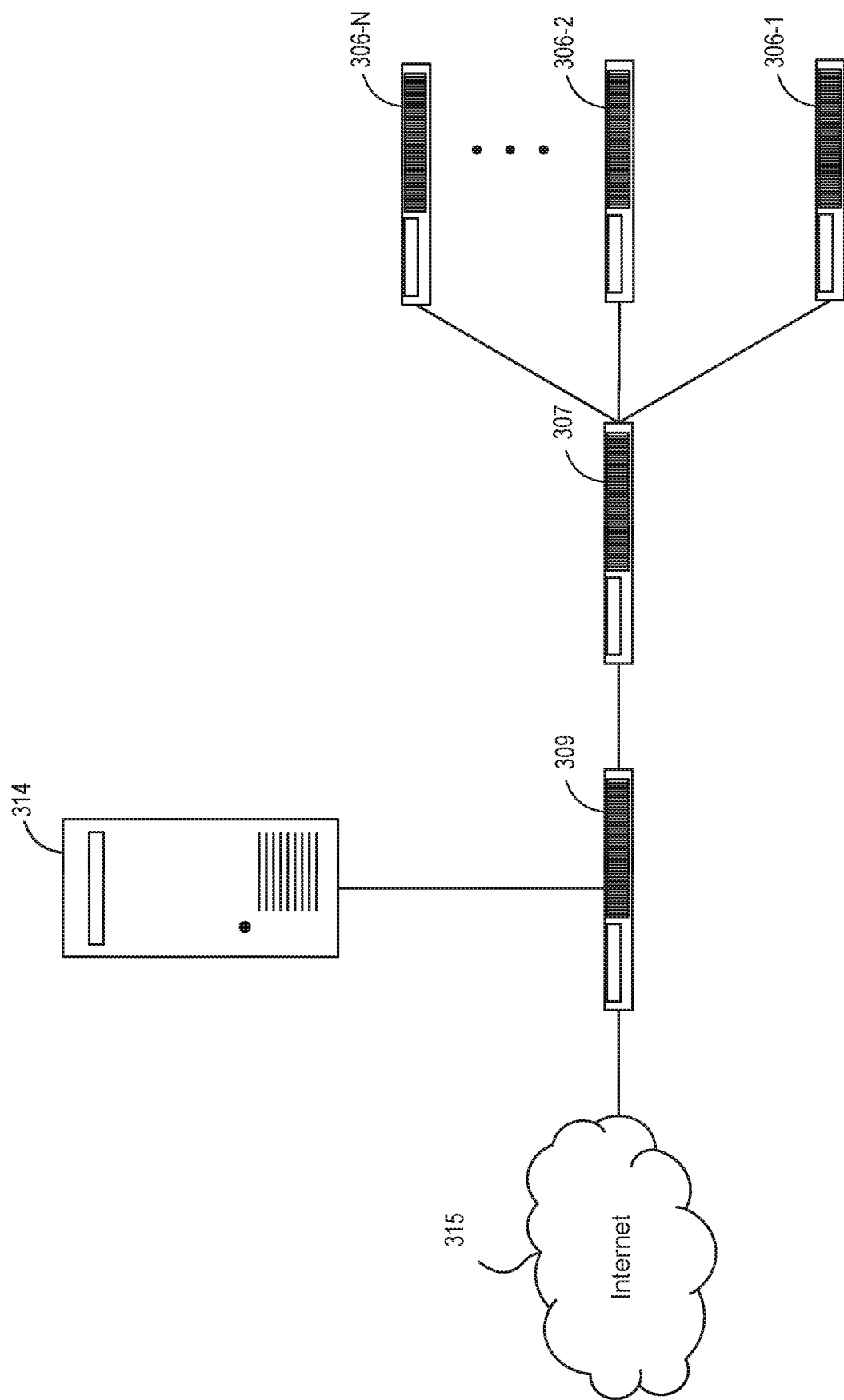


FIG. 3

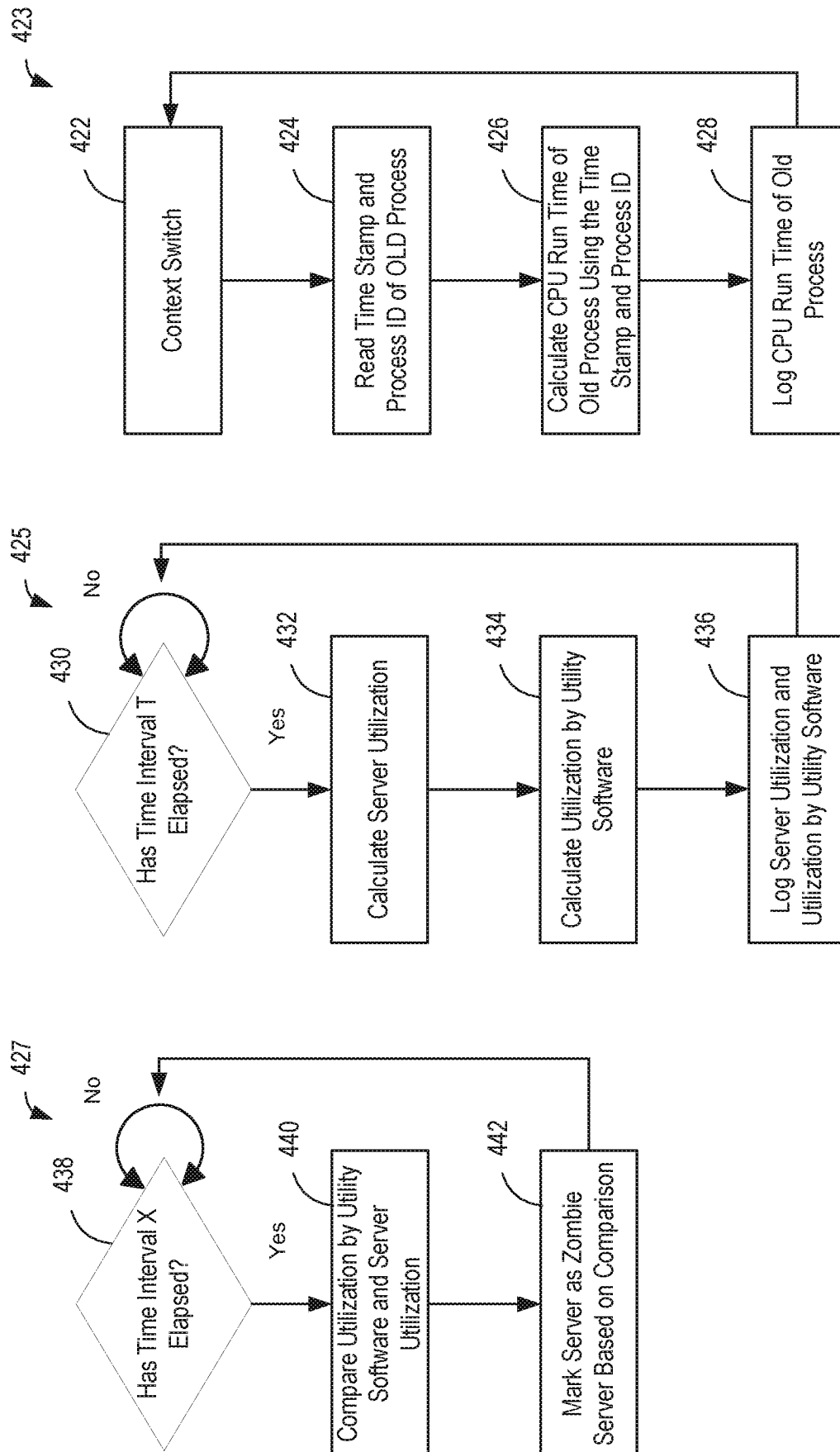


FIG. 4

5/8

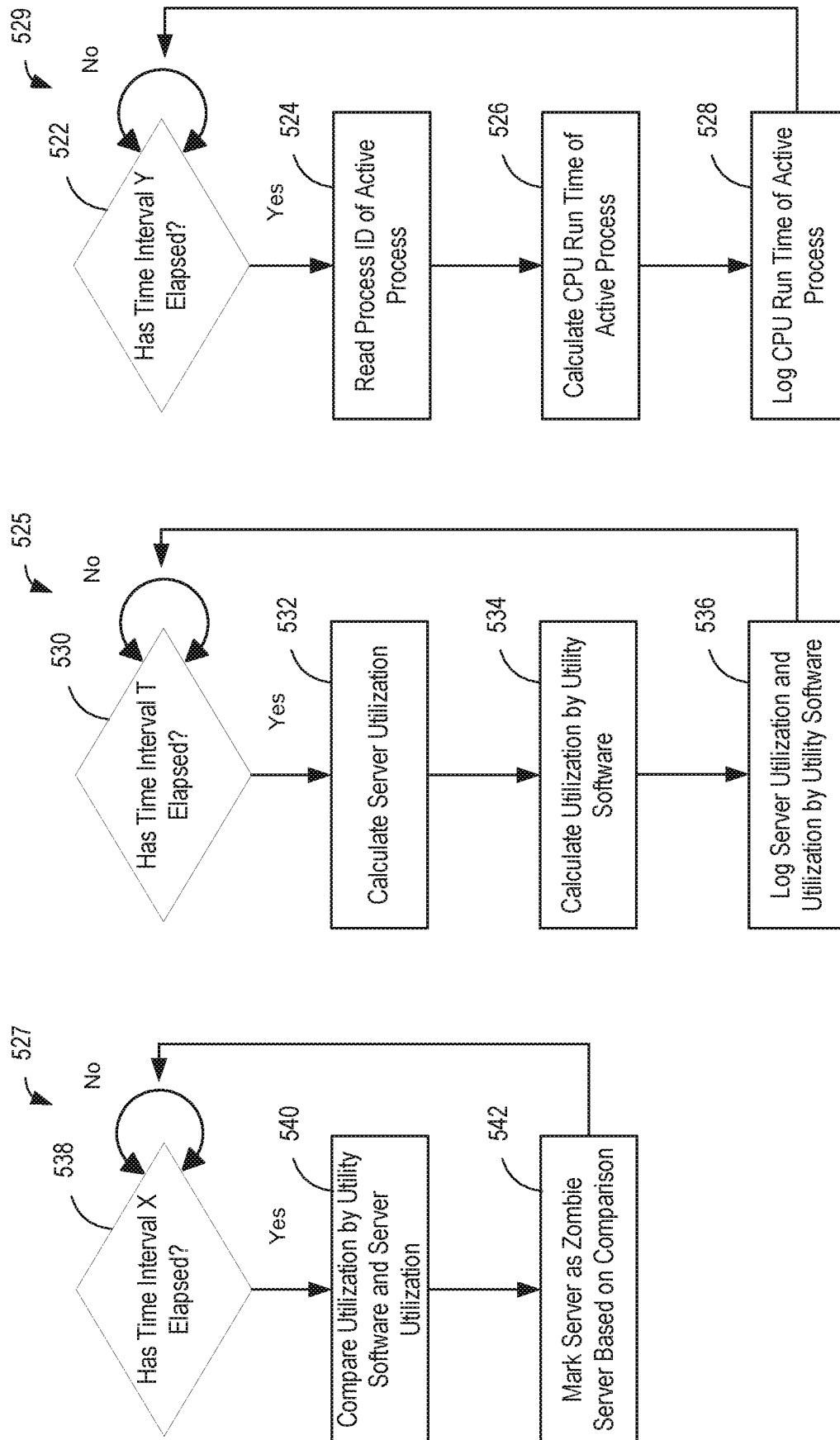


FIG. 5

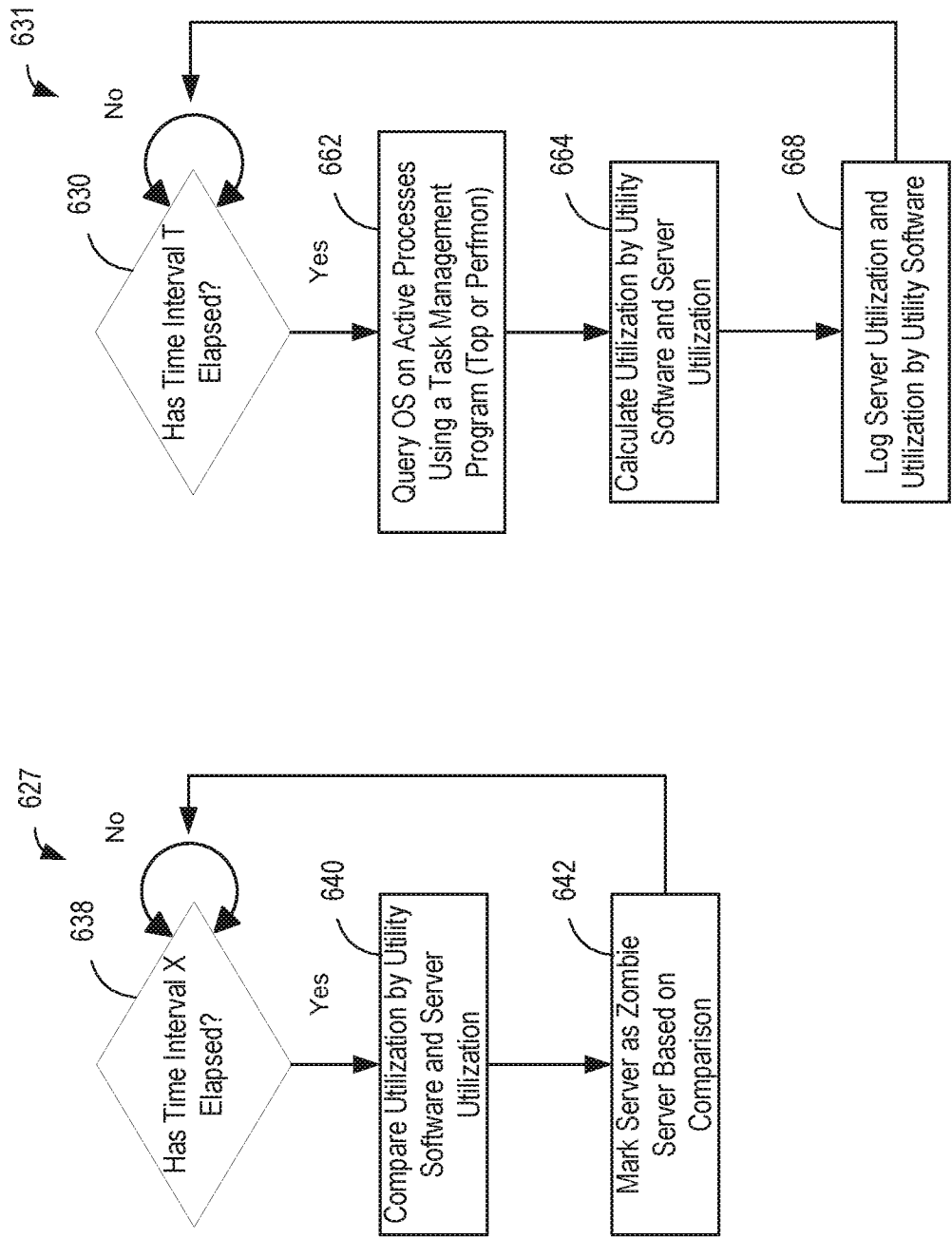


FIG. 6

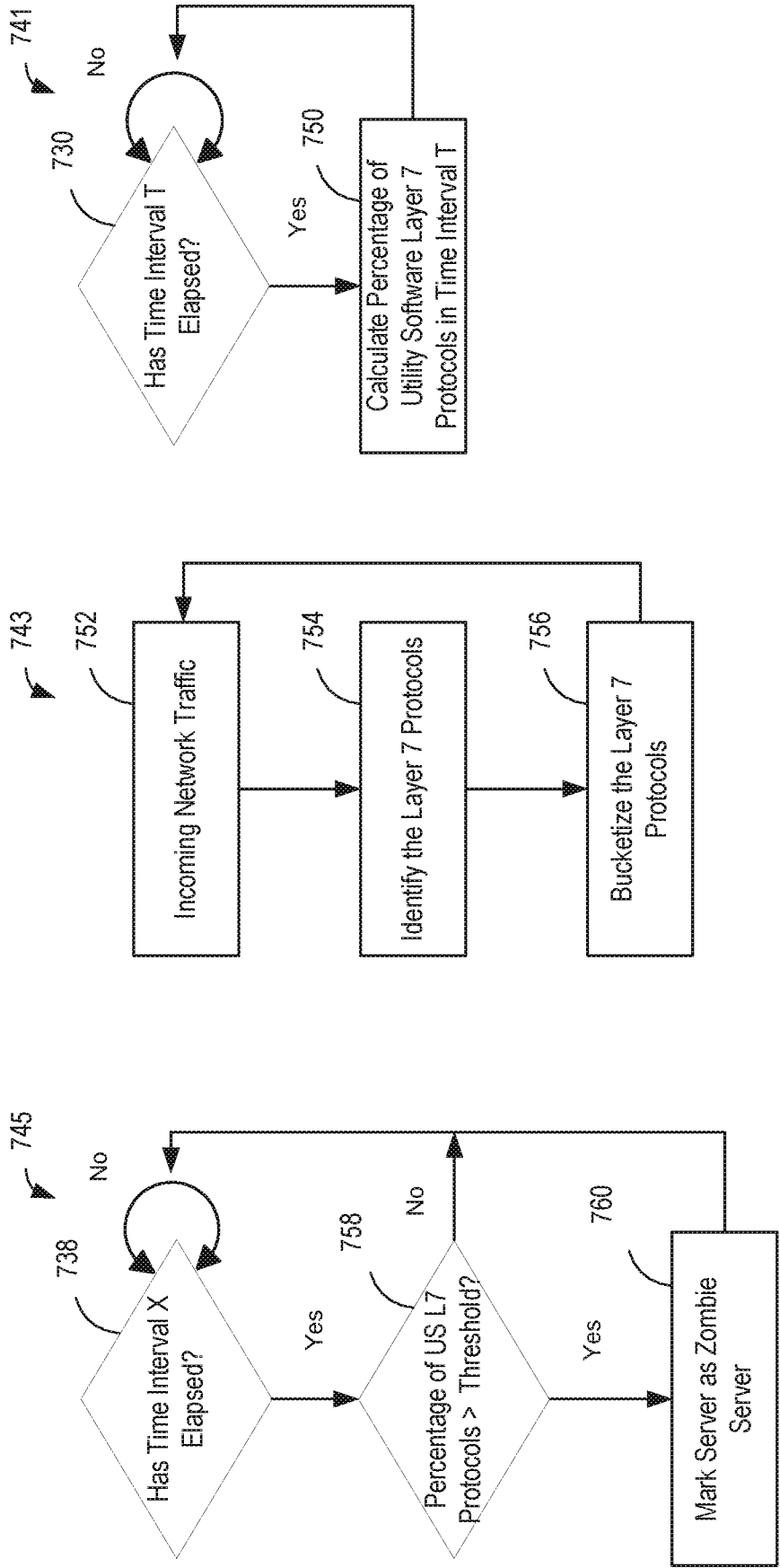


FIG. 7

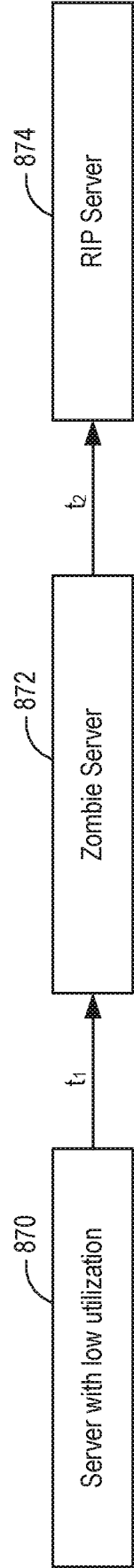


FIG. 8

A. CLASSIFICATION OF SUBJECT MATTER**H04L 29/06(2006.01)i, H04L 29/08(2006.01)i**

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 29/06; G06F 11/07; G06F 15/16; G06F 17/00; G06F 9/46; G06F 11/36; G08B 23/00; H04L 29/08

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

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

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

eKOMPASS(KIPO internal) & Keywords: zombie, layer7, application, botnet, percentage, productivity, utility

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2015-0193330 A1 (CENTURYLINK INTELLECTUAL PROPERTY LLC) 09 July 2015 See paragraph [0075], claim 1 and figure 1.	1-19
A	US 8468601 B1 (ANDREY V. BAKHMUTOV) 18 June 2013 See column 6, lines 14-59, claims 1, 4, 8, 11 and figure 1.	1-19
A	US 8631244 B1 (JAMES N. POTTS et al.) 14 January 2014 See column 4, lines 1-52, claim 11 and figures 1, 3.	1-19
A	US 2004-0268358 A1 (CHRISTOPHER L. DARLING et al.) 30 December 2004 See paragraphs [0263]-[0264] and figure 22.	1-19
A	US 2009-0287768 A1 (ATSUHIRO TANAKA) 19 November 2009 See claims 13-16 and figure 1.	1-19



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search

22 August 2017 (22.08.2017)

Date of mailing of the international search report

28 August 2017 (28.08.2017)

Name and mailing address of the ISA/KR

International Application Division

Korean Intellectual Property Office

189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea

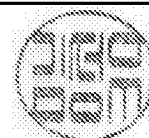


Facsimile No. +82-42-481-8578

Authorized officer

KIM, Seong Woo

Telephone No. +82-42-481-3348



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2017/034137

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2015-0193330 A1	09/07/2015	US 2014-0325481 A1 US 2017-0052886 A1 US 9021439 B2	30/10/2014 23/02/2017 28/04/2015
US 8468601 B1	18/06/2013	EP 2180660 A1 EP 2180660 B1 US 8195750 B1	28/04/2010 05/10/2016 05/06/2012
US 8631244 B1	14/01/2014	None	
US 2004-0268358 A1	30/12/2004	AT 407506 T AU 2004-202389 A1 AU 2004-202403 A1 AU 202389 B2 AU 202403 B2 BR PI0402591 A CA 2470300 A1 CA 2470420 A1 CA 2470420 C CN 1578320 A CN 1578320 B CN 1607781 A CN 1607781 B CO 5590203 A1 EG 23984 A EP 1494421 A1 EP 1494421 B1 EP 1494422 A2 EP 1494422 A3 EP 1494422 B1 HK 1073025 A1 IL 162164 A JP 2005-025756 A JP 2005-027304 A JP 4583091 B2 JP 4942921 B2 KR 10-1109218 B1 KR 10-1169073 B1 KR 10-2005-0002617 A MX PA04006411 A MY 142244 A NO 20042744 A NO 331320 B1 NZ 533148 A RU 2004117220 A RU 2380746 C2 SG 117505 A1 TW 200506646 A TW I356311 B	15/09/2008 20/01/2005 20/01/2005 10/12/2009 17/12/2009 09/02/2005 30/12/2004 30/12/2004 24/01/2012 09/02/2005 20/06/2012 20/04/2005 04/06/2014 30/12/2005 27/02/2008 05/01/2005 03/09/2008 05/01/2005 24/05/2006 03/12/2008 29/05/2009 20/11/2005 27/01/2005 27/01/2005 17/11/2010 30/05/2012 30/01/2012 27/07/2012 07/01/2005 31/03/2005 15/11/2010 03/01/2005 21/11/2011 24/02/2006 10/01/2006 27/01/2010 29/12/2005 16/02/2005 11/01/2012

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2017/034137

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0287768 A1	19/11/2009	US 2004-0264481 A1	30/12/2004
		US 2004-0267920 A1	30/12/2004
		US 2004-0268357 A1	30/12/2004
		US 2005-0055435 A1	10/03/2005
		US 7567504 B2	28/07/2009
		US 7590736 B2	15/09/2009
		US 7606929 B2	20/10/2009
		US 7613822 B2	03/11/2009
		US 7636917 B2	22/12/2009
		ZA 200404375 A	09/12/2004
		ZA 200404375 B	09/12/2004
		EP 2040171 A1	25/03/2009
		EP 2040171 A4	22/09/2010
		JP 2009-007669 A1	10/12/2009
		JP 4900881 B2	21/03/2012
US 8171133 B2		US 8171133 B2	01/05/2012
		WO 2008-007669 A1	17/01/2008