



Industria y Comercio

S U P E R I N T E N D E N C I A

Delegatura de propiedad Industrial

División de Nuevas Creaciones

SOLICITUD

PATENTE DE INVENCION

21. EXPEDIENTE No. 04-61217
54. TÍTULO Balanco de carga de red con
información de Estado de Anfitrión.
51. CLASIFICACIÓN INTERNACIONAL G06f 9/30
71. SOLICITANTE Microsoft -corporation
- DOMICILIO Redmond washington. E.U.A.
74. APODERADO Dario Corderas navas
22. BOGOTÁ, D. C., 28- Julio - 2004

PETITORIO



SUPERINDUSTRIA Y COMERCIO
 Radicación : 04861217 00000000 Folios: 551
 Fecha (IMP): 2004-06-28 16:59:25
 Trámite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
 Dependencia: 0029 DIVISION DE NUEVAS CREACIONES

FORMULARIO UNICO DE SOLICITUD DE PATENTE 2000-3

SOLICITUD DE:

☒ Patente de Invención

☐ Patente de Modelo de Utilidad

SOLICITANTE
(71)

Nombre: MICROSOFT CORPORATION
 Dirección: One Microsoft Way Redmond, Washington 98052, Estados Unidos de America
 Teléfono: 3123600 Fax 3122410
 E-Mail: general@cardenasvcardenas.com
 Lugar de Constitución: Estado de Washington, Estados Unidos de América
 Domicilio: REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA

IDENTIFICACION

C.C. ☐ NIT ☐
 C.E. ☐ Otro ☒
 Número _____

REPRESENTANTE
O APODERADO
(74)

Nombre: DARIO CARDENAS NAVAS
 Dirección: Carrera 7 No 71-52 Torre B Piso 9
 Teléfono: 3123600 Fax 3122410
 E-Mail: general@cardenasvcardenas.com

IDENTIFICACION

C.C. ☒ NIT ☐
 C.E. ☐ Otro ☐
 Número 17.068.629 BTA
 TP 1.250 C.S.J.

INVENTOR(ES)
(72)

Nombre:
 1. Christopher L. Darling
 2. Joseph M. Joy
 3. Sunita Shrivastava
 4. Chittur Subbaraman
 Dirección:
 1. 2721 233rd Place NE, Sammamish, WA 98074, EUA
 2. 17219 NE 32nd Street Redmond, WA 98052, EUA
 3. 14824 NE 76th Ct Redmond, WA 98052, EUA
 4. 5037 326th Place NE Carnation, WA 98014, EUA
 Teléfono: 3123600 Fax 3123600
 E-Mail: general@cardenasvcardenas.com
 Domicilio:
 1. WASHINGTON, ESTADOS UNIDOS DE AMERICA
 2. WASHINGTON, ESTADOS UNIDOS DE AMERICA
 3. WASHINGTON, ESTADOS UNIDOS DE AMERICA
 4. WASHINGTON, ESTADOS UNIDOS DE AMERICA

IDENTIFICACION

C.C. ☐ NIT ☐
 C.E. ☐ Otro ☒
 Número _____

Título(54):

BALANCEO DE CARGA DE RED CON INFORMACION DE ESTADO DE ANFITRION

Prioridad

SI ☒ NO ☐

(33) País de Origen

USA

(32) Fecha

10/610.519

(31) N° de Solicitud

Junio 30, 2003

Para publicar a partir de la fecha de la presente solicitud a los:

6 meses ☐ 12 meses ☐ 18 meses ☐ Otro ☐

06 F 9/30

Comprobante de Pago No.:

04-50,354 / 04-50,351

Fecha

Junio 25, 2004

ANEXOS

- ☒ Comprobante de pago de la tasa de presentación de la solicitud.
- ☐ Comprobante de pago de la tasa por concepto de excedente de palabras en la publicación.
- ☒ Comprobante de pago de la tasa por concepto de reivindicación de prioridad.
- ☒ Poderes, si fuere el caso. (Protocolo No 16.069)
- ☒ Certificado de existencia y representación legal cuando el solicitante sea persona jurídica. Protocolo No 16.069)
- ☒ Copia certificada de la primera solicitud, si se reivindica prioridad.
- ☒ Traducción simple de la primera solicitud, si se reivindica prioridad.
- ☒ Documento de cesión del inventor al solicitante o a su causante.
- ☒ Descripción de la invención.
- ☒ Una o más reivindicaciones.
- ☒ Dibujos y/o planos necesarios.
- ☐ De ser el caso, copia del contrato de acceso.
- ☐ De ser el caso, documento que acredite la licencia o autorización de uso de conocimientos tradicionales de la comunidad de las comunidades indígenas.
- ☐ De ser el caso, certificado de depósito del material biológico.
- ☐ Arte final 12x12 y 6x6 cm.
- ☒ Resumen y extracto para publicación.

10

NOMBRE:

DARIO CARDENAS NAVAS

FIRMA:

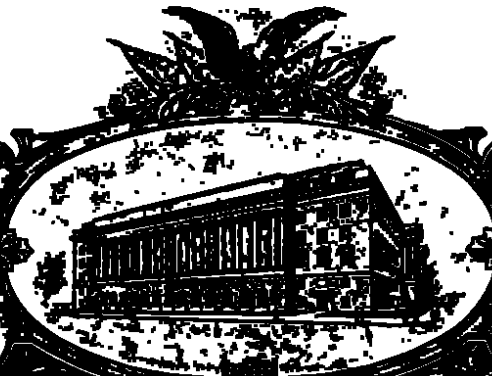


OK

C.C. 17.086.829 BTA

TP. 1.250 C.SJ.

2



THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

May 18, 2004

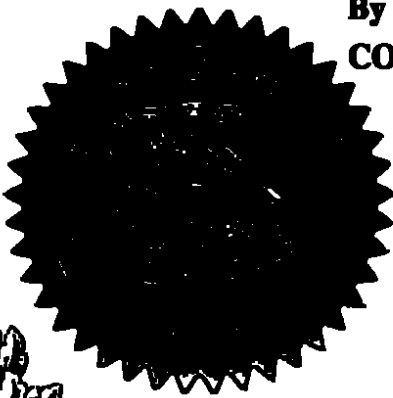
THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THE UNITED STATES PATENT AND TRADEMARK OFFICE OF THOSE PAPERS OF THE BELOW IDENTIFIED PATENT APPLICATION THAT MET THE REQUIREMENTS TO BE GRANTED A FILING DATE UNDER 35 USC 111.

APPLICATION NUMBER: 10/610,519

FILING DATE: June 30, 2003

CD DISK IS THE APPLICATION OF THE ABOVE REFERENCED INFORMATION

**By Authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS**



L. Edelen

**L. EDELEN
Certifying Officer**

EV355227736

B
(3)

4

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

**Network Load Balancing
with Host Status Information**

Inventor(s):
Christopher L. Darling
Joseph M. Joy
Sunita Shrivastava
Chittur Subbaraman

ATTORNEY'S DOCKET NO. MS1-1520US

y³

**Network Load Balancing
with Host Status Information**

TECHNICAL FIELD

This disclosure relates in general to network load balancing and in particular, by way of example but not limitation, to network load balancing with host status information.

BACKGROUND

Communication, and many facets of life that involve communication, has been greatly impacted by the Internet. The Internet enables information to be communicated between two people and/or entities quickly and relatively easily. The Internet includes many network nodes that are linked together such that information may be transferred between and among them. Some network nodes may be routers that propagate a packet from one link to another, may be individual client computers, may be personal networks for different entities (e.g., intranets for businesses), and so forth.

For this personal network case, as well as others, packets arriving at an Internet node or nodes are distributed to other nodes within the personal network. Such a personal network may be formed, for example, from a set of servers that can each work on packets that arrive at the personal network. A business, a university, a government office, etc. may receive many packets in a short timeframe at its personal network. In order to respond in a timely manner and to reduce the likelihood of rejection or loss of arriving packets, the personal network

ju

1 may rely on multiple servers that can each work on the arriving packets
2 simultaneously.

3 The arriving packets are often inquiries pertaining to certain information,
4 such as a document, a catalog item, a web page, and so forth. The arriving packets
5 can also pertain to an economic transaction between a customer and a merchant.
6 Other purposes for the packets of a packet-based communication are possible.
7 Regardless, the arriving packets are distributed among different servers of a set of
8 servers to accommodate a rapid arrival of the packets and/or complex
9 communication exchanges.

10 The distribution of arriving packets among different servers of a set of
11 servers is often termed network load balancing. In other words, a load balancing
12 operation may be performed on packets as they arrive at a node or nodes of the
13 Internet when the node or nodes constitute a personal network and/or when they
14 connect the personal network to the Internet.

15 Such a load balancing operation is accomplished using dedicated hardware
16 that fronts the personal network at the node or nodes that connect the personal
17 network to the Internet and/or that provide a presence for the personal network on
18 the Internet. The physical hardware that performs the load balancing operation is
19 usually duplicated in its entirety to realize redundancy and improve availability of
20 the load balancing operation. To increase capacity for load balancing operations,
21 more-powerful hardware that replicates the entirety of the previous load balancing
22 hardware, and thus the operational capability thereof, is substituted for the
23 previous load balancing hardware. Such scaling up of the load balancing
24 operational capabilities is therefore confined to increasing the power of the
25 hardware via substitution thereof.

45

1 To implement a load balancing operation, the hardware usually performs a
2 round robin distribution of arriving connection requests. In other words, arriving
3 connection requests are distributed to servers of a set of servers in a linear,
4 repeating manner with a single connection request being distributed to each server.
5 This round-robin load balancing distribution of connections is typically utilized
6 irrespective of the condition of the personal network or the nature of an arriving
7 connection request. If a load balancing operation does extend beyond a round
8 robin distribution, these other factors are only considered to the extent that they
9 may be inferred from network traffic and/or from a congestion level of the
10 personal network.

11 Accordingly, there is a need for schemes and/or techniques that improve
12 network load balancing and/or the options associated therewith.

13
14 **SUMMARY**

15 In a first exemplary media implementation, one or more processor-
16 accessible media include processor-executable instructions that, when executed,
17 direct a system to perform actions that include: accumulating host status
18 information at multiple hosts; and sending the accumulated host status information
19 from the multiple hosts. In a second exemplary media implementation, one or
20 more processor-accessible media include processor-executable instructions that,
21 when executed, direct a system to perform actions that include: receiving host
22 status information from multiple hosts; and making load balancing decisions
23 responsive to the received host status information. In a third exemplary media
24 implementation, one or more processor-accessible media include processor-
25 executable instructions that, when executed, direct a system to perform actions

1 that include: determining health and load information on a per application basis;
2 and selecting an application from among multiple applications responsive to the
3 health and load information.

4 In a fourth exemplary media implementation, one or more processor-
5 accessible media include processor-executable instructions that, when executed,
6 direct a system to perform actions that include: analyzing health and/or load
7 information for multiple application endpoints; and ascertaining a token allotment
8 for the multiple application endpoints responsive to the analyzing.

9 In a fifth exemplary media implementation, one or more processor-
10 accessible media include processor-executable instructions that, when executed,
11 enable a system to implement a message protocol between at least one host and
12 one or more load balancing units, the message protocol usable for communicating
13 health and/or load information between the at least one host and the one or more
14 load balancing units.

15 In an exemplary system implementation, a system includes: at least one
16 device that is hosting one or more applications, the at least one device including a
17 health and load table that includes multiple entries, each entry of the multiple
18 entries associated with an application of the one or more applications; each entry
19 of the multiple entries including: an application identifier for a particular
20 application of the one or more applications; information characterizing at least one
21 status of the particular application; and at least one load balancing directive
22 regarding the particular application.

23 Other method, system, approach, apparatus, application programming
24 interface (API), device, media, procedure, arrangement, etc. implementations are
25 described herein.

BRIEF DESCRIPTION OF THE DRAWINGS

The same numbers are used throughout the drawings to reference like and/or corresponding aspects, features, and components.

FIG. 1 is an exemplary network load balancing paradigm that illustrates a load balancing infrastructure and multiple hosts.

FIG. 2 is an exemplary network load balancing paradigm that illustrates multiple load balancing units and multiple hosts.

FIG. 3 illustrates an exemplary load balancing unit having separated functionality and an exemplary host.

FIG. 4 illustrates exemplary network load balancing infrastructure having separated classifying and forwarding functionality.

FIG. 5 is a flow diagram that illustrates an exemplary method for scaling out network load balancing infrastructure into different configurations.

FIG. 6 illustrates a first exemplary network load balancing infrastructure configuration from a device perspective.

FIG. 7 illustrates a second exemplary network load balancing infrastructure configuration from a device perspective.

FIGS. 8A and 8B illustrate first and second exemplary network load balancing infrastructure configurations from a component perspective.

FIGS. 9A and 9B illustrate first and second exemplary network load balancing infrastructure configurations from a resource perspective.

FIG. 10 illustrates an exemplary network load balancing approach that involves host status information.

1 FIG 11 is a flow diagram that illustrates an exemplary method for network
2 load balancing that involves host status information.

3 FIG 12 illustrates an exemplary network load balancing approach that
4 involves health and load information.

5 FIG 13A is an exemplary health and load table as illustrated in FIG 12.

6 FIG 13B is an exemplary consolidated health and load cache as illustrated
7 in FIG 12.

8 FIG 14 is a flow diagram that illustrates an exemplary method for network
9 load balancing that involves health and load information.

10 FIG 15 illustrates an exemplary message protocol for communications
11 between the hosts and load balancing units that are illustrated in FIG 12.

12 FIG 16 illustrates an exemplary message transmission scheme for
13 communications between the hosts and load balancing units that are illustrated in
14 FIG 12.

15 FIGS. 17A and 17B illustrate exemplary health and load information proxy
16 storage scenarios for health and load tables of FIG 13A and for consolidated
17 health and load caches of FIG 13B, respectively.

18 FIG 18 illustrates an exemplary target host allotment procedure that utilizes
19 health and load information.

20 FIG 19 illustrates an exemplary network load balancing approach that
21 involves session information.

22 FIG 20 illustrates an exemplary network load balancing approach that
23 involves communicating session information using notifications and messages.
24
25

1 FIG. 21 is a flow diagram that illustrates an exemplary method for network
2 load balancing that involves communicating session information using
3 notifications and messages.

4 FIG. 22 illustrates an exemplary approach to managing session information
5 at multiple load balancing units.

6 FIG. 23A is an exemplary session table as illustrated in FIG. 20.

7 FIG. 23B is an exemplary distributed atom manager (DAM) table (DAMT)
8 as illustrated in FIG. 22.

9 FIG. 24 is a flow diagram that illustrates an exemplary method for
10 managing session information at multiple load balancing units.

11 FIG. 25 illustrates exemplary network load balancing infrastructure having
12 request routing functionality.

13 FIG. 26 is a flow diagram that illustrates an exemplary method for routing
14 incoming packets with regard to (i) session information and (ii) health and load
15 information.

16 FIG. 27 illustrates an exemplary traffic routing flow in the absence of
17 failures.

18 FIG. 28 illustrates an exemplary traffic routing flow in the presence of
19 failure(s).

20 FIG. 29 illustrates additional exemplary failover procedures for high
21 availability of network load balancing infrastructure.

22 FIG. 30 illustrates an exemplary operational implementation of traffic
23 routing interaction with health and load information.

24 FIG. 31 illustrates exemplary high availability mechanisms for network
25 load balancing infrastructure.

1 FIG. 32 illustrates an exemplary approach to application-level network load
2 balancing with connection migration.

3 FIG. 33 is a flow diagram that illustrates an exemplary method for
4 migrating a connection from a first device to a second device.

5 FIG. 34 illustrates an exemplary approach to connection migration from the
6 perspective of an originating device.

7 FIG. 35 illustrates an exemplary approach to connection migration from the
8 perspective of a targeted device.

9 FIG. 36 illustrates an exemplary approach to an offloading procedure for a
10 connection migration.

11 FIG. 37 illustrates an exemplary approach to an uploading procedure for a
12 connection migration.

13 FIG. 38 illustrates an exemplary approach to packet tunneling between a
14 forwarder and a host.

15 FIG. 39 is a flow diagram that illustrates an exemplary method for packet
16 tunneling between a first device and a second device.

17 FIG. 40 illustrates an exemplary computing (or general device) operating
18 environment that is capable of (wholly or partially) implementing at least one
19 aspect of network load balancing as described herein.

20 21 **DETAILED DESCRIPTION**

22 **Exemplary Network Load Balancing Paradigms**

23 This section describes exemplary paradigms for network load balancing
24 and is used to provide foundations, environments, contexts, etc. for the
25

1 descriptions in the following sections. This section primarily references FIGS. 1-
2 3.

3 FIG. 1 is an exemplary network load balancing paradigm 100 that illustrates
4 a load balancing infrastructure 106 and multiple hosts 108. Exemplary network
5 load balancing paradigm 100 includes multiple clients 102(1), 102(2) ... 102(m)
6 and multiple hosts 108(1), 108(2) ... 108(n), as well as network 104 and load
7 balancing infrastructure 106.

8 Each of clients 102 may be any device that is capable of network
9 communication, such as a computer, a mobile station, an entertainment appliance,
10 another network, and so forth. Clients 102 may also relate to a person and/or
11 entity that is operating a client device. In other words, clients 102 may comprise
12 logical clients that are users and/or machines. Network 104 may be formed from
13 one or more networks, such as the Internet, an intranet, a wired or wireless
14 telephone network, and so forth. Additional examples of devices for clients 102
15 and network types/topologies for network 104 are described below with reference
16 to FIG. 40 in the section entitled "Exemplary Operating Environment for
17 Computer or Other Device".

18 Individual clients 102 are capable of communicating with one or more
19 hosts 108, and vice versa, across network 104 via load balancing infrastructure
20 106. Hosts 108 host one or more applications for interaction/communication with
21 clients 102, for use by clients 102, and so forth. Each host 108 may correspond to
22 a server and/or a device, multiple servers and/or multiple devices, part of a server
23 and/or part of a device, some combination thereof, and so forth. Particular
24 implementations for hosts 108 are described further below in the context of
25 different network load balancing situations. (However, back-end support for hosts

1 108 is generally not shown for the sake of clarity.) Furthermore, additional
2 examples of devices for hosts 108 are also described below with reference to FIG.
3 40 in the section entitled "Exemplary Operating Environment for Computer or
4 Other Device".

5 Load balancing infrastructure 106 is reachable or locatable through network
6 104 at one or more virtual internet protocol (IP) addresses. Communications from
7 clients 102 (or other nodes) that are directed to the virtual IP address of load
8 balancing infrastructure 106 are received there and forwarded to a host 108. Load
9 balancing infrastructure 106 is comprised of hardware and/or software
10 components (not explicitly shown in FIG. 1).

11 Although load balancing infrastructure 106 is shown as an integral ellipse,
12 the infrastructure to effectuate load balancing may also be distributed to other
13 aspects of exemplary network load balancing paradigm 100. For example,
14 software component(s) of load balancing infrastructure 106 may be located at one
15 or more of hosts 108 as is described further below. Examples of architectures for
16 load balancing infrastructure 106 are described below with reference to FIG. 40 in
17 the section entitled "Exemplary Operating Environment for Computer or Other
18 Device".

19 As indicated at (1), one or more of hosts 108 may provide host status
20 information from hosts 108 to load balancing infrastructure 106. This host status
21 information may be application specific. Examples of such host status information
22 are described further below and include health and/or load information, session
23 information, etc. for hosts 108. A particular implementation that includes
24 providing health and/or load information from hosts 108 to load balancing
25

fs

1 infrastructure 106 is described below in the section entitled "Exemplary Health
2 and Load Handling".

3 At (2), a request is sent from client 102(1) across network 104 to load
4 balancing infrastructure 106 at the virtual IP address thereof. The content, format,
5 etc. of a request from a client 102 may depend on the application to which the
6 request is directed, and the term "request" may implicitly include a response or
7 responses from host(s) 108, depending on the context. Kinds of client requests
8 include, but are not limited to:

9 1. Hyper text transfer protocol (HTTP) GET requests
10 from a client using a browser program. Depending on the
11 application (and more specifically, on the uniform resource locator
12 (URL) of the requests), it may be better to service the requests by
13 different sets of hosts, and the existence of a client "session" state on
14 the hosts may militate that requests from specific clients be routed to
15 specific hosts. The requests may be over a secure sockets layer
(SSL) (or other encrypted) connection.

16 2. Virtual private network (VPN) connections (e.g., the
17 hosts are a set of VPN servers). In this case, the "request" can be
18 considered to be a layer-2 tunneling protocol (L2TP) or point-to-
19 point tunneling protocol (PPTP) "connection" (the latter is a
20 combination of a transmission control protocol (TCP) control
21 connection and associated generic routing encapsulation (GRE) data
traffic).

22 3. Terminal server connections (e.g., the hosts are a set of
23 terminal servers).

24
25

24

1 4. Proprietary requests in the form of individual TCP
2 connections (one per request) employing a proprietary application-
3 specific protocol.

4 5. Simple object access protocol (SOAP) requests.

5 6. Real-time communication requests involving control
6 information over a TCP connection and latency-sensitive media
7 streaming over real-time protocol (RTP).

8 Thus, requests can take many diverse, application-specific forms. In certain
9 described implementations, load balancing infrastructure 106 may make
10 application-specific forwarding decisions.

11 At (3), load balancing infrastructure 106 forwards the request from 102(1)
12 to host 108(2) (in this example). Load balancing infrastructure 106 may consider
13 one or more of many factors when selecting a host 108 to which the request is to
14 be forwarded, depending on which implementation(s) described herein are being
15 employed. For example, load balancing infrastructure 106 may take into account:
16 the application health and/or load information of each host 108, session
17 information relating to client 102(1) as stored at a host 108, and so forth.

18 FIG 2 is an exemplary network load balancing paradigm 200 that illustrates
19 multiple load balancing units 106 and multiple hosts 108. Specifically, load
20 balancing infrastructure 106 is shown as multiple load balancing units 106(1),
21 106(2) ... 106(u) in exemplary network load balancing paradigm 200.
22 Additionally, two router and/or switches 202(1) and 202(2) are illustrated.

23 Router/switches 202, if present, may be considered as part of or separate
24 from load balancing infrastructure 106 (of FIG. 1). Router/switches 202 are
25 responsible of directing overall requests and individual packets that are received
from network 104 to the shared virtual IP (VIP) address(es) of load balancing units

β

47
1 106. If a first router/switch 202 fails, the second router/switch 202 may takeover
2 for the first. Although two router/switches 202 are illustrated, one or more than
3 two router/switches 202 may alternatively be employed.

4 Router/switches 202 may be ignorant of the load balancing infrastructure or
5 load-balancing aware. If router/switches 202 are not load-balancing aware, one of
6 two exemplary options may be employed: For a first option, one load balancing
7 unit 106 is "assigned" the shared VIP address, and all network traffic is forwarded
8 thereto. This one load balancing unit 106 then evenly redistributes the traffic
9 across the other load balancing units 106. However, there are bottleneck /and
10 failover issues with this first option (which can be mitigated if multiple VIP
11 addresses are shared and are split between multiple load balancing units 106). For
12 a second option, router/switches 202 are "tricked" into directing network traffic to
13 all load balancing units 106, which individually decide what traffic each should
14 accept for load balancing. However, there are inefficient effort duplication and
15 switch performance/compatibility issues with this second option.

16 If, on the other hand, router/switches 202 are load-balancing aware,
17 router/switches 202 can be made to distribute incoming network traffic
18 between/among multiple load balancing units 106 (e.g., in a round-robin fashion).
19 It should be understood that such load-balancing-aware routers/switches 202 are
20 capable of performing load balancing functions at a rudimentary level (e.g., in
21 hardware). For example, load-balancing-aware routers/switches 202 can perform
22 simple IP-address-based session affinity so that all packets from a specific source
23 IP address are directed to a same load balancing unit 106.

24 Each separately-illustrated load balancing unit 106 of load balancing units
25 106 may represent one physical device, multiple physical devices, or part of a

1 single physical device. For example, load balancing unit 106(1) may correspond
2 to one server, two servers, or more. Alternatively, load balancing unit 106(1) and
3 load balancing unit 106(2) may together correspond to a single server. An
4 exemplary load balancing unit 106 is described further below from a functional
5 perspective with reference to FIG. 3.

6 Two exemplary request paths [1] and [2] are illustrated in FIG. 2. For
7 request path [1], client 102(2) transmits a request over network 104 that reaches
8 router/switch 202(1). Router/switch 202(1) directs the packet(s) of the request that
9 originated from client 102(2) to load balancing unit 106(1). Load balancing unit
10 106(1) then forwards the packet(s) of the request to host 108(1) in accordance with
11 some load-balancing functionality (e.g., policy). For request path [2], client
12 102(m) transmits a request over network 104 that reaches router/switch 202(2).
13 Router/switch 202(2) directs the packet(s) of the request that originated from
14 client 102(m) to load balancing unit 106(u). Load balancing unit 106(u) then
15 forwards the packet(s) of the request to host 108(n) in accordance with some load-
16 balancing functionality. Exemplary load-balancing functionality is described
17 further below with reference to FIG. 3.

18 FIG. 3 illustrates an exemplary load balancing unit 106 having separated
19 functionality and an exemplary host 108. Load balancing unit 106 includes seven
20 (7) functional blocks 302-314. These functional blocks of load balancing unit 106
21 may be realized at least partially using software. Host 108 includes one or more
22 applications 316. In a described implementation, load balancing unit 106 includes
23 a forwarder 302, a classifier 304, a request router 306, a session tracker 308, a
24 connection migrator 310, a tunneler 312, and a health and load handler 314.
25

[Handwritten mark]

1 Health and load handler 314 is located partly at hosts 108 and partly on
2 devices of load balancing units 106. Health and load handler 314 monitors the
3 health and/or load (or more generally the status) of hosts 108 so that health and/or
4 load information thereof may be used for the load-balancing functionality (e.g.,
5 when making load-balancing decisions). Exemplary implementations for health
6 and load handler 314 are described further below, particularly in the section
7 entitled "Exemplary Health and Load Handling".

8 Session tracker 308 may also be located partly at hosts 108 and partly on
9 devices of load balancing units 106. Session tracker 308 monitors sessions that
10 are established by clients 102 so that reconnections/continuations of previously-
11 established sessions may be facilitated by the load-balancing functionality. For
12 example, some applications keep application-specific client session data on the
13 hosts (which is also a type of host status information). These applications
14 typically expect that clients use the same host for the duration of any given
15 session. Exemplary types of sessions include: (i) a TCP connection (which is,
16 strictly speaking, a session); (ii) an SSL session; (iii) a secure IP (IPsec) session;
17 (iv) an HTTP cookie-based session; and so forth.

18 Although session tracker 308 is illustrated as a discrete block in load
19 balancing unit 106, session tracking functionality of session tracker 308 may
20 actually be implemented at a global level. In other words, session affinity is
21 supported across multiple load balancing units 106. Session tracker 308 includes a
22 centralized database and/or a distributed database of session information in order
23 to preserve session affinity. Exemplary implementations for session tracker 308,
24 with an emphasis on a distributed database approach, are described further below,
25 particularly in the section entitled "Exemplary Session Tracking".

1 Classifier 304 uses the data acquired and maintained by health and load
2 handler 314 and/or session tracker 308, possibly in conjunction with other factors,
3 to classify incoming requests. In other words, classifier 304 selects a target host
4 108 for each incoming request from a client 102. Forwarder 302 forwards client
5 requests (and/or the packets thereof) in accordance with the targeted host 108 as
6 selected by classifier 304. Forwarder 302 and classifier 304 may operate on a per-
7 packet basis. Exemplary implementations for forwarder 302 and classifier 304 are
8 described further below, particularly in the sections entitled "Exemplary Approach
9 to Flexible Network Load Balancing" and "Exemplary Classifying, Forwarding,
10 and Request Routing".

11 Request router 306, as contrasted with per-packet implementations of
12 forwarder 302 and classifier 304, can act as a proxy for an application running on
13 a host 108. For example, request router 306 may terminate TCP connections,
14 parse (perhaps partially) each logical request from a client 102, and resubmit each
15 logical request to the targeted host 108. Consequently, each logical request from a
16 client 102 may be directed to a different host 108, depending on the decisions
17 made by request router 306. Furthermore, request router 306 may perform pre-
18 processing on a connection (e.g., SSL decryption), may choose to absorb certain
19 requests (e.g., because request router 306 maintains a cache of responses), may
20 arbitrarily modify requests before forwarding them to hosts 108, and so forth.
21 Exemplary implementations for request router 306 are also described further
22 below, particularly in the sections entitled "Exemplary Approach to Flexible
23 Network Load Balancing" and "Exemplary Classifying, Forwarding, and Request
24 Routing".
25

1 Connection migrator 310 enables a connection to be initially terminated at
2 load balancing unit 106 and then migrated such that the connection is subsequently
3 terminated at host 108. This connection migration can facilitate application-level
4 load balancing. Connection migrator 310 is capable of migrating a connection
5 from load balancing unit 106 to a host 108 in such a manner that that the original
6 termination at load balancing unit 106 is transparent to a requesting client 102 and
7 to applications 316 of the newly-terminating host 108. Tunneler 312 may utilize
8 an encapsulation scheme for the tunneling of packets that does not introduce an
9 overhead to each tunneled packet.

10 The functionality of tunneler 312 may also be used in situations that do not
11 involve a connection migration. Furthermore, connection migrator 310 and/or
12 tunneler 312 may additionally be used in non-load-balancing implementations.
13 Exemplary implementations for connection migrator 310, as well as for tunneler
14 312, are described further below, particularly in the section entitled "Exemplary
15 Connection Migrating with Optional Tunneling and/or Application-Level Load
16 Balancing".

17 Any given implementation of a load balancing unit 106 may include one or
18 more of the illustrated functions. Although illustrated separately, each of the
19 functions of blocks 302-314 may actually be interrelated with, overlapping with,
20 and/or inclusive of other functions. For example, health and/or load information
21 of health and load handler 314 may be used by classifier 304. Also, connection
22 migrator 310 and tunneler 312 work in conjunction with forwarder 302 and
23 classifier 304. Certain other exemplary overlapping and interactions are described
24 herein below.
25

1 In a described implementation, host 108 runs and provides access to one or
2 more applications 316. Generally, applications 316 include file delivery programs,
3 web site management/server programs, remote access programs, electronic mail
4 programs, database access programs, and so forth. Specifically, applications 316
5 may include, but are not limited to, web servers such as Internet Information
6 Server[®] (IIS) from Microsoft[®] Corporation, terminal servers such as Microsoft[®]
7 Terminal Server[™], and firewall and proxy products such as Internet Security and
8 Acceleration Server[™] (ISA). Although the specific application 316 examples in
9 the preceding sentence relate to Microsoft[®] products, network load balancing as
10 described herein is not limited to any particular vendor(s), application(s), or
11 operating system(s).

12 **Exemplary Approach to Flexible Network Load Balancing**

13 This section illuminates how the network load balancing implementations
14 described in this and other sections herein provide a flexible approach to network
15 load balancing. This section primarily references FIGS. 4-9B.

16 As noted above, network load balancing functionality may be scaled up by
17 replacing a first network load balancer with a second, bigger and more powerful
18 network load balancer. The hardware capabilities of the second network load
19 balancer replicate the entirety of the hardware capabilities of the first network load
20 balancer, except that a greater capacity is provided. This is an inflexible approach
21 that can be very inefficient, especially when only one network load balancing
22 feature is limiting performance and precipitating an upgrade of a network load
23 balancer.

24 FIG. 4 illustrates exemplary network load balancing infrastructure having
25 separated classifying and forwarding functionality. The separated classifying

1 functionality and forwarding functionality are represented by classifier 304 and
2 forwarder 302, respectively. Although classifying and forwarding functions are
3 described further below, especially in the section entitled "Exemplary Classifying,
4 Forwarding, and Request Routing", an initial description is presented here as an
5 example of interaction between network load balancing infrastructure functionality
6 and hosts 108.

7 In a described implementation, forwarder 302 corresponds to, and is the
8 network endpoint for, the virtual IP (VIP) address (or addresses). Forwarder 302
9 is a relatively low-level component that makes simplified and/or elementary policy
10 decisions, if any, when routing packets to a further or final destination. Forwarder
11 302 consults a routing table to determine this destination. Classifier 304 populates
12 the routing table based on one or more factors (e.g., host status information),
13 which are described further in other sections herein.

14 Clients 102 and hosts 108 also correspond to indicated network addresses.
15 Specifically, client 102(1) corresponds to address C1, client 102(2) corresponds to
16 address C2 ... client 102(m) corresponds to address Cm. Also, host 108(1)
17 corresponds to address H1, host 108(2) corresponds to address H2 ... host 108(n)
18 corresponds to address Hn.

19 Five communication paths (1)-(5) are shown in FIG. 4. Communication
20 path (1) is between client 102(1) and forwarder 302, and communication path (5)
21 is between forwarder 302 and host 108(1). Communication paths (2)-(4) are
22 between forwarder 302 and classifier 304. For simplicity in this example, the
23 connection associated with communication paths (1)-(5) is an HTTP TCP
24 connection. Furthermore, load balancing in this example relates to routing
25

1 incoming connections to the least loaded host 108, at least without any explicit
2 consideration of application-level load balancing.

3 Communication paths (1)-(5) indicate how forwarder 302 and classifier 304
4 load-balance a single HTTP TCP connection from client 102(1). At (1), client
5 102(1) initiates the TCP connection by sending a TCP SYN packet addressed to
6 the VIP address. The routing infrastructure of network 104 routes this packet to
7 forwarder 302 via router/switch 202(1), which is the "closest" router/switch 202 to
8 forwarder 302.

9 At (2), forwarder 302 consults a routing table, which may be internal to
10 forwarder 302 or otherwise accessible therefrom, in order to look up this
11 connection. This connection may be identified in the routing table by the TCP/IP
12 4-tuple (i.e., source IP address, source TCP port, destination IP address, destination
13 TCP port). Because this is the first packet of the connection, there is no entry in
14 the routing table. Forwarder 302 therefore applies the "default route" action,
15 which is to send this packet to classifier 304.

16 At (3), classifier 304 consults its (e.g., consolidated) cache of host status
17 information for hosts 108(1), 108(2) ... 108(n). Classifier 304 concludes that host
18 108(1) is available and the least loaded host 108 at this instant for this example.
19 Classifier 304 also "plumbs" a route in the routing table consulted by forwarder
20 302 for this TCP connection. For example, classifier 304 adds a route entry or
21 instructs forwarder 302 to add a route entry to the routing table that maps the TCP
22 connection (e.g., identified by the TCP 4-tuple) to a specific destination host 108,
23 which is host 108(1) in this example. More particularly, the route entry specifies
24 the network address H1 of host 108(1).

25

3p

1 At (4), classifier 304 sends the TCP SYN packet back to forwarder 302.
2 Alternatively, classifier 304 may forward this initial TCP SYN packet to host
3 108(1) without using forwarder 302. Other options available to classifier 304 are
4 described further below.

5 At (5), forwarder 302 can access a route entry for the connection
6 represented by the SYN packet, so it forwards the packet to host 108(1) at address
7 H1. Forwarder 302 also forwards all subsequent packets from client 102(1) for
8 this connection directly to host 108(1). In other words, forwarder 302 can avoid
9 further interaction with classifier 304 for this connection. One or a combination of
10 mechanisms, which are described further below, may be used to delete the route
11 entry when the connection ceases.

12 For communication path (5) in many protocol environments, forwarder 302
13 cannot simply send the packets from client 102(1) as-is to host 108(1) at network
14 address H1 because these packets are addressed to the VIP address, which is
15 hosted by forwarder 302 itself. Instead, forwarder 302 may employ one or more
16 of the following exemplary options:

17 1. Forwarder 302 performs Network Address Translation
18 (NAT) by (i) overwriting the source (client 102(1)) IP address (C1)
19 and port number with the IP address and NAT-generated port number
20 of forwarder 302 and (ii) overwriting the destination IP address
21 (VIP) with the IP address (H1) of the host (108(1)).

22 2. Forwarder 302 performs "Half-NAT" by overwriting
23 the destination IP address (VIP) with the IP address (H1) of the host
24 (108(1)) so that the source (client 102(1)) IP address (C1) and port
25 number are preserved.

3/1

1 3. Forwarder 302 "tunnels" the packets received from
 2 client 102(1) from forwarder 302 to host 108(1). Specifically in this
 3 example, tunneling can be effectuated by encapsulating each packet
 4 within a new IP packet that is addressed to host 108(1). Network-
 5 load-balancing-aware software on host 108(1) reconstructs the
 6 original packet as received at forwarder 302 from client 102(1).
 7 This original packet is then indicated up on a virtual interface at host
 8 108(1) (e.g., the VIP address corresponding to forwarder 302 is
 9 bound to this virtual interface at host 108(1)). Exemplary
 10 implementations of such tunneling are described further below with
 11 reference to tunneler 312, especially for connection migration
 12 scenarios and particularly in the section entitled "Exemplary
 13 Connection Migrating with Optional Tunneling and/or Application-
 14 Level Load Balancing".

15 Although FIGS. 4-9B show two specific separated functions, namely
 16 classifying and forwarding, it should be understood that other functions, such as
 17 those of request router 306, session tracker 308, connection migrator 310, and
 18 health and load handler 314, may also be scaled out independently (e.g., factored
 19 out independently), as is described further below. Furthermore, it should be noted
 20 that one or more than two functions may be separated and scaled out
 21 independently at different times and/or simultaneously. Also, although TCP/IP is
 22 used for the sake of clarity in many examples in this and other sections, the
 23 network load balancing principles described herein are applicable to other
 24 transmission and/or communication protocols.

25

51-

1 In the exemplary manner of FIG 4, network load balancing functions (such
2 as those shown in FIG 3) may be separated from each other for scalability
3 purposes. They may also be separated and duplicated into various configurations
4 for increased availability. Exemplary configurations for scalability and/or
5 availability are described below with reference to FIGS. 6-9B after the method of
6 FIG 5 is described.

7 FIG 5 is a flow diagram 500 that illustrates an exemplary method for
8 scaling out network load balancing infrastructure into different configurations.
9 Flow diagram 500 includes three blocks 502-506. Although the actions of flow
10 diagram 500 may be performed in other environments and with a variety of
11 software schemes, FIGS. 1-4 and 6-9B are used in particular to illustrate certain
12 aspects and examples of the method.

13 At block 502, network load balancing infrastructure is operated in a first
14 configuration. For example, each configuration may relate to one or more of a
15 selection, proportion, and/or interrelationship of different load balancing
16 functionalities; a number of and/or type(s) of different devices; an organization
17 and/or layout of different components; a distribution and/or allocation of
18 resources; and so forth. At block 504, the network load balancing infrastructure is
19 scaled out. For example, separated load balancing functionalities may be
20 expanded and/or concomitantly contracted on an individual and/or independent
21 basis. At block 506, the scaled out network load balancing infrastructure is
22 operated in a second configuration.

23 As noted above, a monolithic network load balancer may be scaled up by
24 increasing network load balancing functionality in its entirety by supplanting
25 previous network load balancing hardware with more-powerful network load

1 balancing hardware. In contradistinction, scaling out network load balancing
2 infrastructure can enable network load balancing (sub-)functions to be scaled out
3 individually and/or independently. It can also enable network load balancing
4 functions to be scaled out together or individually between and among different
5 numbers of devices. Device, component, and resource-oriented scaling out
6 examples are provided below.

7 FIG 6 illustrates a first exemplary network load balancing infrastructure
8 configuration from a device perspective. In this first device-oriented network load
9 balancing infrastructure configuration, three devices 602(1), 602(2), and 602(3)
10 are illustrated. However, one, two, or more than three devices 602 may
11 alternatively be employed.

12 As illustrated, a forwarder 302(1), a classifier 304(1), and a host 108(1) are
13 resident at and executing on device 602(1). A forwarder 302(2), a classifier
14 304(2), and a host 108(2) are resident at and executing on device 602(2). Also, a
15 forwarder 302(3), a classifier 304(3), and a host 108(3) are resident at and
16 executing on device 602(3). Thus, in this first device-oriented network load
17 balancing infrastructure configuration, a respective forwarder 302, classifier 304,
18 and host 108 are sharing the resources of each respective device 602.

19 In operation, forwarders 302 are the network endpoints for the VIP
20 address(es). Any classifier 304 may plumb a route for a connection to any host
21 108, depending on host status information. For example, classifier 304(2) may
22 plumb a route for a new incoming connection to host 108(3). In accordance with a
23 new route entry for this connection, forwarder 302(2) forwards subsequent packets
24 to host 108(3).
25

1 In one alternative device-oriented network load balancing infrastructure
2 configuration to which the illustrated first one may be scaled out, a fourth device
3 602(4) (not explicitly shown in FIG. 6) may be added that includes a forwarder
4 302(4), a classifier 304(4), and a host 108(4). If, on the other hand, sufficient
5 classification functionality is already present with classifiers 304(1-3) but
6 additional forwarding functionality can benefit the request handling of hosts 108, a
7 fourth device 602(4) may be added that includes a forwarder 302(4) and optionally
8 a host 108(4). For this scaled-out configuration, another classifier 304(1, 2, or 3)
9 may plumb routes for forwarder 302(4) to any of hosts 108(1, 2, or 3) and host
10 108(4), if present.

11 The first device-oriented exemplary network load balancing infrastructure
12 configuration of FIG. 6 may be especially appropriate for smaller hosting
13 situations in which separate devices for the network load balancing infrastructure
14 are not technically and/or economically worthwhile or viable. However, as the
15 hosting duties expand to a greater number (and/or a greater demand on the same
16 number) of hosts 108 or if the network load on hosts 108 is significant, the first
17 device-oriented exemplary network load balancing infrastructure configuration
18 may be scaled out to accommodate this expansion, as represented by a second
19 device-oriented exemplary network load balancing infrastructure configuration of
20 FIG. 7.

21 FIG. 7 illustrates a second exemplary network load balancing infrastructure
22 configuration from a device perspective. In this second device-oriented network
23 load balancing infrastructure configuration, three devices 602(1), 602(2), and
24 602(3) are also illustrated. Again, one, two, or more than three devices 602 may
25 alternatively be employed.

1 As illustrated, forwarder 302(1) and classifier 304(1) are resident at and
2 executing on device 602(1). Forwarder 302(2) and classifier 304(2) are resident at
3 and executing on device 602(2). Also, forwarder 302(3) and classifier 304(3) are
4 resident at and executing on device 602(3). Thus, in this second device-oriented
5 network load balancing infrastructure configuration, each respective forwarder
6 302 and classifier 304 are not sharing the resources of each respective device 602
7 with a host 108. Furthermore, the network load balancing infrastructure may be
8 servicing any number of hosts 108.

9 In operation, forwarders 302 are again the network endpoints for the VIP
10 address(es). Also, any classifier 304 may plumb a route for a connection to any
11 host 108, depending on host status information. For example, classifier 304(3)
12 may plumb a route for a new incoming connection to host 108(2). In accordance
13 with a new route entry for this connection, forwarder 302(3) forwards subsequent
14 packets to host 108(2).

15 Hence, network load balancing infrastructure as realized in software, for
16 example, may be scaled out by moving the network load balancing infrastructure
17 (or part thereof) from devices that are shared with hosts 108 to devices that are not
18 shared with hosts 108. Also, as alluded to above for FIG. 6, another device 602(4)
19 may be added to the network load balancing infrastructure to provide additional
20 forwarding functionality, additional classifying functionality, additional
21 functionality of both types, and so forth.

22 FIGS. 8A and 8B illustrate first and second exemplary network load
23 balancing infrastructure configurations from a component perspective. As
24 illustrated, first component-oriented exemplary network load balancing
25 infrastructure configuration 800 includes four components. Second component-

1 oriented exemplary network load balancing infrastructure configuration 850
2 includes six components. An alternative second configuration 850 includes a
3 seventh component as indicated by the dashed-line block, which is described
4 further below.

5 Specifically, first component-oriented exemplary network load balancing
6 infrastructure configuration 800 (or first configuration 800) includes (i) two
7 forwarders 302(1) and 302(2) and (ii) two classifiers 304(1) and 304(2). Second
8 exemplary component-oriented network load balancing infrastructure
9 configuration 850 (or second configuration 850) includes (i) four forwarders
10 302(1), 302(2), 302(3), and 302(4) and (ii) two classifiers 304(1) and 304(2).
11 Thus, first configuration 800 is scaled out to second configuration 850 by adding
12 two components, which are forwarding components in this example.

13 In a described implementation, each respective network-load-balancing-
14 related functional component corresponds to a respective device (not explicitly
15 shown in FIG. 8A or 8B); however, each component may alternatively correspond
16 to part of a device or more than one device. For example, forwarders 302(1) and
17 302(2) may be distributed across three devices. Or forwarder 302(1) and classifier
18 304(1) may correspond to a first device, and forwarder 302(2) and classifier
19 304(2) may correspond to a second device.

20 Two network-load-balancing-related functional components are added to
21 scale out first configuration 800 to second configuration 850. However, one
22 component (or more than two) may alternatively be added to scale out the network
23 load balancing infrastructure. Furthermore, two or more different types of
24 functional components may be scaled out "simultaneously". For example, as
25 illustrated by the dashed-line block, another classifying component (e.g., classifier

304(3)) may also be added when scaling out first configuration 800 to second configuration 850.

Moreover, scaling by two or more different types of functional components may be performed in similar (e.g., equivalent) or dissimilar proportions to each other. As illustrated, adding forwarder components 302(3) and 302(4) while not adding any classifier component 304 or while adding a single classifier component 304(3) represent a scaling out at dissimilar proportions. However, two classifier components 304(3) and 304(4) (the latter of which is not explicitly illustrated in FIG. 8B) may be added while the two forwarder components 302(3) and 302(4) are added for a scaling out at similar proportions. Regardless, each individual network-load-balancing-related functional component may consume a different amount of the available network load balancing infrastructure resources, as is described with reference to FIGS. 9A and 9B.

FIGS. 9A and 9B illustrate first and second exemplary network load balancing infrastructure configurations from a resource perspective. First resource-oriented exemplary network load balancing infrastructure configuration 900 (or first configuration 900) includes a first resource distribution or allocation for a load balancing unit 106. Second resource-oriented exemplary network load balancing infrastructure configuration 950 (or second configuration 950) includes a second resource distribution for load balancing unit 106.

As illustrated, first configuration 900 includes a 70%-30% resource distribution, and second configuration 950 includes a 40%-60% resource distribution. Such resources may include total device resources (e.g., number of devices), processing resources (e.g., number of processor cycles), memory resources (e.g., portion of cache, main memory, etc.), network bandwidth and/or

1 interface resources (e.g., bits per second and/or physical network interface cards
2 (NICs)), and so forth.

3 Specifically for first configuration 900, forwarder 302 consumes 70% of
4 the resources of load balancing unit 106 while classifier 304 consumes 30% of
5 these resources. After reallocation during a scaling out procedure to produce
6 second configuration 950, forwarder 302 consumes 40% of the resources of load
7 balancing unit 106 while classifier 304 consumes 60% of these resources.

8 In an exemplary situation, first configuration 900 might facilitate better
9 network load balancing performance when fewer, longer transactions are being
10 handled by the associated hosts (not shown in FIGS. 9A and 9B) because
11 classification functionality is utilized upon initial communication for a connection
12 and forwarding functionality is utilized thereafter. Second configuration 950, on
13 the other hand, might facilitate better network load balancing performance when
14 more, shorter transactions are being handled by the associated hosts because the
15 classification functionality is utilized for a greater percentage of the total number
16 of packets funneled through the network load balancing infrastructure. In this
17 situation, if request routing functionality is also being employed, then request
18 router(s) 306 are also allocated a percentage of the total computing resources. The
19 resource distribution among the three functionalities may be adjusted while
20 handling connections (e.g., adjusted "on the fly") depending on current resource
21 consumption and/or deficits.

22 As indicated above with reference to FIGS. 2 and 3, each load balancing
23 unit 106 may correspond to all or a part of a total network load balancing
24 infrastructure 106. For any given physically, logically, arbitrarily, etc. defined or
25 stipulated load balancing unit 106, the resources thereof may be re-allocated

42

1 during a scale out procedure. More specifically, a resource distribution
2 between/among different network-load-balancing-related separated functions of a
3 load balancing unit 106 may be altered in a scale out procedure. Furthermore,
4 more than two different functions, as well as other network-load-balancing-related
5 functions that are not specifically illustrated in FIGS. 9A and 9B, may be allocated
6 differing resource percentages.

7 The percentage of total system resources allocated to all load balancing
8 functions may also be altered in a scale out procedure. As a general processing
9 power example, the percentage of total processing power that is devoted to load
10 balancing may be gradually increased as the amount of traffic that needs to be load
11 balanced increases.

12 Network load balancing software may optionally perform monitoring to
13 analyze and determine whether resources should be reallocated. For example, the
14 network load balancing software may monitor the processor utilization of different
15 network-load-balancing-related functions. The actual reallocation may also
16 optionally be performed automatically by the network load balancing software in
17 an offline or online mode.

18 It should be understood that a scaling out capability of network load
19 balancing infrastructure (e.g., as realized at least partially in software) as described
20 herein may relate to different installations and not necessarily a change to a single
21 installation. In a resource-oriented example, network load balancing infrastructure
22 as described herein may be configured in accordance with one resource
23 distribution in one installation environment and may be configured in accordance
24 with another different resource distribution in another installation environment
25 having different operational parameters. Additionally, the capabilities, features,

1 options, etc. described above with regard to scaling out are also applicable for
2 "scaling in". In other words, resources devoted to network load balancing
3 infrastructure (or sub-functions thereof) may also be reduced.

4 **Exemplary Health and Load Handling**

5 This section describes how host status information, such as health and/or
6 load information, may be collected for and utilized in network load balancing.
7 This section primarily references FIGS. 10-18 and illuminates health and load
8 functionality such as that provided by health and load handler 314 (of FIG 3). As
9 described above with reference to FIG 3, each host 108 hosts one or more
10 applications 316. Health and load handler 314 utilizes health and/or load
11 information that relates to applications 316 and/or hosts 108 for certain described
12 implementations of network load balancing.

13 FIG 10 illustrates an exemplary network load balancing approach that
14 involves host status information (HSI) 1006. Each host 108(1), 108(2) ... 108(n)
15 includes one or more applications 316(1), 316(2) ... 316(n), respectively. These
16 hosts 108 generally and these applications 316 specifically may change statuses
17 from time to time.

18 For example, hosts 108 and applications 316 may be accepting new
19 connections or not accepting new connections. Also, they may be quickly
20 handling client requests or slowly handling client requests. Furthermore, they may
21 have many resources in reserve or few unused resources. All or any part of such
22 data, or other data, may comprise host status information 1006. Generally, host
23 status information 1006 provides an indication of the status of some aspect of
24 hosts 108 and/or applications 316 that are running thereon.
25

cfm

1 In a described implementation, each host 108(1), 108(2) ... 108(n) includes
2 a host status information (HSI) determiner 1002(1), 1002(2) ... and 1002(n),
3 respectively. Each host 108(1), 108(2) ... 108(n) also includes a host status
4 information (HSI) disseminator 1004(1), 1004(2) ... and 1004(n), respectively.
5 Each host status information determiner 1002 and/or host status information
6 disseminator 1004 may be part of load balancing infrastructure (LBI) 106.

7 Each host status information determiner 1002 determines host status
8 information 1006 for its respective host 108 and/or applications 316 that are
9 running thereon. Exemplary techniques for determining such host status
10 information 1006 are described below with reference to FIGS. 12-14, and
11 particularly FIG. 13A. Each host status information disseminator 1004
12 disseminates host status information 1006 for its respective host 108 and/or
13 applications 316 to load balancing infrastructure 106 (e.g., those portion(s) of load
14 balancing infrastructure 106 that are not located on hosts 108). Exemplary
15 techniques for disseminating such host status information 1006 are described
16 below with reference to FIGS. 12-17, and particularly FIGS. 13B and 15-17.

17 Specifically, each host status information disseminator 1004 disseminates
18 host status information 1006 (directly or indirectly) to each load balancing unit
19 (LBU) 106 of load balancing infrastructure 106 that includes at least one health
20 and load handler 314 and/or classifier 304. Load balancing infrastructure 106
21 refers to host status information 1006 when implementing network load balancing.
22 For example, as indicated by logic 1008, load balancing infrastructure 106 is
23 capable of making load balancing decisions responsive to host status information
24 1006.
25

1 In operation at (1), host status information determiners 1002 determine host
2 status information 1006 for respective hosts 108 and/or applications 316. At (1)
3 and (2), host status information disseminators 1004 disseminate host status
4 information 1006 from hosts 108 to load balancing infrastructure 106. For
5 example, host status information 1006 may be disseminated to individual load
6 balancing units 106. At (3), logic 1008 makes network load balancing decisions
7 responsive to host status information 1006. At (4), connections are forwarded to
8 targeted hosts 108 based on these network load balancing decisions.

9 FIG. 11 is a flow diagram 1100 that illustrates an exemplary method for
10 network load balancing that involves host status information. Flow diagram 1100
11 includes three blocks 1102-1106. Although the actions of flow diagram 1100 may
12 be performed in other environments and with a variety of software schemes, FIGS.
13 1-3 and 10 are used in particular to illustrate certain aspects and examples of the
14 method.

15 At block 1102, host status information is sent from hosts to load balancing
16 units. For example, host status information 1006 may be sent from hosts 108 to
17 load balancing units 106. At block 1104, the host status information is received
18 from the hosts at the load balancing units. For example, load balancing units 106
19 may receive host status information 1006 from hosts 108. At block 1106, load
20 balancing decisions are made responsive to the received host status information.
21 For example, logic 1008 at load balancing units 106 may make decisions for
22 network load balancing responsive to host status information 1006.

23 Thus in FIG. 10, load balancing infrastructure 106 collects host status
24 information 1006 from hosts 108 (and/or applications 316 thereof) and load
25 balances incoming requests that are directed to hosts 108 responsive to host status

1 information 1006. As described further below with reference to FIGS. 12-18, this
2 host status information 1006 may be application-specific. As also described
3 further below, examples of host status information 1006 include health and/or load
4 information.

5 FIG. 12 illustrates an exemplary network load balancing approach that
6 involves health and/or load information (HLI) 1206. Hosts 108(1), 108(2) ...
7 108(n) are coupled to load balancing units 106(1), 106(2) ... 106(u) via a
8 communication linkage 1210 such as a network.

9 As illustrated, hosts 108 communicate health and load information 1206 to
10 load balancing units 106 using communication linkage 1210. The bi-directional
11 communication of health and load information 1206, as indicated by the double-
12 pointed arrow, refers to a two-way communication from load balancing units 106
13 to hosts 108 that provides certain completeness, coherency, correctness, etc. such
14 that hosts 108 and/or load balancing units 106 may fail independently of one
15 another. Such two-way communications from load balancing units 106 to hosts
16 108 are described further below with particular reference to FIG. 15.

17 Health information reflects whether a given host and/or application is
18 capable of handling client requests. Load information reflects the number,
19 amount, and/or level of client requests that the given host and/or application is
20 capable of handling at a particular moment. In other words, load can reflect
21 directly and/or inversely an available number, amount, and/or level of total
22 capacity of the given host and/or application. As noted above, implementations
23 described with reference to FIGS. 12-18 focus on health and/or load information;
24 however, those implementations are also applicable to general status information
25 for hosts (including the applications thereof).

1 In a described implementation, each host 108(1), 108(2) ... 108(n) includes
2 a respective health and load infrastructure (H&LI) component 1202(1), 1202(2) ...
3 1202(n). Each health and load infrastructure component 1202 may optionally be a
4 portion of load balancing infrastructure 106 that is resident at and executing on
5 each host 108. Health and load information 1206 may be realized in software.
6 When functioning, each health and load infrastructure 1202(1), 1202(2) ...
7 1202(n) creates and maintains a respective health and load (H&L) table 1204(1),
8 1204(2) ... 1204(n).

9 These health and load tables 1204 may include application-specific entries.
10 Health and load information 1206 that is stored in health and load tables 1204 may
11 be independent of load balancing infrastructure 106. For example, administrators,
12 designers, etc. may specify criteria for health and load information 1206 at
13 configuration time. Additionally, entities external to a device that is or that has a
14 host 108 may contribute to determining health and load information 1206 for
15 applications 316 on the device. An exemplary health and load table 1204 is
16 described further below with reference to FIG. 13A.

17 Each load balancing unit 106(1), 106(2) ... 106(u) includes a respective
18 consolidated health and load (H&L) cache 1208(1), 1208(2) ... 1208(u). Each
19 consolidated health and load cache 1208 includes the information from each health
20 and load table 1204(1), 1204(2) ... 1204(n). Consequently, each load balancing
21 unit 106 is provided with quick (e.g., cached) access to health and load
22 information 1206 for each host 108 for which load balancing units 106 are load
23 balancing network traffic.

24 In operation, health and load infrastructures 1202 push health and load
25 information 1206 from health and load tables 1204 to consolidated health and load

1 caches 1208. The mechanism to provide health and load information 1206 is
2 event driven such that changes to health and load tables 1204 are provided to
3 consolidated health and load caches 1208 in a timely, scaleable manner.

4 FIG 13A is an exemplary health and load table 1204 as illustrated in FIG.
5 12. In a described implementation, health and load table 1204 includes multiple
6 entries 1302 that are each associated with a different application 316. Each entry
7 1302 may correspond to a row in health and load table 1204 that has three
8 columns. These columns correspond to application identifier (ID) 1302(A),
9 application status characterization 1302(B), and load balancer directive 1302(C).

10 Because each entry 1302 is associated with a particular application 316, a
11 row is added as each application is spun up (e.g., by an administrator). Likewise,
12 a row is deleted/removed each time an application is closed down. Similarly,
13 individual fields in columns 1302(A), 1302(B), and/or 1302(C) are
14 modified/updated when a value thereof changes. For example, when a status
15 characterization value changes for a given application 316, a value in a field of
16 application status characterization 1302(B) for entry 1302 of the given application
17 316 is updated.

18 The additions and deletions of entries 1302 for applications 316 may be
19 effectuated with input from a control manager at the host 108. For example, a
20 control manager portion of an operating system knows when an application 316 is
21 started and stopped because it is actively involved in the starting and stopping of
22 applications 316. Hence, a control manager may identify that it has, at least in
23 part, started an application 316, and the control manager may establish that it has,
24 at least in part, stopped the application 316. Health and load infrastructure 1202
25 may therefore be informed of the starting and stopping of applications 316 by the

41

1 control manager. Hence, no such explicit communication from applications 316
2 has to be provided to health and load infrastructure 1202. An example of a control
3 manager is the Service Control Manager (SCM) of the Windows® Operating
4 System from Microsoft® Corporation.

5 Application identifier 1302(A) includes information that is used to uniquely
6 identify the application 316 to which entry 1302 is associated. Application
7 identifier 1302(A) may include one or more of the following for the associated
8 application 316: the virtual IP address and port, the physical IP address and port,
9 the protocol used, and any protocol-specific information. The protocol may be
10 HTTP, IPsec, SOAP, and so forth. The protocol-specific information may be a
11 URL pattern or string to further delineate the application associated with entry
12 1302. Thus, application identifier 1302(A) more particularly refers to a specific
13 application endpoint on a particular host 108.

14 Other application identifiers may alternatively be employed. For example,
15 to reduce communication bandwidth, application identifier 1302(A) may be a 32-
16 bit number that maps to the above exemplary information at health and load
17 infrastructure 1202 and at load balancing units 106. Moreover, any of the fields in
18 entry 1302 may actually contain a globally unique identifier (GUID) that is used
19 as a key to look up the true information for the field.

20 Application status characterization 1302(B) includes information that
21 reflects the status of the application 316 to which entry 1302 is associated.
22 Application status characterization 1302(B) includes the following for the
23 associated application 316: application health, application load, and application
24 capacity. Application health is a quasi-Boolean value that indicates whether an
25 application is functioning. Application health can be healthy, failing, or unknown.

1 Application health is a relatively-instantaneous value and is communicated with
2 relatively low latency (e.g., of approximately a second or a few seconds) to load
3 balancing units 106 when the application health value changes.

4 Application load is a value that indicates how occupied or busy a given
5 application is and thus, directly or inversely, how much additional load the given
6 application can handle. Application load is a relatively slowly-changing or
7 averaged value that can be smoothed with a hysteresis-inducing mechanism, if
8 desired, to eliminate transient spikes of increased or decreased load. It is
9 communicated relatively infrequently to load balancing units 106 (e.g.,
10 approximately one to four times a minute). The value of application load is given
11 meaning with regard to application capacity.

12 Application capacity is a value that indicates the maximum capacity of the
13 application. It is selected in a generic manner to be meaningful for a given context
14 but still sufficiently flexible for other contexts. Application capacity is a unit-less,
15 bounded number (e.g., 0-99) that is determinable at configuration time. It may be
16 based on processing power, memory size/speed, network access, some
17 combination thereof, and so forth. Application capacity expresses relative
18 capacities between and among other applications of the same type in a set of hosts
19 108(1, 2 ... n).

20 Thus, relative to application capacity, application load gains meaning.
21 Application load for a given application is a percentage of application capacity for
22 the given application. Alternatively, application load can be expressed as a unit-
23 less number from which the percentage can be ascertained in conjunction with the
24 value of application capacity.

25

1 Load balancer directive 1302(C) includes information that reflects the
2 desired and/or expected state of the directive established by health and load
3 infrastructure 1202 for load balancing units 106 with respect to an application 316
4 to which entry 1302 is associated. Load balancer directive 1302(C) includes the
5 following for the associated application 316: target load balancing state and
6 current load balancing state.

7 The target load balancing state reflects the state of the directive to load
8 balancing units 106 as desired by health and load infrastructure 1202. The current
9 load balancing state reflects what health and load infrastructure 1202 understands
10 the current state of the directive to load balancing units 106 to be as recorded at
11 load balancing units 106. The current load balancing state thus reflects the load
12 balancing directive that health and load infrastructure 1202 expects load balancing
13 units 106 to be currently operating under as dictated using a communication
14 protocol. Such an exemplary communication protocol is described further below
15 with reference to FIG. 15. The interaction and relationship between the target load
16 balancing state and the current load balancing state is also further clarified with
17 the description of FIG. 15.

18 The target load balancing state and the current load balancing state may
19 each take a value of active, inactive, or draining. An active directive indicates that
20 new requests/connections are welcome and may be targeted at the application that
21 is associated with entry 1302. An inactive directive indicates that no additional
22 packets should be forwarded to the associated application. A draining directive
23 indicates that no packets for new requests/connections should be sent to the
24 associated application but that packets for existing requests/connections should
25 continue to be forwarded to the associated application.

1 In a described implementation, the definitive version of respective health
2 and load information 1206 is stored at health and load tables 1204 that are located
3 at each respective host 108 of multiple hosts 108. With this implementation, if a
4 host 108 crashes, the health and load information 1206 that is lost pertains to those
5 applications 316 that also crashed. A measure of high availability is therefore
6 gained automatically without duplicating data. However, the definitive version of
7 health and load information 1206 may alternatively be stored elsewhere. Other
8 such storage options include load balancing units 106 themselves, a host 108 that
9 (as its sole task or along with hosting duties) stores and maintains health and load
10 information 1206 for multiple other (including all other) hosts 108, another
11 separate and/or external device, and so forth.

12 If the definitive version of health and load information 1206 is stored and
13 maintained elsewhere besides being distributed across hosts 108(1, 2 ...n), such
14 health and load information 1206 may be stored redundantly (e.g., also stored in a
15 duplicative device, backed-up, etc.) for high-availability purposes. Exemplary
16 proxy scenarios for storing health and load information 1206 are described below
17 with reference to FIGS. 17A and 17B. FIG. 17A is directed to a proxy scenario for
18 health and load tables 1204, and FIG. 17B is directed to a proxy scenario for
19 consolidated health and load caches 1208.

20 FIG. 13B is an exemplary consolidated health and load cache 1208 as
21 illustrated in FIG. 12. In a described implementation, each consolidated health and
22 load cache 1208 in each load balancing unit 106 includes at least part of the
23 information stored in each health and load table 1204 for each health and load
24 infrastructure 1202 at each host 108. The cached health and load information may
25 be organized in any manner in consolidated health and load cache 1208.

45

1 As illustrated, consolidated health and load cache 1208 includes a cache for
2 each host 108(1), 108(2) ... 108(n) that replicates part or all of the information in
3 the health and load table 1204 of each respective host 108(1, 2 ... n). Specifically,
4 consolidated health and load cache 1208 includes a cache for host #1 1304(1), a
5 cache for host #2 1304(2) ... a cache for host #n 1304(n). Thus, the illustrated
6 consolidated health and load cache 1208 is organized at a broad level by host
7 108(1, 2 ... n), with each individual cache 1304 including application-specific
8 entries for the corresponding respective host 108(1, 2 ... n). Alternatively,
9 consolidated health and load cache 1208 may be organized at a broad level by type
10 of application 316, with individual blocks that are directed to a specific application
11 type further divided by host 108(1, 2 ... n). Other data structure formats may also
12 be employed.

13 FIG. 14 is a flow diagram that illustrates an exemplary method for network
14 load balancing that involves health and load information. Flow diagram 1400
15 includes eight blocks 1402-1416. Although the actions of flow diagram 1400 may
16 be performed in other environments and with a variety of software schemes, FIGS.
17 1-3 and 12-13B are used in particular to illustrate certain aspects and examples of
18 the method. For example, the actions of two blocks 1402-1404 are performed by a
19 host 108, and the actions of six blocks 1406-1416 are performed by a load
20 balancing unit 106.

21 At block 1402, health and load information at a host is determined. For
22 example, health and load information 1206 for applications 316(2) may be
23 ascertained by health and load infrastructure 1202(2) and stored in health and load
24 table 1204(2) at host 108(2). At block 1404, the determined health and load
25 information is disseminated to load balancing units. For example, health and load

46

1 infrastructure 1202(2) may send health and load information 1206 for applications
2 316(2) to load balancing units 106(1, 2 ... u). As indicated by arrow 1418, the
3 actions of blocks 1402 and 1404 are repeated so that (application) health and load
4 may be continually monitored and updated as changes occur.

5 At block 1406, health and load information is received from hosts. For
6 example, load balancing unit 106(1) may receive health and load information 1206
7 from multiple hosts 108(1, 2 ... n), which includes health and load information
8 1206 for applications 316(2) of host 108(2). At block 1408, the received health
9 and load information is cached. For example, load balancing unit 106(1) may
10 store health and load information 1206 from hosts 108(1, 2 ... n) into consolidated
11 health and load cache 1208(1). With reference to the FIG. 13B implementation of
12 a consolidated health and load cache 1208(1), health and load information 1206
13 for applications 316(2) from host 108(2) may be stored in cache for host #2
14 1304(2). As indicated by arrow 1420, the actions of blocks 1406 and 1408 are
15 repeated so that (application) health and load information may be continually
16 received and updated as changes occur.

17 As indicated by dashed arrow 1422, load balancing units 106 are also
18 handling communications from clients 102 while handling (application) health and
19 load issues. At block 1410, a packet requesting a new connection is received. For
20 example, load balancing unit 106(1) may receive a TCP SYN packet from client
21 102(2) through network 104. At block 1412, the cached health and load
22 information is consulted. For example, load balancing unit 106(1) may consult
23 consolidated health and load cache 1208(1). More particularly, load balancing unit
24 106(1) may consult entries that are associated with the application to which the
25 TCP SYN packet is directed across caches for hosts #1, #2 ... #n 1304(1, 2 ... n).

1 At block 1414, a host is selected responsive to the cached health and load
 2 information. For example, load balancing unit 106(1) may select host 108(2)
 3 having application(s) 316(2) responsive to health and load information 1206 that is
 4 cached in consolidated health and load cache 1208(1). The selected application
 5 316 (and host 108) should be healthy and able to accept additional load (e.g.,
 6 possibly the least loaded application among those applications that are of the
 7 application type to which the TCP SYN packet is directed).

8 The consulting of the cached health and load information (at block 1412)
 9 and the host-selecting responsive to the cached health and load information (at
 10 block 1414) may be performed prior to reception of a specific new-connection-
 11 requesting packet and/or using a batched scheme. Also, the selecting may be in
 12 accordance with any of many schemes. For example, a token based or a round-
 13 robin based scheme may be employed. With either scheme, the selection may
 14 involve a weighting of relative loads among the application options. This
 15 consultation and selection, along with the token and round-robin based schemes,
 16 are described further below with reference to FIG. 18 and in the section entitled
 17 "Exemplary Classifying, Forwarding, and Request Routing", especially with
 18 regard to classifying functionality.

19 After the target host is selected at block 1414, the new-connection-
 20 requesting packet may be sent thereto. At block 1416, the packet received from
 21 the client is forwarded to the selected host. For example, the TCP SYN packet is
 22 forwarded from load balancing unit 106(1) to selected host 108(2). The
 23 forwarding of this initial packet may be effectuated directly by a classifier 304 or
 24 by a forwarder 302, as is also described further below in the section entitled
 25 "Exemplary Classifying, Forwarding, and Request Routing".

1 For a described implementation, health and load infrastructure 1202 is
2 resident at and distributed across multiple hosts 108 as well as being located at
3 load balancing units 106 (as represented by health and load handler 314). Health
4 and load infrastructure 1202 has three responsibilities. First, it exposes listening
5 point(s) to attain application status updates for application status characterizations
6 1302(B) of health and load tables 1204. Second, it synthesizes the application
7 status information to determine what load balancing units 106 should do, which is
8 embodied in load balancer directive 1302(C). Third, health and load infrastructure
9 1202 communicates this directive from hosts 108 to load balancing units 106.

10 The directive content of load balancer directive 1302(C) is effectively a
11 digested version of the information for application status characterizations
12 1302(B). However, load balancing units 106 may also receive the raw information
13 of application status characterizations 1302(B) as well as this processed directive.
14 The communication of the content of these and other fields of health and load
15 tables 1204 is accomplished using a message protocol that is described below with
16 reference to FIG. 15.

17 FIG. 15 illustrates an exemplary message protocol 1500 for the health and
18 load information-related communications that are illustrated in FIG. 12 between
19 hosts 108 and load balancing units 106. Generally, an event-driven mechanism is
20 used to push changes to health and load tables 1204 from hosts 108 to load
21 balancing units 106. In other words, for a described implementation, information
22 is transmitted from hosts 108 to load balancing units 106 when health and load
23 tables 1204 are updated. This avoids periodically sending a snapshot of all of each
24 health and load table 1204, which reduces network bandwidth consumption by
25 health and load infrastructure 1202.

1 Message protocol 1500 may be implemented using any available message
2 transport mechanism. Such mechanisms include reliable multicast transmission,
3 point-to-point transmission (e.g., user datagram protocol (UDP)), and so forth. As
4 illustrated, message protocol 1500 includes seven message types 1502-1514: a
5 heartbeat message 1502, a goodbye message 1504, a row change message 1506, a
6 get table snapshot message 1508, a send table snapshot message 1510, a postulate
7 table state message 1512, and a postulate wrong message 1514.

8 It should be understood that, with the exception of arrows 1516 and 1518,
9 no temporal relationship between or among the different messages types 1502-
10 1514 is implied by the illustration. For example, a row change message 1506 does
11 not typically follow a goodbye message 1504.

12 Heartbeat message 1502 indicates that a particular host 108 is functioning
13 and provides some error checking for the content of a corresponding particular
14 health and load table 1204 with respect to a corresponding particular cache for the
15 particular host 1304 in consolidated health and load cache 1208. Each health and
16 load infrastructure 1202 at each host 108 sends a heartbeat message directly or
17 indirectly to each consolidated health and load cache 1208 at each load balancing
18 unit 106.

19 Heartbeat messages 1502 address the aging-out problem for data in
20 consolidated health and load caches 1208 that arises, in part, because a snapshot of
21 the entirety of each health and load table 1204 is not periodically transmitted to
22 each load balancing unit 106. A transmission scheme for heartbeat messages 1502
23 is described further below with reference to FIG. 16.

24 Heartbeat messages 1502 include an identifier for the host, error checking
25 data, and optionally a DNS name. The identifier of the host may be a unique (e.g.,

1 32-bit) number that is selected at configuration time. The error checking data may
2 be a checksum, a state-change sequence number, a generation number, a CRC
3 value, etc. that enables a receiving load balancing unit 106 to validate that the
4 contents of its consolidated health and load cache 1208 comports with the contents
5 of the health and load table 1204 of the transmitting host 108. If a generation
6 number approach is employed, then multiple generation IDs can be used with each
7 generation ID assigned to a "chunk" of applications. Messages can then refer to a
8 chunk number or a chunk number/generation ID pair, depending on the context.

9 The error checking data may be a single value for the health and load table
10 1204 overall, or it may be multiple values determined on a per-entry 1302 basis.
11 The DNS name may optionally be sent (e.g., every "x" heartbeats) to verify or
12 update the current correct network address for the host.

13 Goodbye message 1504 is sent from a particular host 108 to load balancing
14 units 106 to indicate that the particular host 108 is planning to shutdown.
15 Goodbye message 1504 includes a host identifier that may be indexed/mapped to a
16 network address for the particular host 108. Goodbye message 1504 is used for
17 clean, intentional shutdowns by hosts 108 to precipitate a "fast clear". However, if
18 a goodbye message 1504 is lost, caches eventually age out the particular host's
19 108 entries because heartbeat messages 1502 are no longer sent.

20 Row change message 1506 is sent from a particular host 108 to load
21 balancing units 106 to indicate that the health and/or load for a given application
22 316 of the particular host 108 has changed. Row change message 1506 includes a
23 host identifier, an application identifier, an operation, and data for the operation.
24 Exemplary host identifiers are described above with regard to heartbeat messages
25 1502 and goodbye messages 1504. Exemplary application identifiers are

1 described above with regard to application identifier 1302(A) of an application-
2 associated entry 1302 of health and load tables 1204.

3 The row change operation may be add, delete, or update. In other words,
4 the data for the operation may be added to (for an add operation) or a replacement
5 for (for an update operation) information already present at consolidated health
6 and load caches 1208 at load balancing units 106. For a delete operation, no data
7 need be provided. Message protocol 1500 is defined such that multiple operations
8 may be stipulated to be performed for a single row change message 1506. Hence
9 for a particular host identifier, sets of an application identifier, operation, and
10 operation data may be repeated for multiple applications 316 of the host 108 that is
11 identified by the particular host identifier.

12 Get table snapshot message 1508 is sent from a particular load balancing
13 unit 106 for a particular consolidated health and load cache 1208 to an individual
14 host 108 or hosts 108. This get table snapshot message 1508 requests that health
15 and load infrastructure 1202 at hosts 108 provide a snapshot of the respective
16 health and load table 1204 for the respective host 108. This message includes an
17 identification of the requesting load balancing unit 106 and may be used by a load
18 balancing unit 106 (i) after it has failed and then recovered; (ii) after a host 108
19 fails, recovers, and begins sending heartbeat messages 1502 again; (iii) if a row
20 change message 1506 is sent to load balancing unit 106, but the message gets
21 dropped, so its consolidated health and load cache 1208 is out of sync with the
22 respective health and load table 1204 for the respective host 108; and (iv) so forth.

23 For the third (iii) situation, the lack of synchronization between
24 consolidated health and load cache 1208 and the respective health and load table
25 1204 for the respective host 108 is discovered by a subsequent heartbeat message

4b

1 1502 from the respective host 108 because the "error checking" will indicate that
2 consolidated health and load cache 1208 is out of date. Load balancing unit 106
3 can then send a get table snapshot message 1508 so that it can update its
4 consolidated health and load cache 1208. Thus, for any of the three (i, ii, iii)
5 exemplary situations, load balancing unit 106 subsequently reconstitutes its
6 consolidated health and load cache 1208 using get table snapshot 1508. Get table
7 snapshot 1508 may be sent repeatedly to each host 108 in a point-to-point manner
8 or may be sent one time to many hosts 108 in a multicast manner.

9 Send table snapshot message 1510 is sent from an individual host 108 to a
10 particular load balancing unit 106 after the individual host 108 has received a get
11 table snapshot message 1508 from the particular load balancing unit 106 as
12 indicated by arrow 1516. The contents of a send table snapshot message 1510 is
13 prepared by health and load infrastructure 1202 and includes all or at least
14 multiple rows of the health and load table 1204 of the individual host 108 so that
15 the particular load balancing unit 106 may rebuild its consolidated health and load
16 cache 1208. Send table snapshot message 1510 may be a separately designed
17 message, or it may be equivalent to a sequence of add operations in a row change
18 message 1506.

19 Postulate table state message 1512 and postulate wrong message 1514 are
20 related to the target load balancing state and the current load balancing state of
21 load balancer directive 1302(C) of an entry 1302 in a health and load table 1204.
22 The target load balancing state is the directive that health and load infrastructure
23 1202 desires load balancing units 106 to be operating under. The current load
24 balancing state is the directive that health and load infrastructure 1202 expects or
25

d/1

1 believes that load balancing units 106 are currently operating under. Generally,
2 the two load balancing states are identical.

3 However, the target load balancing state may differ from the current load
4 balancing state during a transitional period for a state directive change. For
5 example, the target load balancing state and the current load balancing state are
6 both initially set to active. A problem with host 108 and/or an application 316
7 thereof is detected and the target load balancing state directive is switched to
8 draining. This draining directive is communicated to load balancing units 106
9 using a row change message 1506.

10 There is a delay before this directive change is noted in all consolidated
11 health and load caches 1208 of all load balancing units 106. During this
12 transitional period, the target load balancing state is draining while the current load
13 balancing state is still active at health and load table 1204 of host 108. Before
14 changing the current load balancing state to draining, health and load
15 infrastructure 1202 wants to ensure that consolidated health and load caches 1208
16 have actually been updated to the new directive state of draining.

17 To verify that consolidated health and load caches 1208 of load balancing
18 units 106 have been updated to a new state directive, health and load infrastructure
19 1202 sends a postulate table state message 1512 to load balancing units 106.
20 Postulate table state message 1512 is sent some time (e.g., a predetermined delay
21 period) after transmission of a row change message 1506 indicating that the state
22 directive is to be changed. The postulate table state message 1512, in this
23 example, indicates that the table state should be draining. As indicated by the
24 dashed arrow 1518, a load balancing unit 106 responds to this postulate table state
25

1 message 1512 if its consolidated health and load cache 1208 differs from the
2 postulated state directive.

3 If the directive in consolidated health and load cache 1208 does differ from
4 the postulated state directive, then that load balancing unit 106 sends a postulate
5 wrong message 1514 to the health and load infrastructure 1202 of the host 108 that
6 issued the postulate table state message 1512. This health and load infrastructure
7 1202 then periodically resends postulate table state message 1512 until no further
8 postulate wrong messages 1514 are received from consolidated health and load
9 caches 1208. At that point, health and load infrastructure 1202 sends a row change
10 message 1506 with the new current load balancing state. In this sense,
11 consolidated health and load caches 1208 are the definitive determiners of the
12 current load balancing state, and health and load infrastructure 1202 is the
13 definitive determiner of the target load balancing state.

14 FIG. 16 illustrates an exemplary message transmission scheme for the
15 communications that are illustrated in FIG. 12 between hosts 108 and load
16 balancing units 106. The exemplary message transmission scheme can reduce the
17 bandwidth consumed by heartbeat messages 1502 on communication linkage
18 1210. The message transmission scheme of FIG. 16 is particularly adapted to
19 heartbeat messages 1502, but it may also be utilized for other messages of
20 message protocol 1500.

21 A group of hosts 108(1), 108(2), 108(3) ... 108(11), and 108(12) are
22 illustrated along with load balancing units 106(1), 106(2) ... 106(u). Each line
23 represents membership linkage or inclusion among the group of hosts 108(1, 2 ...
24 12). The group of hosts 108(1, 2 ... 12) form a membership of nodes that work
25 together to propagate heartbeat information to load balancing units 106. Although

1 twelve hosts are shown, more or fewer may be part of any given group of hosts.
 2 Also, a total set of hosts 108 that are being served by a load balancing
 3 infrastructure 106 may be divided into one, two, three, or more groups of hosts.

4 In a described implementation, the membership of nodes for group of hosts
 5 108(1, 2 ... 12) elect a leader that is responsible for transmitting heartbeat
 6 messages 1502 to load balancing units 106. Each (non-leading) host 108 in group
 7 of hosts 108(1, 2 ... 12) sends its heartbeat messages 1502 to the elected leader.
 8 Host 108(4) is the elected leader in this example.

9 With the membership of nodes, heartbeat information for each host 108 in
 10 group of hosts 108(1, 2 ... 12) propagates to the group leader host 108(4). Host
 11 108(4) collects the heartbeat information and consolidates it into a consolidated
 12 heartbeat message 1602. Consolidated heartbeat messages 1602(1), 1602(2) ...
 13 1602(u) are then sent to respective load balancing units 106(1), 106(2) ... 106(u).
 14 These consolidated heartbeat messages 1602 may optionally be compressed to
 15 further reduce bandwidth consumption.

16 As another alternative, the leader host 108(4) may only forward changes in
 17 group membership to consolidated health and load caches 1208. In other words,
 18 in this mode, consolidated health and load caches 1208 deal primarily if not solely
 19 with state changes to membership. It is the responsibility of the leader host 108(4)
 20 to ensure that the first hello is forwarded when a host 108 comes online and that a
 21 goodbye message 1504 gets sent when that host 108 goes offline. Additionally, a
 22 host 108 can periodically specify that a heartbeat message 1502 is to be
 23 "forwarded". This indicates to the leader host 108(4) to send it to consolidated
 24 health and load caches 1208 even though it does not represent a membership
 25 change.

1 Heartbeat messages 1502 (including consolidated heartbeat messages 1602)
2 are used by load balancing units 106 when their consolidated health and load
3 caches 1208 are unsynchronized with health and load tables 1204. This lack of
4 synchronization may arise, for example, from a crash or other failure of
5 consolidated health and load cache 1208 and/or of load balancing unit 106. As
6 described above, each heartbeat message 1502 includes error checking data that is
7 usable to verify equivalency between a consolidated health and load cache 1208
8 and health and load tables 1204. If non-equivalency is discovered with regard to a
9 particular host 108 and/or an application 316 thereof, the DNS name of the
10 particular host 108 is acquired from the heartbeat messages 1502.

11 The DNS name is used by consolidated health and load cache 1208 to send
12 a get table snapshot message 1508 to the particular host 108 in order to get
13 updated health and load information 1206 in the form of a send table snapshot
14 message 1510. A different or the same get table snapshot message 1508 is sent to
15 each host 108 for which non-equivalency is discovered. Eventually, the health and
16 load information 1206 in the consolidated health and load cache 1208 is equivalent
17 to the health and load information 1206 in health and load tables 1204 as verifiable
18 by new heartbeat messages 1502. In this manner, a failed consolidated health and
19 load cache 1208 can be bootstrapped back into operation without manual oversight
20 using message protocol 1500 and an equivalency-checking scheme.

21 FIG. 17A and FIG. 17B illustrate exemplary health and load information
22 proxy storage scenarios for health and load tables 1204 and for consolidated health
23 and load caches 1208, respectively. In implementations described above with
24 reference to FIGS. 12-16, hosts 108 include health and load infrastructure 1202.
25

1 However, other implementations may entail hosts that do not include health and
2 load infrastructure 1202.

3 For example, a host may be running a version of application(s) and/or an
4 operating system for which health and load infrastructure is either not
5 implemented or for policy reasons may not be installed on the host. Consequently,
6 these types of hosts do not have health and load infrastructure 1202 executing
7 thereon. Host 1702 is such a host that does not execute health and load
8 infrastructure 1202. Nevertheless, host 1702 can utilize a health and load
9 infrastructure 1202 that is executing on one or more proxies, such as proxy 1704.

10 Proxy 1704 has resident thereat and executing thereon a health and load
11 infrastructure 1202, which includes a health and load table 1204. Host 1702 can
12 use the functionality of health and load infrastructure 1202 by providing health
13 and load information 1206 to health and load table 1204 for applications that are
14 running on host 1702. Alternatively, proxy 1704 can deduce health and load on
15 host 1702 by performing external monitoring actions. Proxy 1704 is illustrated as
16 proxy 1704(1) and 1704(2) for redundancy and the resulting high availability.

17 In implementations described above with reference to FIGS. 12-16 and
18 below with reference to FIG. 18, load balancing is effectuated with load balancing
19 units 106 that include consolidated health and load caches 1208. However, other
20 implementations may entail load balancing that does not include consolidated
21 health and load caches 1208.

22 For example, load balancing may be effectuated by monolithic load
23 balancing hardware or other load balancing infrastructure that does not and/or
24 cannot store or otherwise include a consolidated health and load cache 1208. Load
25 balancer 1706 reflects such a load balancing device or devices that do not have a

1 consolidated health and load cache 1208. Nevertheless, load balancer 1706 can
2 utilize a consolidated health and load cache 1208 that exists on one or more
3 proxies, such as proxy 1708.

4 Proxy 1708 includes a consolidated health and load cache 1208, which
5 stores health and load information 1206 for hosted applications being serviced by
6 load balancer 1706. Load balancer 1706 can use the health and load information
7 1206 of consolidated health and load cache 1208 when performing load balancing
8 functions by accessing such information using application programming interfaces
9 (APIs) native to and supported by load balancer 1706. Alternatively, consolidated
10 health and load cache 1208 can invoke APIs to push health and load information
11 1206, including directives, to load balancer 1706. Proxy 1708 is illustrated as
12 proxy 1708(1) and 1708(2) for redundancy and the resulting high availability.

13 FIG 18 illustrates an exemplary target application endpoint allotment
14 procedure that involves a classifier 304 and a health and load handler 314 of a load
15 balancing unit 106. After health and load handler 314 has acquired a consolidated
16 health and load cache 1208, health and load information 1206 thereof is utilized in
17 the selection of application endpoints for new requests/connections.

18 As described above with reference to FIG. 13B, consolidated health and
19 load cache 1208 includes cached health and load information 1206 for multiple
20 hosts 108. To facilitate the creation and updating of consolidated health and load
21 cache 1208 from health and load information 1206 that originates from multiple
22 hosts 108, the health and load information 1206 therein is organized so that it may
23 be accessed by identifier of each host 108. However, the health and load
24 information 1206 therein is also organized such that it can be accessed by type of
25 application 316 in order to facilitate application endpoint selection.

X

1 In other words, health and load handler 314 is capable of accessing health
2 and load information 1206 on a per-application 316 basis across health and load
3 information 1206 for multiple hosts 108. Once health and load information 1206
4 for a given application 316 has been accessed for each host 108, allocation of
5 incoming connection requests may be performed in accordance with such health
6 and load information 1206. For example, possible endpoints for the given
7 application 316 may be allocated to incoming connection requests by selection of
8 the endpoints of the given application 316 with consideration of available relative
9 load capacity among healthy endpoints for the given application 316.

10 In a described implementation, classifier 304 makes a target application
11 endpoint allotment request 1802 to health and load handler 314. As illustrated,
12 target application endpoint allotment request 1802 includes (i) a virtual IP address
13 and port, (ii) a protocol, and (iii) protocol-specification information. Target
14 application endpoint allotment request 1802 therefore identifies a type of
15 application 316 to which incoming connection requests are directed.

16 Health and load handler 314 receives target application endpoint allotment
17 request 1802 and selects at least one physical endpoint corresponding to the
18 identified type of application 316 using any one or more of many selection
19 mechanisms. To reduce latency, health and load handler 314 selects an allotment
20 of application endpoints to be used over a number of incoming connection
21 requests. This allotment is provided from health and load handler 314 to classifier
22 304 using target application endpoint allotment response 1804. As illustrated,
23 target application endpoint allotment response 1804 includes an allotment of
24 physical IP addresses and ports (such as endpoints IP1, IP2, and IP3) for the
25 identified type of application 316.

1 The allotment for target application endpoint allotment response 1804 may
2 be completed using one or more allotment schemes. By way of example, a token
3 allotment scheme 1806 and a percentage allotment scheme 1808 are illustrated.
4 Token allotment scheme 1806 is a unit-based allotment scheme, and percentage
5 allotment scheme 1808 is a time-based allotment scheme.

6 Token allotment scheme 1806 allocates tokens for each healthy endpoint
7 IP1, IP2, and IP3 responsive to their relative load and capacity ratios. For the
8 example as illustrated, of the total available capacity, IP1 has 40% of the available
9 capacity, IP2 has 35% of the available capacity, and IP3 has 25% of the available
10 capacity. Thus, the total number of tokens is divided along these percentages. The
11 total number of tokens may be provided as part of target application endpoint
12 allotment request 1802 or determined by health and load handler 314.

13 Any value for the total number of tokens may be used, such as 10, 45, 100,
14 250, 637, 1000, and so forth. This value may be set in dependence on the number
15 of connection requests per second and the speed/frequency at which application
16 health and/or load is changing. Classifier 304 "uses up"/consumes one token
17 when responding to each connection request with an application endpoint
18 allocation until the tokens are exhausted; classifier 304 then requests another token
19 allotment using target application endpoint allotment request 1802.

20 Percentage allotment scheme 1808 determines available relative capacity in
21 a similar manner. However, instead of tokens, these determined available relative
22 capacities per application endpoint are provided to classifier 304 along with a
23 duration timer 1810. Classifier 304 allocates target application endpoints to
24 incoming connection requests in accordance with these available relative capacity
25 percentages until expiration of duration timer 1810.

21

61

1 For percentage allotment scheme 1808, classifier 304 maintains a running
2 record of application endpoint allocations to adhere to the allotted percentages and
3 keeps track of time for duration timer 1810. When the timer expires, classifier 304
4 then requests another percentage allotment using target application endpoint
5 allotment request 1802.

6 It should be noted that token allotment scheme 1806 can also use a time
7 limit. If allotted tokens are too old, they should be discarded and new ones
8 acquired. Otherwise, classifier 304 may consume stale tokens that were
9 previously allocated based on health and load information that is currently too
10 outdated. Use of application endpoint allotments by classifier 304 is described
11 further below in the section entitled "Exemplary Classifying, Forwarding, and
12 Request Routing".

13 Exemplary Session Tracking

14 This section describes how host status information, such as session
15 information, may be collected for and utilized in network load balancing. This
16 section primarily references FIGS. 19-24 and illuminates session affinity
17 preservation functionality such as that provided by session tracker 308 (of FIG. 3).
18 As described above with reference to FIGS. 1-3, each host 108 hosts one or more
19 applications 316 that provide service(s) to clients 102. Session tracker 308 utilizes
20 session information that relates to contexts for the connections established
21 between applications 316 and clients 102 for certain described implementations of
22 network load balancing.

23 FIG. 19 illustrates an exemplary network load balancing approach that
24 involves session information 1902. At connection [1], client 102(1) is shown
25 making a new connection with host 108(2) via load balancing infrastructure 106.

1 Load balancing infrastructure 106 may be comprised of one or more load
2 balancing units 106. When the connection request arrives at load balancing
3 infrastructure 106, the request is typically routed to a host 108 using network load
4 balancing functionality responsive to health and/or load information of hosts 108
5 and/or applications 316 (not explicitly shown in FIG. 19) thereof.

6 When connection [1] is made, a session is established between client 102(1)
7 and the servicing application 316, which is on host 108(2) in this example. The
8 session provides a context for the communication exchange between client 102(1)
9 and host 108(2). The information for the session context is stored at host 108(2).
10 When connection [1] is completed, the session context may not be used again. On
11 the other hand, the session context may be useful again if client 102(1) attempts to
12 initiate another connection with hosts 108 for the service provided by application
13 316. If this other connection is not routed to the same host 108(2) that stores that
14 session context, then client 102(1) has to establish a new session context, which
15 can be time consuming, data/processing intensive, and/or frustrating to the human
16 user of client 102(1). With health and/or load information-based network load
17 balancing, there is no likelihood greater than random chance that the second
18 connection will be routed to 108(2).

19 However, if load balancing infrastructure 106 has access to a mapping
20 between session information and hosts 108, load balancing infrastructure 106 can
21 route connection requests that relate to previously established sessions to the
22 appropriate host 108. Some session information may be inferred from the contents
23 of packets flowing through load balancing infrastructure 106. However, this
24 approach is imprecise and haphazard for a number of reasons. First, session
25 establishment and termination is merely inferred. Second, some sessions are not

71

1 "officially" terminated with an appropriate indication that is included in a packet.
2 For example, some sessions simply time out. Third, packets being transmitted
3 from host 108(2) to client 102(1) may take a path that does not include load
4 balancing infrastructure 106, which precludes any snooping of such packets by
5 load balancing infrastructure 106 for session information.

6 As shown in FIG. 19, hosts 108 provide session information (SI) 1902 to
7 load balancing infrastructure 106. Using session information 1902 from hosts
8 108, a session affinity preserver 1904 can preserve the affinity between an
9 established session and the host 108 on which the session was established.
10 Session information 1902 includes a linkage between or a mapping from each
11 session established between a client 102 and a particular host 108 to that particular
12 host 108. This mapping is accessible to session affinity preserver 1904 as part of
13 host-session information mapping 1906. More-specific examples of session
14 information 1902 are provided below especially with reference to FIGS. 20, 22,
15 23A, and 23B.

16 In certain described implementations for session tracking, the logical nature
17 of clients 102 is pertinent. As noted above with reference to FIG. 1, a client 102
18 may be a specific device and/or a specific user of a device. Consequently, session
19 affinity for a user client 102 that is accessing hosts 108 from different devices can
20 still be preserved. Session continuations using session information 1902 can
21 therefore still be effectuated in proxy scenarios (e.g., those of some internet
22 service providers (ISPs)).

23 Continuing with the connection [1] example, the session established at host
24 108(2) is provided to load balancing infrastructure 106 as session information
25 1902. Specifically, a linkage/mapping between (i) the session context of client

1 102(1) and host 108(2) and (ii) an identifier for host 108(2) is created at host-
2 session information mapping 1906. When a connection request for connection [2]
3 subsequently arrives for the same session context, session affinity preserver 1904
4 locates this session context in host-session information mapping 1906 and
5 ascertains that host 108(2) is associated with this session context from the
6 linkage/mapping.

7 Responsive to the mapping of host 108(2) to the requested session context
8 as ascertained by session affinity preserver 1904 from host-session information
9 mapping 1906, connection [2] is routed to host 108(2). In this sense, preserving
10 session affinity is a higher priority for load balancing infrastructure 106 than
11 application health and load-based network load balancing decisions. However,
12 health and/or load may be a more important network load balancing factor than
13 session tracking when, for example, loading is extremely heavy or when the
14 session-relevant application and/or host is in a failed condition.

15 Many types of connections may be session-related. Examples include: a
16 TCP connection, a transport layer security (TLS)/SSL session, a PPTP session, an
17 IPSec/L2TP session, an ISA session, an HTTP cookie-based session, a Terminal
18 Server session, an administrator-defined session, and so forth. By way of
19 clarification, a TCP connection is considered to be a session of TCP packets. Also,
20 a model for defining sessions by an administrator may be enumerated and
21 supported. Furthermore, client IP-address-based sessions that are delineated by
22 timeouts may also be supported. This is relatively non-intelligent session support,
23 but is expected by some users.

24 A connection request from a client 102 varies by the type of desired
25 session. For example, for sessions of type "TCP connection", the connection

1 request comprises a TCP packet. For sessions of type "SSL session", the
2 connection request comprises a TCP connection. Other such connection requests
3 correspond to other session types. These examples also show how there may be
4 session layers. At a lower session level, a session context for a TCP connection
5 may include a TCP 4-tuple, a session number, the number of bytes sent/received,
6 and so forth. At a higher session level, a session context for an SSL session may
7 include a 32-byte session ID, a public key of the client 102 that is provided to the
8 host 108, and so forth.

9 FIG 20 illustrates an exemplary network load balancing approach that
10 involves communicating session information using notifications 2006 and
11 messages 2008. Multiple load balancing units 106(1), 106(2) ... 106(u) and
12 multiple hosts 108(1), 108(2) ... 108(n) are shown. Each respective host 108(1),
13 108(2) ... 108(n) includes one or more respective applications 316(1), 316(2) ...
14 316(n) which are resident thereat and executing thereon. Notifications 2006 are
15 used to provide session information from applications 316, and messages 2008 are
16 used to provide session information from hosts 108 to load balancing units 106.

17 As illustrated, each respective host 108(1), 108(2) ... 108(n) includes
18 respective session tracking infrastructure (STI) 2002(1), 2002(2) ... 2002(n).
19 Each respective session tracking infrastructure 2002(1), 2002(2) ... 2002(n)
20 includes a respective session table 2014(1), 2014(2) ... 2014(n) (although only
21 session table 2014(1) is explicitly illustrated in FIG 19).

22 Each respective load balancing unit 106(1), 106(2) ... 106(u) includes
23 respective traffic routing functionality (TRF) 2012(1), 2012(2) ... 2012(u). Traffic
24 routing functionality 2012 may comprise, for example, classifying and/or
25 requesting routing functionality, such as that provided by classifier 304 and

1 request router 306, respectively. Distributed across load balancing units 106(1),
2 106(2) ... 106(u) is a distributed session tracking manager 2010.

3 In a described implementation, traffic routing functionality 2012 and
4 distributed session tracking manager 2010 are part of load balancing infrastructure
5 106. Session tracking infrastructure 2002 may also be (e.g., a remote) part of load
6 balancing infrastructure 106.

7 An API 2004 is employed to provide session information from applications
8 316 to session tracking infrastructure 2002. Using API 2004, applications 316 are
9 empowered to notify session tracking infrastructure 2002 of session information,
10 including various changes thereto. More specifically, each application 316 is
11 capable of providing, and session tracking infrastructure 2002 is capable of
12 accepting, notifications 2006.

13 A notification that a session has been established (or session establishment
14 notification 2006(E)) is provided from application 316 when a session is newly
15 established or opened. Session establishment notification 2006(E) includes a
16 session identifier and optionally an identifier of application 316. A notification
17 that a session has been terminated (or session termination notification 2006(T)) is
18 provided from application 316 when a session is terminated or closed. Session
19 termination notification 2006(T) also includes the session identifier and optionally
20 the identifier of application 316.

21 When session tracking infrastructure 2002 accepts a session establishment
22 notification 2006(E), it inserts an entry in session table 2014 for the new session.
23 An exemplary session table 2014 is described further below with reference to FIG.
24 23A. When session tracking infrastructure 2002 accepts a session termination
25 notification 2006(T), it removes the entry in session table 2014 for the old session.

75

1 Session table 2014(1) is the authoritative source for session information
2 1902 with respect to applications 316(1) on host 108(1). There is generally too
3 much latency, however, to require traffic routing functionality 2012 to contact
4 hosts 108 for access to session tables 2014 upon receipt of each incoming
5 connection request having a session reference. Session information 1902 is
6 therefore cached at load balancing units 106.

7 At load balancing units 106, distributed session tracking manager 2010
8 caches session information 1902 as part of its session tracking management
9 responsibilities. Generally, distributed session tracking manager 2010 is a
10 distributed application and/or virtual service that resides partially on each load
11 balancing unit 106. For each logical session, distributed session tracking manager
12 2010 keeps at least one cached copy of session information therefor in a reliable
13 and scalable manner that may be quickly utilized for routing traffic as incoming
14 connection requests that have a session reference are received by load balancing
15 infrastructure 106.

16 Communications between hosts 108 and load balancing units 106 are
17 effectuated with a reliable protocol that ensures that messages 2008 sent from a
18 host 108 arrive at the intended load balancing unit 106. Each host 108 is bound to
19 at least one specific load balancing unit 106 that is the intended load balancing
20 unit 106 for messages 2008. This binding is created by assigning an IP address of
21 a specific load balancing unit 106 to each host 108 for sending session-tracking
22 messages 2008 between session tracking infrastructure 2002 and distributed
23 session tracking manager 2010. To facilitate high availability of load balancing
24 infrastructure 106, if a load balancing unit 106 fails, another load balancing unit
25 106 assumes the IP address of the failed load balancing unit 106. Failure detection

1 for IP address assumption may be accomplished using a heartbeat or another
2 aliveness monitoring scheme.

3 Thus, messages 2008 communicate session information 1902 from session
4 tracking infrastructure 2002 to distributed session tracking manager 2010. For
5 example, when session tracking infrastructure 2002 accepts a session
6 establishment notification 2006(E), it also sends a session up message 2008(U) to
7 distributed session tracking manager 2010. Session up message 2008(U) includes
8 the session identifier, a host identifier, and optionally other information. Contents
9 for a session up message 2008(U) are described further below with reference to
10 FIG. 23B with respect to information that may be stored for each session by an
11 implementation of distributed session tracking manager 2010. When session
12 tracking infrastructure 2002 accepts a session termination notification 2006(T), it
13 also sends a session down message 2008(D) to distributed session tracking
14 manager 2010. Messages 2008 can be sent before, during, or after session
15 tracking infrastructure 2002 appropriately modifies session table 2014 in response
16 to notifications 2006.

17 FIG. 21 is a flow diagram 2100 that illustrates an exemplary method for
18 network load balancing that involves communicating session information using
19 notifications and messages. Flow diagram 2100 includes fifteen blocks 2102-
20 2130. Although the actions of flow diagram 2100 may be performed in other
21 environments and with a variety of software schemes, FIGS. 1-3 and 19-20 are
22 used in particular to illustrate certain aspects and examples of the method.

23 For example, the actions of four blocks 2102-2104 and 2118-2120 are
24 performed by an application 316, the actions of six blocks 2106-2110 and 2122-
25 2126 are performed by session tracking infrastructure 2002, and the actions of five

1 blocks 2112-2116 and 2128-2130 are performed by a distributed session tracking
2 manager 2010. The actions of eight of these blocks 2102-2116 are primarily
3 directed to opening a session, and the actions of seven of these blocks 2118-2130
4 are primarily directed to closing a session.

5 At block 2102, a session is opened. For example, application 316 may
6 open a session with a client 102. At block 2104, a session establishment
7 notification is provided. For example, application 316 may provide a session
8 establishment notification 2006(E) to session tracking infrastructure 2002 using
9 API 2004 as a consequence of and/or in conjunction with opening the session.

10 At block 2106, the session establishment notification is accepted. For
11 example, session tracking infrastructure 2002 may accept session establishment
12 notification 2006(E) from application 316 in accordance with API 2004. At block
13 2108, an entry in a session table is inserted. For example, session tracking
14 infrastructure 2002 may insert an entry in session table 2014 for the opened
15 session. Examples of such insertion are described further below especially with
16 reference to FIG. 23A. At block 2110, a session up message is sent. For example,
17 session tracking infrastructure 2002 may send a session up message 2008(U) to
18 distributed session tracking manager 2010 using a reliable communication
19 protocol.

20 At block 2112, the session up message is received. For example,
21 distributed session tracking manager 2010 may receive session up message
22 2008(U) from session tracking infrastructure 2002 in accordance with the reliable
23 communication protocol. At block 2114, a session information entry is created.
24 For example, distributed session tracking manager 2010 may create a session
25 information entry for cached session information 1902 at one or more load

76

balancing units 106. Examples of such creating and subsequent adding are described further below especially with reference to FIGS. 22 and 23B.

At block 2116, network traffic is routed with the session information. For example, traffic routing functionality 2012 in conjunction with distributed session tracking manager 2010 may use cached session information 1902, including the created session information entry, to route incoming connection requests that have a session reference. An example of such traffic routing is described further below especially with reference to FIG. 24. Additional examples are described below in the section entitled "Exemplary Classifying, Forwarding, and Request Routing".

At block 2118, the session is closed. For example, application 316 may close the session with client 102. At block 2120, a session termination notification is provided. For example, application 316 may provide a session termination notification 2006(T) to session tracking infrastructure 2002 using API 2004 as a consequence of and/or in conjunction with closing the session.

At block 2122, the session termination notification is accepted. For example, session tracking infrastructure 2002 may accept session termination notification 2006(T) from application 316 in accordance with API 2004. At block 2124, the entry in the session table is removed. For example, session tracking infrastructure 2002 may remove the entry in session table 2014 for the closed session. At block 2126, a session down message is sent. For example, session tracking infrastructure 2002 may send a session down message 2008(D) to distributed session tracking manager 2010 using the reliable communication protocol.

At block 2128, the session down message is received. For example, distributed session tracking manager 2010 may receive session down message



1 2008(D) from session tracking infrastructure 2002 in accordance with the reliable
2 communication protocol. At block 2130, the session information entry is
3 destroyed. For example, distributed session tracking manager 2010 may destroy
4 the session information entry at the cached session information 1902 at any load
5 balancing units 106 that have the session information entry. Examples of such
6 destroying and subsequent deleting are described further below especially with
7 reference to FIGS. 22 and 23B.

8 FIG. 22 illustrates an exemplary approach to managing session information
9 at multiple load balancing units 106. Each respective load balancing unit 106(1),
10 106(2) ... 106(u) includes a respective part 2202(1), 2202(2) ... 2202(u) of a
11 distributed atom manager (DAM) 2202. DAM 2202 is an exemplary
12 implementation of distributed session tracking manager 2010. Each respective
13 DAM portion 2202(1), 2202(2) ... 2202(u) includes a respective part 2206(1),
14 2206(2) ... 2206(u) of a DAM table (DAMT) 2206.

15 DAM 2202 is a distributed application or virtual service that manages
16 session information 1902 in a reliable and scalable manner so that traffic routing
17 functionality 2012 can use it to preserve session affinity. For example, traffic
18 routing functionality 2012 can access DAM 2202 using an API (not specifically
19 shown) to search or have searched DAMT 2206. Function calls 2204, operation of
20 DAM 2202, and other aspects of FIG. 22 are described further below after the
21 description of FIGS. 23A and 23B.

22 FIG. 23A is an exemplary session table 2014 as illustrated in FIG. 20.
23 Session table 2014 includes "v" entries 2302(1), 2302(2) ... 2302(v). Each entry
24 2302 is inserted by session tracking infrastructure 2002 responsive to a session
25 establishment notification 2006(E) that is accepted from an application 316. Each

f0

entry 2302 is removed by session tracking infrastructure 2002 responsive to a session termination notification 2006(T) that is accepted from application 316.

As described above, each session establishment notification 2006(E) includes a session identifier and optionally an identifier of application 316. Each respective entry 2302(1), 2302(2) ... 2302(v) in session table 2014 includes respective fields of (i) session identifier 2302(1I), 2302(2I) ... 2302(vI) and (ii) session type and/or application 2302(1T), 2302(2T) ... 2302(vT).

Session type and/or application 2302(T) may be "TCP", "IPSEC", "Terminal Server," "HTTP-cookie", an application type as noted above, and so forth. Session identifier 2302(I) may be "<source IP address, source TCP port, destination IP address, destination TCP port >", "Client IP = 172.30.189.122", "User = 'joe_user'", "Cookie = '{b7595cc9-e68b-4eb0-9bfl-bb717b31d447}'", another e.g. application-specific identification for a session, and so forth. For TCP connection/session types, session identifier 2302(I) may alternatively be a canonical version of the TCP 4-tuple (for IPv4 or IPv6). Other values for the fields of session identifier 2302(I) and application/session type 2302(T) may alternatively be used.

FIG. 23B is an exemplary distributed atom manager (DAM) table (DAMT) 2206 as illustrated in FIG. 22. DAM table 2206 includes "w" entries 2304(1), 2304(2) ... 2304(w). Each session information entry 2304 is created by DAM 2202 responsive to a session up message 2008(U) that is received from session tracking infrastructure 2002. Each session information entry 2304 is destroyed responsive to a session down message 2008(D) that is received from session tracking infrastructure 2002. As described further below, session information

1 entries 2304 of DAM tables 2206 may actually be manipulated by DAM 2202
2 using function calls 2204.

3 As described above, session up message 2008(U) includes the session
4 identifier, a host identifier, and optionally other information. Each respective
5 session information entry 2304(1), 2304(2) ... 2304(w) in DAM table 2206
6 includes respective fields of (i) key 2304(1K), 2304(2K) ... 2304(wK), (ii) data
7 2304(1D), 2304(2D) ... 2304(wD), and (iii) metadata 2304(1M), 2304(2M) ...
8 2304(wM). For example, values for key 2304(K) fields may be alphanumeric
9 strings, and values for data 2304(D) fields may be binary bits. Values for key
10 2304(K) may be binary bits, too.

11 Key 2304(K) may correspond to the session identifier 2302(I). Data
12 2304(D) may correspond to the host identifier, such as a network address of the
13 host 108 on which the session context exists. Metadata 2304(M) may correspond
14 to other, optional information. Examples of such metadata 2304(M) include data
15 that is used internally by DAM 2202 to resolve atom collisions and to track atom
16 aliveness (e.g., via a time-out mechanism). (This characterization of entries 2304
17 as being atomic is described more fully in the following paragraph.) More
18 specifically, metadata 2304(M) includes, among other things, the identity of the
19 entity (e.g., the instance of traffic routing functionality 2012) that added the
20 session information entry 2304 to the DAM table 2206.

21 In a described implementation, each session information entry 2304 is
22 atomic in the sense that DAM 2202 may add, delete, copy, etc. the entries 2304 as
23 a whole, but DAM 2202 does not ordinarily modify a portion of any whole entry
24 2304. Thus, atomic entries 2304 are added, deleted, copied, otherwise
25

1 manipulated, etc. across DAM tables 2206 by DAM 2202 in order to implement
2 availability and scalability for a session affinity preservation implementation.

3 Function calls 2204 (of FIG. 22) are usable by DAM 2202 to manipulate the
4 atomic entries 2304 of DAM table 2206. Function calls 2204 may be
5 communicated from one load balancing unit 106 to one or more other load
6 balancing units 106 in a point-to-point or a multicast manner. These function calls
7 include add atom 2204(A), delete atom 2204(D), query atom 2204(Q), and return
8 atom 2204(R).

9 Add atom 2204(A) takes the form AddAtom(key, data) and is used to add
10 an atomic entry 2304 to one or more DAM tables 2206. Hence, an add atom
11 2204(A) function call may be formulated as AddAtom(<session identifier>, host
12 IP address). Delete atom 2204(D) takes the form DeleteAtom(key) and is used to
13 delete an atomic entry 2304 at one or more DAM tables 2206. Delete atom
14 2204(D) function calls may be directed at those DAM tables 2206 known to have
15 a copy of the session that is identified by the key 2304(K) or may be multicast to
16 all DAM tables 2206 to ensure that any copies are deleted.

17 Query atom 2204(Q) takes the form QueryAtom(key) and is used by a
18 particular DAM portion 2202 when a session identifier as referenced by an
19 incoming connection request is not located in the particular local DAM table 2206
20 of the particular DAM portion 2202. Query atom 2204(Q) function calls are sent
21 to one or more (including possibly all) other DAM portions 2202. In response,
22 each other DAM portion 2202 checks its local DAM table 2206 for the
23 key/session identifier. If the key is located by another DAM portion 2202, this
24 other DAM portion 2202 replies with a return atom 2204(R).
25

φ

1 Return atom 2204(R) takes the form ReturnAtom(key, data) and is used to
2 reply to a query atom 2204(Q) function call. Return atom 2204(R) function calls
3 are used when a DAM portion 2202 has a requested atomic entry 2304 in its local
4 DAM table 2206 as identified by a key 2304(K) specified in the query atom
5 2204(Q) function call. Return atom 2204(R) function calls may be directed back
6 to the DAM portion 2202 that issued the query atom 2204(Q) function call.

7 Add atom 2204(A) function calls are used in response to session up
8 messages 2008(U) and/or to replicate an atomic entry 2304 to one or more other
9 DAM tables 2206. Such replication may be for redundancy and/or scalability.

10 Delete atom 2204(D) function calls are used in response to session down
11 messages 2008(D) and may also be sent to one or more other DAM tables 2206.
12 After an atomic entry 2304 is deleted, the atomic entry 2304 may enter a "zombie"
13 state such that it remains with DAM 2202, and optionally so that it is actually still
14 stored with DAM table 2206 with a zombie indication in the metadata 2304(M)
15 field of the atomic entry 2304.

16 Thus, once an atomic entry 2304 is deleted, it may stay on in DAM 2202
17 and DAM table 2206 in a zombie state so that packets for this (now dead and
18 closed) session are directed to the host 108 of the session context for proper,
19 protocol-specific treatment. For example, TCP packets received after a TCP
20 connection has been torn down are directed to the host 108 that terminated the
21 connection. This host 108 can respond appropriately -- perhaps by sending an RST
22 or by resending a FIN-ACK. The time the atomic entry 2304 spends in this
23 zombie state matches (as closely as reasonably possible) the protocol-specific dead
24 time of the reliable communication protocol that is employed.
25

1 A query atom 2204(Q) function call is used to attain an atomic entry 2304
2 when a first load balancing unit 106 receives an incoming connection request that
3 references a session that is not stored in the local DAM table 2206 of the DAM
4 2202 of the first load balancing unit 106. It should be noted that other DAM
5 portions 2202 may be queried simultaneously in a broadcast query atom 2204(Q)
6 function call or sequentially until a positive return atom 2204(R) function call is
7 received.

8 A return atom 2204(R) function call is used by a DAM portion 2202 of a
9 second load balancing unit 106 to provide an atomic entry 2304 to the DAM
10 portion 2202 of the first load balancing unit 106, where the atomic entry 2304 has
11 a key 2304(K) that is specified by the key/session identifier in a query atom
12 2204(Q) function call, which was previously issued by the DAM portion 2202 of
13 the first load balancing unit 106. It should be noted that other components, such
14 as traffic routing functionality 2012, may also be capable of calling functions
15 2204, especially a query atom 2204(Q) function call, in accordance with an API or
16 similar.

17 DAM portions 2202 and DAM tables 2206 may be organized and managed
18 in a myriad of manners. Exemplary manners relate to replication/redundancy,
19 local caching upon acquisition, hashing for location selection, and so forth. Zero,
20 one, two, or more levels of replication up to full replication may be employed.
21 With a zero level of replication, each atomic entry 2304 is stored at the DAM 2202
22 that receives a session up message 2008(U) therefor without replication to other
23 DAM portions 2202.

24 With a first level of replication, each atomic entry 2304 is stored at the
25 DAM 2202 that receives a session up message 2008(U) therefor, and it is also

added (copied) to one other DAM portion 2202 using an add atom 2204(A) function call. This handles one level of failure for a load balancing unit 106. Similarly, with a second level of replication, each atomic entry 2304 is stored at the DAM 2202 that receives a session up message 2008(U) therefor, and it is also added to two other DAM portions 2202. Generally, the one, two, etc. other DAM portions 2202 to which a given DAM portion 2202 copies atomic entries 2304 is predetermined or selected at random. Third, fourth, etc. levels of replication may also be employed.

Furthermore, full replication may be employed by having each atomic entry 2304 that is stored at the DAM 2202 that receives a session up message 2008(U) therefor also being added to every other DAM portion 2202. Several factors are impacted by selection of the replication level: As the replication level increases, availability increases and latency decreases. On the other hand, network traffic and memory usage both increase as the replication level increases.

When full replication is not employed, local caching upon acquisition may be. For example, when a DAM portion 2202 does not locate a referenced session identifier in its part of DAM table 2206, the DAM portion 2202 issues a query atom 2204(Q) function call to attain the atomic entry 2304 associated with the referenced session identifier via a return atom 2204(R) function call. Instead of jettisoning the attained atomic entry 2304 after use thereof, the DAM portion 2202 caches the attained atomic entry 2304 in its part of DAM table 2206. This option offers a tradeoff between the above-enumerated factors.

As another option when full replication is not employed, hashing for location selection may be. The first atomic entry 2304 for a session is stored at the DAM portion 2202 that receives the session up message 2008(U). Replicated

36

1 copy or copies are sent via add atom 2204(A) function calls to specific DAM
2 portion(s) 2202 using a hashing function. Of a total range of possible hash values,
3 each DAM portion 2202 is assigned a subset thereof. Each session identifier is
4 hashed using some hashing function to arrive at a hashing value. This hashing
5 value is mapped to the assigned DAM portion(s) 2202. The DAM portion 2202
6 that first added the atomic entry 2304 then replicates the atomic entry 2304 to the
7 assigned DAM portion(s) 2202.

8 With hashing for location selection, at least one DAM portion 2202 that has
9 a desired atomic entry 2304 locally cached at its DAM table 2206 is knowable
10 from the session identifier. A query atom 2204(Q) function call can therefore be
11 directed to the known DAM portion(s) 2202. This usually reduces network traffic
12 and/or latency.

13 This hashing for location selection may be used with one, two, three, or
14 more levels of replication with each range of hashing values mapping to one, two,
15 three, etc. different DAM portions 2202, respectively. Additionally, hashing for
16 location selection may be used with local caching upon acquisition.

17 FIG. 24 is a flow diagram 2400 that illustrates an exemplary method for
18 managing session information at multiple load balancing units. Flow diagram
19 2400 includes eight blocks 2402-2416. Although the actions of flow diagram
20 2400 may be performed in other environments and with a variety of software
21 schemes, FIGS. 1-3, 19, 20, 22, and 23B are used in particular to illustrate certain
22 aspects and examples of the method.

23 At block 2402, an incoming connection request with a session reference is
24 analyzed. For example, traffic routing functionality 2012 may receive an
25 incoming connection request that references a previously-opened/established

1 session of a particular type. At block 2404, a local DAM table is searched using
2 the session reference. For example, for a given load balancing unit 106 and traffic
3 routing functionality 2012, the DAM portion 2202 thereof may search its
4 corresponding DAM table 2206 looking for the session reference.

5 At block 2406, it is determined if the session reference matches a key of the
6 local DAM table. For example, DAM portion 2202 may search key fields
7 2304(K) of multiple entries 2304 of DAM table 2206 to determine whether the
8 session reference matches any values of the key fields 2304(K). If so, flow
9 diagram 2400 continues at block 2412.

10 If, on the other hand, the session reference does not match any key, flow
11 diagram 2400 continues at block 2408. At block 2408, a query atom function call
12 is made. For example, DAM portion 2202 may make a query atom 2204(Q)
13 function call that includes the session reference/identifier as the key. The query
14 atom 2204(Q) function call may be sent to at least one other DAM portion 2202.
15 The number, selection, order, etc. of possible destination DAM portions 2202 for
16 query atom 2204(Q) may depend on the options (e.g., replication level, hashing for
17 location selection, local caching upon acquisition, point-to-point versus multicast,
18 etc.) employed by DAM 2202.

19 At block 2410, a returned atom is received. For example, information from
20 a returned atom 2204(R) function call that is issued by another DAM portion 2202
21 may be received. The other DAM portion 2202 successfully located an atomic
22 entry 2304 in its corresponding DAM table 2206, with the located atomic entry
23 2304 having a key that matches the session reference. The information from the
24 returned atom 2204(R) function call includes values from key field 2304(K) and
25 data field 2304(D) for the located atomic entry 2304. These values correspond to

f

1 the session identifier of the session and the network address of the host 108 that is
2 affinitized to the session.

3 At block 2412, an atomic entry is extracted. The atomic entry is extracted
4 from the local DAM table if a match was found locally (at blocks 2404 and 2406)
5 or from the returned atom if a match was found elsewhere (at blocks 2408 and
6 2410). For example, an atomic entry 2304 may be extracted from DAM table
7 2206 of the DAM portion 2202 or from information received by a return atom
8 2204(R) function call. The extracted atomic entry 2304 may be cached at the local
9 DAM table 2206 if received as a result of the return atom 2204(R) function call.

10 At block 2414, the host having session affinity with the referenced session
11 is ascertained from the atomic entry. For example, a value of the data field
12 2304(D) of the extracted atomic entry 2304 may be ascertained to thereby
13 ascertain a network address of the affinitized host 108. At block 2416, the
14 incoming connection request is routed to the ascertained host. For example, traffic
15 routing functionality 2012 and/or forwarding functionality may route the incoming
16 connection request having the session reference to the ascertained and affinitized
17 host 108. Exemplary classifying, request routing, and forwarding functionalities
18 are described in the following section.

19 **Exemplary Classifying, Forwarding, and Request Routing**

20 This section describes how traffic routing may be implemented for network
21 load balancing, including with regard to high availability of such traffic routing
22 functionality. Traffic routing functionality may include classifying and/or
23 requesting routing functionality, especially in conjunction with forwarding
24 functionality. This section primarily references FIGS. 25-31. It illuminates the
25 functionality of a request router 306 (of FIG 3), an interrelationship between

1 C1 is communicating across network 104 with hosts 108(1) and 108(2) having
2 network addresses H1 and H2, respectively. The communications are effectuated
3 via load balancing infrastructure that includes an HTTP/SSL request router
4 306(H/S).

5 HTTP/SSL request router 306(H/S) terminates HTTP and SSL traffic,
6 decrypts SSL traffic, examines each HTTP request from client 102, applies
7 application-specific rules to classify each request and to determine the "best"
8 endpoint for that request while taking into account application endpoint health and
9 load information, and submits the request to the endpoint. The request submission
10 to the endpoint uses a separate TCP connection than that of the one originated by
11 client 102 (the latter connection is terminated at HTTP/SSL request router
12 306(H/S)). These actions may be considered as logically equivalent to the actions
13 performed by a classifier 304, but a difference arises in that these actions in
14 HTTP/SSL request router 306(H/S) are occurring at the logical request level for
15 each request within the TCP connection. HTTP/SSL request router 306(H/S), and
16 request routers 306 generally, can use the same (i) application health and load and
17 (ii) session tracking infrastructure that is used by classifiers 304.

18 HTTP/SSL request router 306(H/S) is acting as an intermediary between
19 client 102 and two hosts 108(1) and 108(2). It is handling two requests from client
20 102 over a single TCP connection. In a described implementation, the resulting
21 request routing involves a number of actions. First, client 102 establishes an http
22 or https connection [1] to HTTP/SSL request router 306(H/S) and sends a request
23 #1 2502(1).

24 Second, HTTP/SSL request router 306(H/S) terminates the SSL session (if
25 the traffic is SSL encrypted), parses request #1 2502(1), and examines the content

of request #1 2502(1). Taking into account application health and load as well as session information, HTTP/SSL request router 306(H/S) determines that host 108(1) is the "best" host for this particular request #1 2502(1) in this example.

Third, HTTP/SSL request router 306(H/S) establishes a secondary TCP connection [2] to host 108(1). It may alternatively use an existing connection [2] to host 108(1). HTTP/SSL request router 306(H/S) then sends an e.g. unencrypted version of request #1 2502(1) to host 108(1). Fourth, host 108(1) replies with a response #1 2504(1). Fifth, HTTP/SSL request router 306(H/S) encrypts this response #1 2504(1) and sends it back to client 102 on TCP connection [1].

Sixth, client 102 sends another request, request #2 2502(2). Request #2 2502(2) is handled similarly to the handling of request #1 2502(1), except that HTTP/SSL request router 306(H/S) selects host 108(2). The different selection may be because host 108(1) is now failing or more-heavily loaded, because request #2 2502(2) is directed to a different URL than request #1 2502(1), and so forth. Regardless, HTTP/SSL request router 306(H/S) establishes another secondary TCP connection, but this secondary TCP connection [3] is to host 108(2). Unencrypted request #2 2502(2) is routed to host 108(2), and a response #2 2504(2) is received therefrom as a result. An encrypted version of response #2 2504(2) is then sent to client 102 from HTTP/SSL request router 306(H/S).

Seventh, client 102 closes TCP connection [1] with HTTP/SSL request router 306(H/S). HTTP/SSL request router 306(H/S) (at some future time) closes connections [2] and [3] that it made to hosts 108(1) and 108(2), respectively, on behalf of client 102. TCP connection [2] may alternatively be closed after HTTP/SSL request router 306(H/S) decides to open/use TCP connection [3] for request #2 2502(2).

1 Because an HTTP/SSL request router 306(H/S) terminates the http/https
2 connection, HTTP/SSL request router 306(H/S) can do more than route requests.
3 For example, HTTP/SSL request router 306(H/S) can potentially maintain its own
4 cache of responses (e.g., with an out-of-band mechanism to invalidate the cache).
5 As noted in the above example, HTTP/SSL request router 306(H/S) can also
6 potentially route different kinds of requests to different sets of hosts 108 based on
7 e.g. the requested URL. Conversely, HTTP/SSL request router 306(H/S) can
8 potentially aggregate requests from many short-lived client connections and send
9 them over a few, long-standing TCP connections to hosts 108. Such connection
10 aggregation can reduce the TCP connection processing overhead in hosts 108.

11 Request routers of other classes may correspond to other exemplary
12 protocols besides HTTP. For example, a request router may be a SOAP request
13 router. SOAP request routers function analogously to an HTTP/SSL request router
14 306(H/S). However, SOAP request routers specialize in routing SOAP traffic.
15 SOAP request routers understand SOAP headers and make routing decisions based
16 on the SOAP headers as well as application health and load.

17 Both packet-level classification and forwarding (or packet-level routing)
18 and request-level routing can provide some form of layer-7 load balancing. Layer-
19 7 load balancing is described further below in the section entitled "Exemplary
20 Connection Migrating with Optional Tunneling and/or Application-Level Load
21 Balancing". Packet-level routing provides read-only access to the initial portion of
22 a client's TCP connection data, and request-level routing provides read and modify
23 access to an entire data stream.

24 Packet-level routing typically has several advantages over request-level
25 routing. These advantages include transparency (client packets are delivered to

1 hosts as-is, preserving source and destination IP addresses and port numbers), low
2 processing overhead (generally, forwarding traffic involves a route lookup), low
3 latency (individual packets are forwarded, and packets are not queued once the
4 TCP connection destination has been determined), and high-availability (generally,
5 a failure in a forwarder does not terminate the TCP connection). Request-level
6 routing, on the other hand, typically has the following advantages over packet-
7 level routing: an ability to examine an entire data stream flowing to and from the
8 client; and an ability to transform a data stream, and even to split the data stream
9 among multiple hosts or aggregate data streams from multiple clients.

10 FIG. 26 is a flow diagram 2600 that illustrates an exemplary method for
11 routing incoming packets with regard to (i) session information and (ii) health and
12 load information. Flow diagram 2600 includes eight blocks 2602-2616. Although
13 the actions of flow diagram 2600 may be performed in other environments and
14 with a variety of software schemes, FIGS. 1-3, 12, 18-20, 22, and 23B are used in
15 particular to illustrate certain aspects and examples of the method.

16 At block 2602, an incoming packet is received. For example, a packet from
17 a client 102 may be received at a forwarder 302 of a load balancing unit 106. At
18 block 2604, it is determined if the received packet is for a preexisting session. For
19 example, forwarder 302 may consult a local DAM table 2206() to determine that
20 the received packet is already part of a TCP/IP session.

21 Additionally, forwarder 302 may consult the local DAM table 2206() and
22 determine that the received packet is not already part of a TCP/IP session. In this
23 case, forwarder 302 provides the received packet to a classifier 304, which checks
24 for a higher level session affinity for the received packet if it has a session
25 reference. Examples for these actions are described above with particular

1 reference to FIG. 24 and further below with particular reference to FIGS. 27 and
2 28.

3 If the received packet is for a preexisting session (as determined at block
4 2604), then flow continues at block 2606. At block 2606, a host that is affinitized
5 to the preexisting session is ascertained. For example, an affinitized host 108 may
6 be ascertained from the local DAM 2206() and/or the overall distributed DAM
7 2206 by forwarder 302 or classifier 304.

8 At block 2608, it is determined if the affinitized host is healthy. For
9 example, classifier 304 may consult a consolidated health and load cache 1208 to
10 determine if the affinitized host 108 is healthy, especially for those received
11 packets that are part of sessions that are of a higher logical level than TCP/IP
12 sessions. The action(s) of this block may be accomplished in conjunction with a
13 health and load handler 314.

14 If the affinitized host is healthy (as determined at block 2608), then flow
15 continues at block 2610. At block 2610, the received packet is routed to the
16 affinitized host. For example, forwarder 302 (for TCP/IP sessions) or classifier 304
17 (for higher-level sessions) may route the packet to the affinitized host 108. In an
18 alternative implementation, classifier 304 may return the received packet to
19 forwarder 302 for routing to the affinitized host 108 even for received packets that
20 are part of higher-level sessions.

21 If, on the other hand, the affinitized host is not healthy (as determined at
22 block 2608), then flow continues at block 2612. Also, if on the other hand, the
23 received packet is not for a preexisting session (as determined at block 2604), then
24 flow continues at block 2612. At block 2612, a host is selected responsive to
25 health and load information. For example, classifier 304 may select a host 108

45

1 from and/or using a health and load-related application allotment (e.g., from a
2 target application endpoint allotment response 1804) that is attained from health
3 and load handler 314. Examples for these action(s) are described above with
4 particular reference to FIGS. 19 and 18 and further below with particular reference
5 to FIG. 30.

6 At block 2614, the received packet is routed to the selected host. For
7 example, classifier 304 may route (optionally via forwarder 302) the packet to the
8 selected host 108. At block 2616, a route for a connection path to the selected host
9 is plumbed. For example, classifier 304 may add a session information entry to
10 DAM table 2206, especially at the DAM table 2206() that is local to the forwarder
11 302 that provided the received packet to the classifier 304. This session
12 information entry may be replicated in accordance with the instituted redundancy
13 policy for a DAM 2202 (e.g., of a session tracker 308).

14 The action(s) of block 2614 and those of block 2616 may be performed in
15 the order specifically illustrated, with those of block 2616 being performed prior
16 to those of block 2614, with the actions partially or fully overlapping in any order,
17 and so forth. It should be noted that the actions performed by classifier 304 as
18 described above may alternatively be performed by a request router 306 (or more
19 generally traffic routing functionality 2012).

20 FIG. 27 illustrates an exemplary traffic routing flow in the absence of
21 failures. As illustrated, one or more load-balancing-aware switches 202(LBA)
22 front the remaining load balancing infrastructure 106 (not separately indicated).
23 Forwarding and classifying functionality are distributed across three devices or
24 nodes. A first device includes forwarder 302(1) and classifier 304(1). A second
25 device includes classifier 304(2). A third device includes forwarder 302(2).

yb

1 With classifier 304(2) executing on the second device and forwarder 302(2)
2 executing on the third device, each device may be specially tuned for its respective
3 functions. For example, the hardware, software, firmware, some combination
4 thereof, etc. of the second device and the third device may be adapted to support
5 the desired functionality without excessive over provisioning. Thus, the third
6 device that includes forwarder 302(2) may be akin to a switch and/or router from a
7 hardware capability perspective, and the second device that includes classifier
8 304(2) may be more akin to a server and/or personal computer from a hardware
9 capability perspective.

10 Although shown as three devices that are providing functionality across
11 four components, alternative logical and/or device-level configurations for
12 forwarding and classifying functionality are applicable to the exemplary traffic
13 routing flow that is described here for FIG. 27. Also, although the routing
14 destinations are shown as hosts 108, the descriptions herein of routing
15 implementations may alternatively be applied more generally to a next node
16 destination for the packet and not necessarily a final node that consumes the
17 packet.

18 A DAM 2202 realization of session tracker 308 is used to implement DAM
19 table 2206. However, session affinity preservers 1904 in general are also
20 applicable to the exemplary traffic routing flow of FIG. 27. Forwarder 302(1)
21 includes DAM table portion 2206(1), and forwarder 302(2) includes DAM table
22 portion 2206(2). Incoming packets are routed to host 108(1) or host 108(2).

23 In a described implementation, DAM 2202 is a distributed, in-memory
24 table of "atoms" 2304 (e.g., keyword-value pairs, with optional metadata) having
25 session information. DAM 2202 and DAM table 2206 is described further above

g/

1 with particular reference to FIGS. 22-24. Any node in the cluster of classifiers 304
2 may add, query, and delete atoms 2304. DAM 2202 maintains a highly available
3 DAM table 2206 that includes active (e.g., TCP/IP level) routes as well as higher-
4 level session information.

5 At (1), load-balancing-aware switches 202(LBA) direct an incoming packet
6 to forwarder 302(1). At (2), forwarder 302(1) consults its internal routing table,
7 DAM table 2206(1). When forwarder 302(1) does not find an atomic entry 2304
8 for this packet, it forwards the packet to its assigned and/or associated classifier,
9 classifier 304(1).

10 At (3), classifier 304(1) recognizes that the packet in this example is a TCP-
11 SYN packet. Classifier 304(1) therefore treats the packet as a start of a new TCP
12 connection from a client 108. Using health and load information from a health
13 and load handler 314 (not explicitly illustrated), classifier 304(1) determines that
14 host 108(1) should receive this TCP connection. Classifier 304(1) updates DAM
15 table 2206(1) that serves as the local routing table for forwarder 302(1), and it also
16 inserts an atomic entry 2304 representing the route into the overall DAM 2206.
17 These may be separate operations, a single operation in which the TCP/IP-level
18 sessions of DAM table 2206 are located at forwarders 302, and so forth. DAM
19 2202 internally replicates this route to one or more other members of the cluster of
20 classifiers 304 in accordance with its stipulated redundancy policy.

21 At (4), forwarder 302(1) directly forwards subsequent packets for this
22 connection to host 108(1) without interacting with classifier 304(1). DAM 2202
23 can be used to mask, at least in part, the failure of a forwarder 302, a classifier
24 304, or a forwarder/classifier pair 302/304. DAM 2202 can also be used, at least
25 in part, to preserve client connectivity if load-balancing-aware switches 202(LBA)

y/k

1 inadvertently start sending packets for an established connection to a different
2 forwarder 302.

3 FIG. 28 illustrates an exemplary traffic routing flow in the presence of
4 failure(s). In contrast to the "failure-free" exemplary traffic routing flow of FIG.
5 27, a failure has occurred in a portion of the network load balancing infrastructure
6 106 (not specifically identified) of FIG. 28. Specifically, the first device, on which
7 forwarder 302(1) and classifier 304(1) are resident and executing, fails after the
8 connection that is illustrated in FIG. 27 is established. This failure is masked, at
9 least in part, by DAM 2202.

10 At (1), load-balancing-aware switches 202(LBA) detect the failure of
11 forwarder 302(1) and start forwarding packets for the connection to some other
12 forwarder 302 in the cluster. In this example, the other forwarder 302 is forwarder
13 302(2). Although FIG. 28 illustrates a failure situation, load-balancing-aware
14 switches 202(LBA) may also send this traffic to forwarder 302(2) even if
15 forwarder 302(1) is still available. This non-failure-induced change of forwarders
16 302 may occur, for example, because load-balancing-aware switches 202(LBA)
17 "forget" the affinity of this traffic to forwarder 302(1). The actions of notations
18 (2)-(5) apply to both the failure and the "forgotten affinity" situations.

19 At (2), forwarder 302(2) consults its routing table, DAM table 2206(2).
20 When it does not find a route for this packet, it forwards the packet to its classifier
21 304(2). At (3), classifier 304(2) recognizes that this packet is a "mid-connection"
22 TCP packet, and classifier 304(2) queries DAM 2202 for the route for this packet.
23 DAM 2202 responds with the route for the connection from an atomic entry 2304
24 that is associated therewith.
25

At (4), classifier 304(2) plumbs the route in forwarder 302(2). An exemplary protocol for plumbing routes is described further below. At (5), subsequent packets for this connection that are directed to forwarder 302(2) are routed directly to the correct host, which is host 108(1) in this example, without consulting classifier 304(2).

Generally, a route plumbing protocol for communications between classifiers 304 and forwarders 302 includes instructions to add and remove routes. More specifically, an add route instruction is sent from a classifier 304 to a forwarder 302 in order to plumb a route from the forwarder 302 to a destination host 108 for a given connection. By way of example, an add route instruction can be provided to forwarder 302(2) from classifier 304(2) as indicated at (4) in FIG. 28. The route (e.g., a key and corresponding value) is added to local DAM table 2206(2) for quick access by forwarder 302(2) in the future. In this example, classifier 304(2) is a separate device from forwarder 302(2), so the route plumbing protocol may be an inter-device protocol. However, the route plumbing protocol may also be utilized for intra-device communications.

In a described implementation, classifier 304(2) includes a connection inventory 2802. With connection inventory 2802, classifier 304(2) keeps track of the connections of any forwarders 302 (such as forwarder 302(2)) for which classifier 304(2) plumbs routes. To enable classifier 304(2) to keep track of the connections, including cessations thereof, forwarder 302(2) forwards final packets for connections (such as a TCP FIN packet) to classifier 304(2). Classifier 304(2) then deletes an entry in connection inventory 2802 that corresponds to the connection and sends a delete route instruction to forwarder 302(2). Upon receiving the delete route instruction, forwarder 302(2) removes the corresponding

X85

1 route in DAM table 2206(2). In this manner, the classifying functionality in
2 conjunction with session tracking functionality can control the route tables, and
3 the routes thereof, that are used by the forwarding functionality. Consequently,
4 forwarding functionality that is separated onto a different device may be
5 effectuated using high-speed, but relatively simple, hardware.

6 FIG. 29 illustrates additional exemplary failover procedures for high
7 availability of network load balancing infrastructure 106. Failover procedures for
8 two different failures, failure 2902 and failure 2906, are described. As illustrated,
9 network load balancing infrastructure 106 (not separately indicated) includes five
10 components: forwarder 302(1), forwarder 302(2), forwarder 302(3), classifier
11 304(1), and classifier 304(2).

12 In a described implementation, each of these five components 302(1),
13 302(2), 302(3), 304(1), and 304(2) corresponds to an individual device. However,
14 similar failover procedures apply to environments in which different load
15 balancing components share devices.

16 Initially at [1], router/switch(es) 202 direct an incoming packet that
17 happens to be for a new connection to forwarder 302(1). Because forwarder
18 302(1) does not have a route for this connection in its local routing table, it sends
19 the packet to classifier 304(1) as indicated by the dashed double arrow at (1).
20 Classifier 304(1) first checks session information with reference to session
21 tracking 308 for a possible higher-level session affinity. In this example, the
22 packet is not affinized to an existing session, so classifier 304(1) selects a host 108
23 with reference to health and load information with reference to health and load
24 handling 314.
25

Specifically, classifier 304(1) selects host 108(1) in this example. Assuming the packet is for a TCP/IP connection, this TCP/IP session as linked to host 108(1) is added to DAM 2202 using an add atom 2204(A) function call by classifier 304(1). The initial packet is forwarded to host 108(1) by classifier 304(1) or forwarder 302(1). Classifier 304(1) also plumbs a route in the local routing table of forwarder 302(1). Subsequent packets are forwarded to host 108(1) by forwarder 302(1) without further interaction with classifier 304(1).

At some time during connection [1], there is a failure 2902 at forwarder 302(1). With load-balancing-aware router/switch(es) 202(LBA), this failure 2902 is detected. As a result, at point 2904, router/switch(es) 202 direct later packets that would have been sent to forwarder 302(1) along connection [1] to another forwarder 302, which is forwarder 302(2) in this example.

Forwarder 302(2) thus receives future packets along a connection [2]. Because forwarder 302(2) does not have an entry in its local routing table for the packets that were formerly directed to forwarder 302(1), forwarder 302(2) sends the first received packet of connection [2] to the classifier to which it is assigned/associated. In this example, forwarder 302(2) is assigned to classifier 304(2) as indicated by the dashed double arrow at (2).

Classifier 304(2) uses a query atom 2204(Q) function call to attain the atomic entry 2304 (not explicitly shown) from DAM 2202 that is associated with the existing TCP/IP connection. This atomic entry 2304 is provided through DAM 2202 of session tracking 308 via a return atom 2204(R) function call. Classifier 304(2) extracts the host 108(1) that is affinitized with this TCP/IP connection from the returned atomic entry 2304. Classifier 304(2) forwards the first received packet for connection [2] to host 108(1) and also plumbs a route in the local

10/2

1 routing table of forwarder 302(2). Subsequent packets are forwarded to host
2 108(1) by forwarder 302(2) without further interaction with classifier 304(2).

3 The above descriptions focus predominantly on failures of individual
4 forwarder 302 components. However, classifier 304 components can also fail.
5 For example, at some point, there is a failure 2906 at classifier 304(2). Forwarder
6 302(2) detects failure 2906 when it attempts to consume classification services or
7 through noticing a lack of some aliveness indication such as a heartbeat-type
8 indicator. To handle failure 2906, forwarder 302(2) is reassigned or re-associated
9 with a different classifier 304, which is classifier 304(1) in this example. Future
10 classification functionality is provided to forwarder 302(2) by classifier 304(1) as
11 indicated by the dashed double arrow at (3).

12 FIG. 30 illustrates an exemplary operational implementation of traffic
13 routing interaction with health and load information. Forwarder 302 and classifier
14 304 interact with health and load handler 314 in order to route packets to hosts
15 108(1), 108(2) ... 108(n). Although a forwarder 302 and a classifier 304 are
16 illustrated, the exemplary operational implementation is also applicable to a
17 request router 306 (or traffic routing functionality 2012 in general).

18 As illustrated, host 108(1) includes application endpoints IP1, IP3, and IP4
19 for application #1, application #1, and application #2, respectively. Host 108(2)
20 includes application endpoints IP2 and IP6 for application #1 and application #2,
21 respectively. Host 108(n) includes application endpoint IP5 for application #2.
22 These hosts 108(1), 108(2) ... 108(n) and application endpoints IP1, IP2, IP3, IP4,
23 IP5, and IP6 are monitored by health and load handler 314 (e.g., using health and
24 load infrastructure 1202, consolidated health and load cache 1208, etc.).
25

1 In a described implementation, at (1) classifier 304 requests one or more
2 application endpoint allotments (e.g., via at least one target application endpoint
3 allotment request 1802) in an environment using a token allotment scheme 1806.
4 Health and load handler 314, in this example, responds by providing token
5 allotments 3002 (e.g., via at least one target application endpoint allotment
6 response 1804).

7 Specifically, a token allotment for application #1 3002(1) and a token
8 allotment for application #2 3002(2) are available to classifier 304. Token
9 allotment for application #1 3002(1) initially provides 40 tokens for IP1, 35 tokens
10 for IP2, and 25 tokens for IP3. Token allotment for application #2 3002(2)
11 provides 10 tokens for IP4, 72 tokens for IP5, and 18 tokens for IP6. For each
12 new connection that is allocated a routing to an application endpoint by classifier
13 304, a token is consumed by classifier 304.

14 At (2), forwarder 302 receives an initial incoming packet for a new
15 connection. Because no routing for this new connection is present in local DAM
16 table portion 2206 of forwarder 302, forwarder 302 forwards the initial packet to
17 classifier 304 at (3).

18 At (4), classifier 304 (e.g., after determining that the initial packet does not
19 include a session reference for a higher-level session) selects an application
20 endpoint (and thus a host 108) responsive to health and load information.
21 Specifically, for a new connection that is to be served by application #1, classifier
22 304 can select any of IP1, IP2, and IP3 if a token for the respective endpoint still
23 exists.

24 Classifier 304 can consume tokens in any of many possible manners. For
25 example, classifier 304 may use a round-robin approach regardless of the number

1 of tokens per endpoint. Alternatively, classifier 304 may simply start from IP1 and
2 progress through IP3 while consuming all tokens for each endpoint before moving
3 to the next endpoint in a linear approach. Also, classifier 304 may consume a
4 token from the endpoint-defined-set of tokens that currently has the greatest
5 number of tokens at any one moment. Using the latter approach, classifier 304
6 selects IP1. Other approaches may also be employed.

7 As illustrated, classifier 304 consumes a token for application endpoint IP2.
8 Consequently, the token set for IP2 is reduced from 35 tokens to 34 tokens as a
9 token is consumed. Also, the initial packet for the new connection is to be routed
10 to application endpoint IP2.

11 At (5A), the initial packet is forwarded from classifier 304 to application
12 endpoint IP2 of host 108(2). Before, during, or after this forwarding, classifier
13 304 at (5B) plumbs a route for this connection in local DAM table portion 2206.
14 Classifier 304 may also add an atomic entry 304 for this session into DAM table
15 2206 for distribution and replication purposes. At (6), future packets for this
16 connection/session are forwarded from forwarder 302 to application endpoint IP2
17 of host 108(2) using the local routing table of forwarder 302 as realized by local
18 DAM table portion 2206 in FIG. 30.

19 FIG. 31 illustrates exemplary high availability mechanisms for network
20 load balancing infrastructure 106. Specifically, exemplary failure detection 3104,
21 exemplary failure handling 3106, and exemplary failure recovery 3108 are shown.
22 These exemplary high availability mechanisms are described with regard to
23 different network load balancing infrastructure 106 components. The network
24 load balancing infrastructure 106 components include a forwarder 302, a classifier
25 304, a request router 306, a session tracker 308, and a health and load handler 314.

1 At 3102(A), forwarder 302 undergoes a local failure. At 3104(A), at least
2 one load-balancing-aware switch detects the failure. To handle local failure
3 3102(A), packets are redirected to other forwarder(s) at 3106(A) by the load-
4 balancing-aware switch. To recover from the failure of forwarder 302, routes that
5 were stored locally at forwarder 302 are rebuilt at 3108(A) at the forwarder(s) to
6 which packets are redirected using a distributed session tracking manager and a
7 table thereof such as a DAM and a DAM table thereof. The distributed session
8 tracking manager may therefore include data redundancies of one or more levels.

9 At 3102(B), classifier 304 undergoes a local failure. At 3104(B), at least
10 one forwarder detects the failure. To handle local failure 3102(B), packets are
11 redirected to other classifier(s) at 3106(B) by the forwarder detecting the failure.
12 To recover from the failure of classifier 304, session information that was stored
13 locally at classifier 304 are rebuilt at 3108(B) at the classifier(s) to which packets
14 are redirected using DAM. This session information may be, for example, session
15 information of a higher level than baseline TCP/IP connections. Also, such
16 session information may be considered as part of session tracking infrastructure
17 that is resident on the same device as classifier 304.

18 At 3102(C), request router 306 undergoes a local failure. At 3104(C), at
19 least one forwarder and/or load-balancing-aware switch detect the failure. To
20 handle local failure 3102(C), packets are redirected to other request router(s) at
21 3106(C) by the forwarder and/or load-balancing-aware switch. Individual current
22 logical requests on which request router 306 is working upon the occurrence of
23 local failure 3102(C) may be lost unless each such individual logical request is
24 replicated while the request is being serviced. To recover from the failure of
25 request router 306, session information and/or routes that were stored locally at

✓

1 request router 306 are rebuilt at 3108(C) at the request router(s) to which packets
2 (and thus new logical requests) are redirected. The session information rebuilding
3 may be effectuated using DAM. Again, such session information may be
4 considered as part of session tracking infrastructure that is resident on the same
5 device as request router 306.

6 At 3102(D), session tracker 308 undergoes a local failure. At 3104(D), at
7 least one forwarder and/or classifier detect the failure. For example, if session
8 tracker 308 is resident on a same device as a classifier, then a forwarder or another
9 classifier may detect the failure. If session tracker 308 is resident on a separate
10 device, then a classifier may detect the failure. To handle local failure 3102(D),
11 data redundancy of one or more levels and distribution across multiple devices are
12 instituted at 3106(D) for the tracked session information. It should be noted that
13 the redundancy and distribution are instituted prior to failure 3102(D). To recover
14 from the failure of session tracker 308, session information from the tables of the
15 DAM may be redistributed and re-replicated at 3108(D) across at least two devices
16 (if not already so distributed and sufficiently replicated) in order to handle a
17 second level of failure.

18 At 3102(E), health and load handler 314 undergoes a local failure. At
19 3104(E), at least one classifier and/or request router detect the failure. For
20 example, a component that is receiving health and load information from health
21 and load handler 314 may detect a failure if health and load handler 314 becomes
22 non-responsive, especially if health and load handler 314 is resident on a different
23 device from that of the inquiring component. To handle local failure 3102(E),
24 cached health and load data redundancy and intrinsic failure handling are
25 employed at 3106(E) for the health and load information.

1 For example, each health and load handler 314 can include a consolidated
2 health and load information cache 1208 that duplicates information in health and
3 load tables 1204 on multiple hosts 108. Also, consumers of the health and load
4 information 1206 of a given health and load handler 314 may be located on a same
5 device as health and load handler 314 so that failure of health and load handler
6 314 is intrinsically acceptable. Similarly, the authoritative version of a respective
7 portion of health and load information 1206 is located on a respective host 108 so
8 that failure of the host 108 renders the loss of the respective portion of the health
9 and load information acceptable.

10 To recover from the failure of health and load handler 314, a given network
11 load balancing component that consumes health and load information may query a
12 different health and load handler because each such health and load handler
13 includes a consolidated cache of health and load handler information. Also, when
14 health and load handler 314 is again accessible, message protocol 1500 may be
15 used at 3108(E) to rebuild its consolidated cache of health and load information.
16 Using these exemplary high availability mechanisms, failures of network load
17 balancing infrastructure 106 components can be detected, handled, and recovered
18 from in order to mask such failures from clients 102.

19 **Exemplary Connection Migrating**

20 **with Optional Tunneling and/or Application-Level Load Balancing**

21 This section describes how connection manipulation, such as connection
22 migration, may be utilized in network load balancing. This section primarily
23 references FIGS. 32-39 and illuminates connection migrating functionality such as
24 that provided by connection migrator 310 (of FIG. 3). As described above with
25 reference to FIGS. 3 and 4, each incoming connection at load balancing

1 infrastructure 106 may be terminated thereat. Afterwards, the connection may be
2 migrated to a host 108 such that the connection is then terminated at the host 108.
3 Connection migrator 310 is capable of performing this connection migration and
4 may be located partially at hosts 108 to effectuate the migration. Such connection
5 migration may be performed in conjunction with application-level load balancing
6 by a classifier 304 and/or using tunneling via tunneler 312.

7 FIG. 32 illustrates an exemplary approach to application-level network load
8 balancing with connection migration. Application-level, or layer-7, load balancing
9 pertains to making load balancing decisions with regard to an application that is to
10 handle a connection. To perform application-level load balancing, load balancing
11 infrastructure 106 usually takes into consideration a data portion of a connection.
12 Unless request routing is employed, a classifier 304 typically takes a peek at the
13 initial portion of a connection and then migrates the connection, in conjunction
14 with connection migrator 310, to a selected host 108.

15 For application-level load balancing in a TCP-based environment generally,
16 classifiers 304 peek at the initial portion of a client's TCP data when deciding
17 where to forward the client's TCP connection. Thus, application-level logic
18 examines the client's data and makes load balancing decisions based on that data.
19 For example, if a connection is an (unencrypted) HTTP connection, a classifier
20 304 can take a peek at the HTTP header of the first HTTP request in the
21 connection, and it can make routing decisions based on some portion of the
22 content of the header (e.g., the URL, a cookie, etc.). Although application-level
23 load balancing, connection migration, and tunneling are applicable to other
24 protocols, TCP/IP is used predominantly in the examples herein.

25

1 As illustrated, load balancing infrastructure 106 (not specifically indicated)
 2 includes a forwarder 302, a classifier 304, a tunneler 312, and a connection
 3 migrator 310 (and possibly e.g. load-balancing-aware router/switches 202(LBA)).
 4 Forwarder 302 corresponds to the virtual IP address and forwards packets to hosts
 5 108 in accordance with host selections by classifier 304. Although not specifically
 6 shown in FIG. 32 for clarity, hosts 108 also include connection migrator 310
 7 functionality and tunneler 312 functionality.

8 In a described implementation, forwarder 302, classifier 304, and
 9 connection migrator 310 (at classifier 304 and on hosts 108), along with TCP
 10 protocol software on classifier 304 and hosts 108, cooperate to provide connection
 11 migration. The connection migration illustrated in FIG. 32 is for a connection
 12 from client 102(1) that is initially terminated at classifier 304. After connection
 13 migration, the connection from client 102(1) is terminated at host 108(1). Once
 14 the connection is terminated at host 108(1), packets for the connection may be
 15 tunneled using tunneler 312 (at forwarder 302 and host 108(1)).

16 At (1), client 102(1) sends a SYN packet to forwarder 302 to signal the start
 17 of a new TCP connection. At (2), forwarder 302 forwards this packet to classifier
 18 304. At (3), classifier 304 accepts the TCP connection on behalf of a host 108
 19 (whose identity is not yet known because the actual target host 108() has yet to be
 20 selected). In TCP protocol terms, classifier 304 sends a SYN-ACK packet to
 21 client 102(1).

22 At (4), client 102(1) begins sending data. (The initial SYN packet may also
 23 contain data.) The data is processed by classifier 304, which can consult
 24 application-specific logic. The application-specific logic can relate to which host
 25 108 is capable of handling or best handling which types of requests or

26

102

connections. Hence, classifier 304 uses the data, as well as application health and load information from health and load handler 314 and optionally application session information from session tracker 308, to determine a host 108 that is better or best suited to handle this connection from client 102(1). In this example, host 108(1) is selected.

At (5), classifier 304 sends a "binary blob" that represents the state of the TCP connection to host 108(1). This connection state is aggregated with cooperation from a TCP stack on classifier 304 by connection migrator 310. The binary blob contains data from client 102(1) that has been acknowledged by classifier 304 and TCP parameters such as the TCP/IP 4-tuple, initial sequence numbers, and so forth.

At (6), a connection migrator 310 component on host 108(1) (not explicitly shown in FIG. 32) "injects" this connection into a TCP stack on host 108(1). This connection state injection is performed in cooperation with the TCP stack on host 108(1), making it appear to applications 316 on host 108(1) that this connection was originally accepted by host 108(1) itself. Client 102(1) is unaware of the connection migration.

At (7), classifier 304, in cooperation with the TCP stack on classifier 304, silently cleans up the internal state maintained for this connection. Classifier 304 also adds a route in a local routing table of forwarder 302 that indicates host 108(1) as the destination for packets of this connection.

At (8), subsequent packets for the connection are routed by forwarder 302 to host 108(1). These packets may be treated the same by forwarder 302 as those packets for connections that are classified and routed without using connection migration. These subsequent packets may optionally be tunneled from forwarder

4/5

1 302 to host 108(1) using tunneler 312. Tunneler 312 is also illustrated (using
2 dashed lines) at connection migrator 310 at classifier 304 because certain
3 parameter(s) used by tunneler 312 may be determined during a connection
4 migration and/or associated with a connection being migrated. Exemplary
5 implementations for tunneler 312 are described further below with particular
6 reference to FIGS. 38 and 39.

7 FIG. 33 is a flow diagram 3300 that illustrates an exemplary method for
8 migrating a connection from a first device to a second device. Flow diagram 3300
9 includes seven blocks 3302-3314. Although FIGS. 32 and 34-37 focus primarily
10 on connection migration in a network load balancing environment, connection
11 migration as described herein may be effectuated between two devices in general
12 that each include connection migration functionality, such as that of connection
13 migrator 310.

14 At block 3302, a connection is accepted at a first device. For example, a
15 first device may terminate an incoming connection in accordance with one or more
16 protocols of a protocol stack portion of a network stack. At block 3304, data is
17 received for the connection at the first device. For example, this data may be
18 received in an initial packet that requests the connection or in one or more packets
19 that are received subsequent to an acceptance of the connection.

20 At block 3306, a connection state for the accepted connection is aggregated
21 from a protocol stack (or more generally from a network stack) at the first device.
22 For example, a protocol state of the one or more protocols of the protocol stack
23 may be compiled and aggregated with any received data that has been
24 acknowledged. At block 3308, the connection state is sent from the first device to
25

1 a second device. For example, the aggregated information of the connection state
2 may be sent using a reliable protocol to a second device.

3 At block 3310, the connection state for the connection being migrated is
4 received from the first device at the second device. At block 3312, the connection
5 state is injected into a protocol stack (or more generally into the network stack) of
6 the second device. For example, the connection may be rehydrated using the
7 protocols of the protocol stack of the second device such that programs above the
8 protocol stack level are unaware that the connection is a migrated connection.
9 More specifically, the protocol state may be infused into the protocol stack. The
10 aggregated data of the connection state is also incorporated at the second device.
11 At block 3314, the connection is continued at the second device. For example, the
12 connection may be continued at the second device as if the connection was not
13 previously terminated elsewhere.

14 FIG. 34 illustrates an exemplary approach to connection migration from the
15 perspective of an originating device 3400. Connection migration in originating
16 device 3400 is effectuated, at least partly, by connection migrator 310. In a
17 described implementation, originating device 3400 is a device that is part of
18 network load balancing infrastructure 106. For example, originating device 3400
19 may comprise a classifier 304, possibly along with a forwarder 302, a request
20 router 306, and so forth.

21 As illustrated, originating device 3400 includes as parts of its network stack
22 a physical network interface (PNI) 3410, a PNI miniport 3408, a protocol-
23 hardware interface 3406, a protocol stack 3404, and a socket layer 3402.
24 Originating device 3400 also includes load balancing functionality 106, such as a
25 classifier 304 at an application level and connection migrator 310. Specifically,

1 connection migrator 310 includes a migrator intermediate driver 3414 and a
2 migrator shim 3412. Connection migrator 310 is capable of offloading a
3 connection from originating device 3400.

4 In a described implementation, physical network interface 3410 may be a
5 network interface card (NIC) (e.g., an Ethernet NIC), a wireless interface, and so
6 forth. Although only one physical network interface 3410 is shown, a given
7 device may actually have multiple such physical network interfaces 3410 (i.e.,
8 originating device 3400 may be multi-homed). Each physical network interface
9 3410 typically corresponds to one or more physical network addresses.

10 PNI miniport 3408 is a software module that understands and interfaces
11 with the specific hardware realization of physical network interface 3410.
12 Protocol-hardware interface 3406 is a layer that includes one or more respective
13 interfaces between one or more respective protocols and PNI miniport 3408.

14 Protocol stack 3404 includes one or more respective modules that are each
15 directed to one or more respective protocols. Examples of such protocols are
16 described further below with reference to FIGS. 36 and 37. In a transient context,
17 protocol stack 3404 includes a protocol state 3420 for each connection existing at
18 originating device 3400. A socket layer 3402 lies between a program such as load
19 balancing functionality 106 and protocol stack 3404. Socket layer 3402 provides
20 APIs between load balancing functionality 106 and protocol stack 3404, and it
21 enables programs to register for connections, among other things.

22 Migrator intermediate driver 3414, or more generally migrator driver 3414,
23 is located at protocol-hardware interface layer 3406. Migrator shim 3412 is
24 located transparently between protocol stack 3404 and socket layer 3402.
25

1 When an initial packet (not shown) requesting a new connection is
2 presented to originating device 3400, the packet is directed upward from physical
3 network interface 3410, to PNI miniport 3408, through protocol-hardware
4 interface layer 3406, and to protocol stack 3404. As the packet traverses the one
5 or more protocols of protocol stack 3404, protocol state 3420 is created thereat.
6 Also, as a result of this initial packet or as a consequence of load balancing
7 functionality 106 accepting the connection to take a peek at the request, data 3416
8 arrives at originating device 3400.

9 In operation, migrator intermediate driver 3414 diverts a copy of data 3416
10 to the logic of connection migrator 310. When load balancing functionality 106
11 issues a migrate connection function call, the migrate function call is passed to a
12 topmost layer of protocol stack 3404 so that connection state aggregation 3418
13 may commence. Protocol state 3420 is compiled from the one or more protocols
14 of protocol stack 3404. In a TCP/IP implementation, protocol state 3420 may
15 include (i) destination and source TCP ports and IP addresses (e.g., a TCP/IP 4-
16 tuple), (ii) TCP window state, (iii) initial sequence numbers, (iv) timeout
17 information, (v) IP fragment ID, (vi) routing information, and (vii) so forth.

18 Connection state aggregation 3418 also aggregates data 3416 that has been
19 diverted to connection migrator 310 and that has already been acknowledged from
20 originating device 3400 (e.g., by load balancing functionality 106). This
21 aggregated connection state 3418 includes protocol state 3420 and data 3416 (and
22 optionally other connection-related information). Aggregated connection state
23 3418 is then sent as a binary blob 3422 away from originating device 3400 toward
24 a targeted device using a reliable protocol. This binary blob 3422 may also be
25 bundled with a flow identifier if the connection is to be tunneled subsequently

1 with tunneler 312. Flow identifiers with tunneling are described further below
2 with particular reference to FIGS. 38 and 39.

3 FIG. 35 illustrates an exemplary approach to connection migration from the
4 perspective of a target device 3500. Target device 3500 is similar to originating
5 device 3400 with respect to the various illustrated layers/modules, including
6 connection migrator 310. As illustrated however, at least one application 316 at an
7 application level is interfacing with socket layer 3402. Target device 3500 may
8 therefore comprise a host 108. Also, connection migrator 310 is capable of
9 uploading a connection from originating device 3400.

10 In a described implementation, application 316 is the destination of the
11 connection-initiating packet received at originating device 3400. From originating
12 device 3400, target device 3500 receives binary blob 3422. Binary blob 3422
13 includes the connection state associated with the connection being migrated to
14 target device 3500 and optionally a flow identifier. This connection state includes
15 protocol state 3420 and acknowledged data 3416 (and possibly other connection-
16 related information).

17 In operation, when binary blob 3422 reaches protocol-hardware interface
18 layer 3406, migrator intermediate driver 3414 recognizes it as a blob for
19 connection migration and diverts it. The connection state is injected at 3502 to
20 create the appearance to application 316 that the connection was originally
21 terminated at target device 3500.

22 Specifically, protocol state 3420 of injected connection state 3502 is infused
23 into protocol stack 3404. In a described implementation, protocol state 3420 is
24 infused first at higher-level protocols and then at lower-level protocols of protocol
25 stack 3404. After protocol state 3420 is infused into protocol stack 3404, data

1 3416 can be indicated up to application 316. This data 3416 can be provided to
2 application 316 as if it were part of a newly and locally terminated connection.

3 After connection state injection 3502 is completed, the connection initiated
4 by the packet received at originating device 3400 is successfully migrated
5 therefrom to target device 3500. Subsequent packets for the connection may be
6 forwarded directly to target device 3500 without passing through originating
7 device 3400, or at least with only simple routing and no application-level analysis
8 being applied thereto. Optionally, these packets may be tunneled such that
9 migrator intermediate driver 3414 effectively operates as a software-based virtual
10 NIC that is bound to the virtual IP address.

11 FIG. 36 illustrates an exemplary approach to an offloading procedure 3600
12 for a connection migration. Migration offloading procedure 3600 illustrates
13 additional exemplary details for a connection migration by an originating device
14 3400. As illustrated, general protocol stack 3404 includes a TCP stack 3404(T), an
15 IP stack 3404(I), and an address resolution protocol (ARP) stack 3404(A).
16 However, other specific protocol stacks 3404() may alternatively be employed.

17 By way of example, protocol-hardware interface layer 3406 may be
18 realized as a network driver interface specification (NDIS)-based layer in a
19 Microsoft® Windows® operating system (OS) environment. Also, socket layer
20 3402 may be realized as a Winsock™ layer in a Microsoft® Windows® OS
21 environment.

22 In a described implementation, migrator intermediate driver 3414 includes
23 protocol-hardware interfaces 3406 at the junctions to ARP stack 3404(A) and to
24 PNI miniport 3408. Migrator intermediate driver 3414 serves as an offload target
25 in migration offloading procedure 3600. The offload target is a protocol-hardware

5/12

1 interface 3406 miniport as illustrated in this example. In a migration uploading
2 procedure 3700 (as in FIG. 37), migrator intermediate driver 3414 serves as an
3 upload diverter.

4 More specifically, migrator intermediate driver 3414 is bound to each
5 physical network interface 3410 through which a TCP connection may be
6 migrated. Migrator intermediate driver 3414 usually operates as a pass-through
7 driver by passing packets upwards or downwards in the network stack without
8 otherwise interacting with the packets. However, migrator intermediate driver
9 3414 does interact with packets related to connection migration (optionally
10 including subsequently tunneled packets).

11 Responsibilities of migrator intermediate driver 3414 include: (i) the
12 acceptance of migrate offload requests; (ii) the aggregation of the protocol state
13 information that is related to the TCP connection being migrated as compiled from
14 the specific protocol stacks 3404(), along with acknowledged data to produce the
15 connection state information; and (iii) the transmission of the aggregated
16 connection state to a targeted device 3500 for a migration uploading procedure
17 3700. A reliable wire protocol for such transmission may be shared with that used
18 by the session tracking components 2002 and 2010 to send and receive session
19 information messages 2008 (e.g., as described above with reference to FIG. 20).

20 Another responsibility of migrator intermediate driver 3414 (e.g., in a
21 migration uploading procedure 3700) is to initiate the uploading of migrated
22 connections that it receives from other devices and to buffer any incoming packets
23 related to the migrating connection while it is in the process of being uploaded. To
24 upload the connection, migrator intermediate driver 3414 sends an upload request
25 to migrator shim 3412. Migrator shim 3412 issues an inject call down into

1 protocol stack 3404 at TCP stack 3404(A) to instantiate the connection in the
2 protocol stack 3404 portion of the network stack.

3 Migrator shim 3412 exposes a client interface to TCP stack 3404(T) and
4 exposes a provider interface to socket layer 3402. Migrator shim 3412 has two
5 roles: (i) to initiate connection migration offload procedure 3600 on an originating
6 device 3400 and subsequently migration upload procedure 3700 on a targeted
7 device 3500 and (ii) to mediate the classification process between a host
8 application 316 program, a load-balancing classifier 304 program, and socket layer
9 3402. Migrator shim 3412 and migrator intermediate driver 3414 are both further
10 described below with reference to FIGS. 36 and 37.

11 For an exemplary migration offloading procedure 3600, the migration of a
12 TCP connection is performed after classifier 304 classifies the incoming TCP
13 connection using one, two, or more packets thereof. Migration offloading
14 procedure 3600 is described at points <1> through <7>.

15 At <1>, an initialization is performed prior to classification operations.
16 Protocol stack 3404 makes queries at protocol-hardware interface layer 3406 to
17 determine what offloading capabilities, if any, are available. Migrator
18 intermediate driver 3414 indicates that connection migration offloading is
19 available and propagates the query down to PNI miniport 3408. If a TCP chimney
20 offload ability is provided by a physical network interface 3410, PNI miniport
21 3408 also so indicates. TCP chimney offload enables some TCP/IP processing to
22 be offloaded to the hardware of physical network interface 3410 and involves
23 some compiling of protocol state 3420. Consequently, some compiling and
24 aggregation logic may be shared between the two offloading mechanisms.

25

111

1 At <2>, once a TCP connection has been classified, classifier 304 initiates a
2 TCP connection migration to a selected host 108. Specifically, a migration
3 command indicating a targeted device 3500 is issued via socket layer 3402 to
4 migrator shim 3412.

5 At <3>, migrator shim 3412 initiates TCP connection migration to compile
6 the TCP protocol state. Specifically, migrator shim 3412 invokes a TCP initiate
7 migrate offload API (or more generally a migrate connection function call or
8 migrate connection command). This routine compiles the relevant state for the
9 specified TCP connection that is used to reinstate the connection on the targeted
10 device 3500. The compiled protocol state 3420 includes state from the
11 intermediate stack layers, including TCP stack 3404(T), IP stack 3404(I), and ARP
12 stack 3404(A).

13 At <4>, once protocol stack 3404 has compiled protocol state 3420 for the
14 TCP connection being migrated, it invokes an initiate migrate offload API on the
15 miniport to which it is bound; in this example, that miniport is migrator
16 intermediate driver 3414. However, in practice, there may be other intermediate
17 drivers inserted between protocol stack 3404 and migrator intermediate driver
18 3414, such as IP QoS. If so, those IM drivers may participate in the migration, if
19 relevant, by compiling/aggregating their state to the connection state information
20 for the connection being migrated. Intermediate drivers continue to propagate the
21 initiate migrate offload call down the network stack, which eventually results in
22 execution of a migrate offload handler at migrator intermediate driver 3414. At
23 this point, migrator intermediate driver 3414 also aggregates any acknowledged
24 data with the remaining connection state for transfer of the TCP connection to
25 targeted device 3500.

1 At <5>, after storing/copying connection state information for the TCP
 2 connection being migrated, migrator intermediate driver 3414 notifies the network
 3 stack that the migration is in its final stages by invoking an initiate migrate offload
 4 complete API. This initiate migrate offload complete API follows the reverse path
 5 up the network stack, through the same intermediate drivers (if any), and
 6 eventually to protocol stack 3404. As each layer processes this call, state
 7 information that is associated with the migrated connection may be released. Until
 8 the processing of this call is complete, each layer may send updating notifications
 9 down the network stack to update any part of the connection state that has changed
 10 since the migration was initiated.

11 At <6>, when the initiate migrate offload complete routine reaches TCP
 12 stack 3404(T), TCP silently (i.e., no reset is sent to client 108) closes the
 13 connection, flushing all state associated with the migrated connection, and
 14 propagates the initiate migrate offload complete call to migrator shim 3412. At
 15 this point, the network stack is free of any residual knowledge of the migrated
 16 TCP connection.

17 At <7>, when the initiate migrate offload complete call returns to migrator
 18 intermediate driver 3414 (via the migrator shim 3412 portion of connection
 19 migrator 310), the migration of the TCP connection from originating device 3400
 20 to targeted device 3500 may commence with the transfer of the connection state
 21 thereto. The connection state may be transferred asynchronously and reliably.

22 Once migration is initiated, originating device 3400 is also responsible for
 23 ensuring that subsequent data from client 108 is forwarded to target device 3500.
 24 Consequently, even after the connection is successfully migrated to the target, the
 25 originator retains some amount of state for the connection (e.g., a routing table

entry) in order to properly route subsequent packets to the target. When the connection is terminated, the target notifies the originator to enable it to purge whatever residual state remains for the migrated connection.

Furthermore, as a consequence of the asynchronous nature of the connection migration, data packets for the migrating connection that are forwarded by originating device 3400 (or a forwarder designated thereby if a separate device) may start arriving at targeted device 3500 before targeted device 3500 receives the migrated connection state. Migrator intermediate driver 3414 at targeted device 3500 is responsible for buffering those packets until the associated migrated connection is established on targeted device 3500.

FIG. 37 illustrates an exemplary approach to an uploading procedure 3700 for a connection migration. Migration uploading procedure 3700 illustrates additional exemplary details for a connection migration by targeted device 3500.

When a migrated connection arrives at targeted device 3500, it is relayed to migrator intermediate driver 3414 for processing. After amalgamating and assimilating the migrated connection state, migrator intermediate driver 3414, in conjunction with migrator shim 3412, injects the migrated connection into the local network stack in a manner transparent to application 316. For an exemplary migration uploading procedure 3700, the migration of a TCP connection at points <1> through <8> is described.

At <1>, as described above with reference to migration offloading procedure 3600, an initialization is performed prior to application hosting operations. Specifically, protocol stack 3404 makes queries regarding what offloading capabilities, if any, are available. Migrator intermediate driver 3414 fills in the TCP connection migration support query to indicate that connection

XP2

1 migration uploading is available and also propagates the query down to PNI
2 miniport 3408 for possible TCP chimney offload capabilities.

3 At <2>, when connection migration data arrives at target device 3500, the
4 connection migration information (e.g., a bundled binary blob 3422) is delivered
5 to migrator intermediate driver 3414. Migrator intermediate driver 3414 re-
6 assembles the connection state, matches it up with any associated data that has
7 arrived during the migration, and prepares for the upload onto the network stack.
8 Any data from client 102 that arrives during the process of uploading the migrated
9 connection is buffered by migrator intermediate driver 3414. Upon successful
10 completion of the migration, the data will be delivered to application 316.

11 At <3>, to initiate the upload of the migrated connection into the local
12 network stack, migrator intermediate driver 3414 notifies migrator shim 3412 that
13 a migrated connection request has arrived. Migrator intermediate driver 3414 also
14 delivers the connection state (or at least protocol state 3420) to migrator shim
15 3412.

16 At <4>, migrator shim 3412 initiates the upload of the migrated connection
17 by invoking a TCP initiate inject routine (or more generally an infuse protocol
18 state routine) and by providing the migrated protocol state 3420 to TCP stack
19 3404(T). At <5>, TCP/IP recreates the migrated connection throughout protocol
20 stack 3404 using the provided protocol state 3420. This protocol state 3420 may
21 include one or more of transport state (TCP), path state (IP), neighbor and next-
22 hop state (ARP), and so forth.

23 At <6>, if the migrated connection is successfully reestablished on target
24 device 3500, TCP initiates a connect event to a client portion of migrator shim
25 3412 to indicate that a new connection has been established. There are a multitude

of possible reasons for failure, but common reasons may include the lack of a corresponding listener, routing failure, etc. In these cases where the network stack is unable to reestablish the migrated connection, no connect event is indicated and a failure status is specified in the initiate inject complete call. Connection migrator 310 is responsible for cleaning up the migration and for sending a reset notification back to client 102 to abandon the connection.

At <7>, migrator shim 3412 acts as a provider to propagate the connect event to socket layer 3402 so as to indicate to the listening application 316 that a new connection has been established. If the application 316 accepts the connection, it processes the requests and responds through normal read and write socket operations; application 316 can be unaware that the connection was migrated. If the connection is not accepted by the application 316, TCP terminates the connection but does not send a reset notification back to client 102. Again, a failure status is specified in the initiate inject complete call, and connection migrator 310 is responsible for cleaning up the migration and for sending a reset notification back to client 102 to abandon the connection.

A special situation arises when application 316 and classifier 304 are co-located on the same device: migrator shim 3412 may referee between them. When both classes of programs reside on the same host 108, they may both be listening to the same IP address(es) and port(s). However, TCP typically has one listener per unique IP address and port. Consequently, migrator shim 3412 can obscure a configuration where two programs are listening on the same IP address and port by multiplexing the two sockets into a single listener at the TCP layer.

In such a case, when connect events arrive at the client portion of migrator shim 3412, migrator shim 3412 as a provider determines on which listening socket

1 to deliver the connect notification at socket layer 3402. If there is only one socket
2 listening to the corresponding IP address and port, then that socket receives the
3 connect event. If there is more than one socket listening, then the recipient
4 depends on the context in which the connect event is indicated. If the connect
5 event is a brand new connection for a virtual IP address, then the connect event is
6 delivered to classifier 304; if the connect event is for a dedicated IP address (non-
7 load-balanced IP address) or the result of uploading a migrated connection, then
8 the connect event is delivered to the target application 316.

9 At <8>, once the injection of the migrated connection is complete, TCP
10 notifies migrator shim 3412 by invoking the provided initiate inject complete
11 handler. A status code is provided to notify migrator shim 3412 whether or not the
12 connection was successfully uploaded. If uploading of the migrated connection
13 fails, connection migrator 310 is responsible for cleaning up the migration and for
14 notifying client 102 that the connection has been abandoned by sending it a reset.
15 If the migrated connection was successfully injected into the local network stack,
16 migrator intermediate driver 3414 may begin delivering any buffered data from
17 client 102 by passing the received packet(s) up through the packet receive path of
18 protocol-hardware interface 3406.

19 When a migrated connection is terminated (because uploading failed,
20 because the migrated connection is subsequently closed through normal means,
21 etc.), target device 3500 notifies originating device 3400. Originating device 3400
22 uses these notifications to more efficiently and reliably clean out lingering state for
23 migrated connections, including routing table entries. Therefore, to account for
24 successfully migrated connections which terminate arbitrarily in the future,
25

117
1 migrator shim 3412 may monitor their activity and notify migrator intermediate
2 driver 3414 when the sockets therefor are closed.

3 FIG. 38 illustrates an exemplary approach to packet tunneling between a
4 forwarder 302 and a host 108. Encapsulated packets 3808 may be tunneled from
5 forwarder 302 to host 108 without incurring overhead for each packet transmitted.
6 As described further below, the tunneling is effectuated using a flow identifier
7 3814 and encapsulation mapping tables 3806 and 3810 of tunnelers 312(F) and
8 312(H), respectively, of forwarder 302 and host 108, respectively. Flow identifier
9 3814 is inserted into encapsulated packets 3808.

10 As noted above with reference to FIG. 32, packets for a connection that
11 arrive subsequent to a connection migration may be routed by forwarder 302 to
12 host 108(1) using tunneling by a tunneler 312. At (8) (of FIG. 32), forwarder 302
13 forwards such subsequent packets from forwarder 302 having a network address
14 of "F" to host 108(1) having a network address of "H1". As described above with
15 reference to FIG. 4, forwarder 302 may perform NAT, half-NAT, tunneling, etc. in
16 order to route the incoming packets to host 108(1).

17 Such incoming packets include a destination IP address of the virtual IP
18 ("VIP") address and a source IP address of "C1" for packets arriving from client
19 102(1). The packets being routed to host 108(1) have a destination IP address of
20 H1 and a source address of C1 (for half-NAT) or "F" (for full NAT). This re-
21 writing of the addresses can interfere with some protocols that expect both of
22 client 102(1) and host 108(1) to have identical views of the source and destination
23 addresses.

24 Furthermore, at least with respect to full NAT, return paths from host
25 108(1) to client 102(1) that do not run through forwarder 302 are prohibitive

1 because host 108(1) does not know the address of client 102(1). Direct paths from
 2 host 108(1) to client 102(1) are desirable in situations in which traffic from host
 3 108(1) to client 102(1) is especially high and/or significantly greater than traffic in
 4 the opposite direction (e.g., when host 108(1) provides streaming media to client
 5 102(1)).

6 Tunneling by tunnelers 312 as described herein can provide for identical
 7 views with respect to the source and destination addresses (and ports) for clients
 8 102 and applications 316 on hosts 108. By way of example and with reference to
 9 FIGS. 34 and 35, tunneler 312 in each of forwarder 302 and host 108 may operate
 10 as part of or in conjunction with a migrator intermediate driver 3414 of a
 11 connection migrator 310.

12 In a described implementation for FIG. 38, connection migrator 310
 13 provides an encapsulation mapping 3812 between a flow identifier 3814 and a
 14 TCP/IP 4-tuple 3804. Connection migrator 310 may be associated with a classifier
 15 304, and connection migrator 310 (optionally along with such a classifier 304)
 16 may be located on a same device as forwarder 302. Alternatively, connection
 17 migrator 310 (as well as the classifier 304) may be located on a different device
 18 from forwarder 302. Encapsulation mapping 3812 may alternatively be provided
 19 by or in conjunction with tunneler 312 functionality that is, for example, located at
 20 and/or associated with a classifier 304.

21 By being mapped to a TCP/IP 4-tuple 3804 in encapsulation mapping 3812,
 22 flow identifier 3814 serves to identify a flow of encapsulated packets 3808 for a
 23 particular connection. TCP/IP 4-tuple 3804 includes network addresses (and ports,
 24 etc.) for the source and destination for a particular connection in accordance with a
 25 TCP/IP protocol, or any similar or analogous protocol. Flow identifier 3814 is 32

122

1 bits in a described implementation because 32 bits is available for connections
2 established in accordance with an internet IPv4 protocol. However, flow
3 identifiers 3814 of other lengths may alternatively be used, especially for other
4 protocols such as internet IPv6, UDP, and so forth.

5 Flow identifiers 3814 may be generated using any appropriate mechanism,
6 such as an incrementing connection counter. Furthermore, TCP/IP 4-tuple 3804 is
7 more generally a source/destination pair. Each source value and destination value
8 of an individual source/destination pair may include a network node identifier
9 (e.g., network address, port, some combination thereof, etc.) for the source and
10 destination, respectively, of a given packet propagating on a particular connection.

11 Connection migrator 310 provides encapsulation mapping 3812 to host 108.
12 Tunneler 312(H) at host 108 stores encapsulation mapping 3812 in encapsulation
13 mapping table 3810 as encapsulation mapping entry 3810(1). Tunneler 312(H)
14 can thereafter use flow identifier 3814 to map to and identify the particular
15 connection corresponding to TCP/IP 4-tuple 3804. Encapsulation mapping 3812
16 may optionally be provided to host 108 as part of a bundled binary blob 3422 in a
17 connection migration operation.

18 Forwarder 302 also includes a tunneler 312(F) component with an
19 encapsulation mapping table 3806. Encapsulation mapping table 3806 stores an
20 encapsulation mapping entry 3806(1) that links/maps TCP/IP 4-tuple 3804 for a
21 particular connection to a flow identifier 3814. Tunneler 312(F) also receives the
22 mapping information for encapsulation mapping entry 3806(1) from connection
23 migrator 310 (e.g., as an encapsulation mapping 3812).

24 Although only one encapsulation mapping entry 3806(1) and 3810(1) is
25 shown, each of encapsulation mapping table 3806 and encapsulation mapping

120
1 table 3810 may have multiple such entries. These encapsulation mapping tables
2 3806 and 3810 may be combined with other information, such as tables for session
3 information of session tracker 308.

4 When a transmitting device (such as forwarder 302) and a receiving device
5 (such as host 108) of encapsulated packets 3808 only tunnel between each other,
6 the encapsulation mapping tables thereof likely have the same encapsulation
7 mapping entries. Otherwise, encapsulation mapping table 3806 and encapsulation
8 mapping table 3810 likely have a different total set of encapsulation mapping
9 entries 3806() and encapsulation mapping entries 3810(), respectively.

10 In operation, an incoming packet 3802 for a particular connection is
11 received at forwarder 302. The particular connection is associated with TCP/IP 4-
12 tuple 3804. Incoming packet 3802 includes TCP/IP 4-tuple 3804 with a source IP
13 address (of a client 102), a destination IP address (the virtual IP), a source TCP
14 port (of the client 102), and a destination TCP port.

15 Tunneler 312(F) accepts incoming packet 3802 for tunneling to host 108.
16 Using TCP/IP 4-tuple 3804, tunneler 312(F) accesses encapsulation mapping table
17 3806 to locate encapsulation mapping entry 3806(1). Flow identifier 3814 is
18 extracted from encapsulation mapping entry 3806(1) as being linked/mapped to
19 TCP/IP 4-tuple 3804.

20 To create encapsulated packet 3808, tunneler 312(F) inserts flow identifier
21 3814 into the source and destination port portions of the TCP/IP 4-tuple header.
22 For an internet IPv4 implementation, these two TCP port portions offer 32 bits of
23 total space. Also, for the source IP address portion of the TCP/IP 4-tuple header,
24 tunneler 312(F) inserts the IP address "F" of forwarder 302. For the destination IP
25

1 address portion of the TCP/IP 4-tuple header, tunneler 312(F) inserts the IP
2 address "H" of host 108.

3 Forwarder 302 routes/transmits encapsulated packet 3808 to host 108, and
4 host 108 receives encapsulated packet 3808 from forwarder 302. The tunneler
5 312(H) component at host 108 detects that encapsulated packet 3808 is a tunneled
6 packet that is to be de-encapsulated.

7 Flow identifier 3814 is extracted from encapsulated packet 3808 and used
8 to look up the corresponding TCP/IP 4-tuple 3804 that is linked thereto in
9 encapsulation mapping entry 3810(1) of encapsulation mapping table 3810.
10 TCP/IP 4-tuple 3804 is used by tunneler 312(H) to recreate the TCP/IP 4-tuple
11 3804 header as originally received in incoming packet 3802 at forwarder 302.

12 Specifically, the IP address F of forwarder 302 is replaced with the source
13 IP address, and the IP address H of host 108 is replaced with the destination IP
14 address. Furthermore, flow identifier 3814 is replaced by the source TCP port and
15 the destination TCP port. The de-encapsulated packet is then indicated up the
16 network stack of host 108 to the targeted application 316.

17 More generally, a portion of a packet header, including a portion of a
18 source/destination pair, for a given packet that is not necessarily used for
19 communicating the given packet may be used to carry a flow identifier 3814. By
20 pre-providing at least part of the source/destination pair at host 108, a flow
21 identifier 3814 may be employed to tunnel (e.g., encapsulate and/or de-
22 encapsulate) packets without incurring an encapsulation overhead on each packet.
23 Furthermore, packets that are full-size with respect to a given protocol may be
24 tunneled without being broken apart.

25

122

1 FIG. 39 is a flow diagram 3900 that illustrates an exemplary method for
 2 packet tunneling between a first device and a second device. For example, the
 3 first device and the second device may correspond to an originating device 3400
 4 and a target device 3500, respectively, of load balancing infrastructure 106 and a
 5 cluster of hosts 108, respectively. Nevertheless, tunneling may be employed in
 6 non-load-balancing implementations.

7 Flow diagram 3900 includes twelve blocks 3902-3924. Although the
 8 actions of flow diagram 3900 may be performed in other environments and with a
 9 variety of software schemes, FIGS. 1-3, 32, 34, 35, and 38 are used in particular to
 10 illustrate certain aspects and examples of the method.

11 At block 3902, a mapping of a flow identifier-to-TCP/IP 4-tuple is sent to a
 12 target device from an originating device. For example, originating device 3400
 13 may send an encapsulation mapping 3812 that links a flow identifier 3814 to a
 14 TCP/IP 4-tuple 3804. At block 3914, the mapping of the flow identifier-to-the
 15 TCP/IP 4-tuple is received at the target device from the originating device. For
 16 example, target device 3500 receives encapsulation mapping 3812 that links flow
 17 identifier 3814 to TCP/IP 4-tuple 3804 from originating device 3400.

18 Alternatively, target device 3500 may receive encapsulation mapping 3812
 19 from another device. As indicated by dashed arrows 3926 and 3928, the actions of
 20 blocks 3904-3912 and blocks 3916-3924 can occur at some time after the actions
 21 of blocks 3902 and 3914, respectively.

22 At block 3904, an incoming packet is received at the originating device
 23 from a client. For example, an incoming packet 3802 having a header with
 24 TCP/IP 4-tuple 3804 may be received at originating device 3400 from a client 102.
 25 At block 3906, a flow identifier is looked up for a connection corresponding to the

131

1 client's packet using the TCP/IP 4-tuple of the incoming packet. For example,
 2 flow identifier 3814 may be looked up for the connection with client 102 using
 3 TCP/IP 4-tuple 3804 that is mapped thereto in an encapsulation mapping entry
 4 3806(1) of an encapsulation mapping table 3806.

5 At block 3908, the source IP and destination IP of the incoming packet are
 6 replaced with an originating IP address of the originating device and a target IP
 7 address of the target device, respectively. For example, originating device 3400
 8 may replace the IP address portions of the TCP/IP 4-tuple 3804 portion of a header
 9 of incoming packet 3802 with IP addresses of originating device 3400 and target
 10 device 3500.

11 At block 3910, the source port and the destination port of the incoming
 12 packet are replaced with the flow identifier. For example, originating device 3400
 13 may replace source and destination TCP ports of the TCP/IP 4-tuple 3804 portion
 14 of the header of incoming packet 3802 with flow identifier 3814. At block 3912,
 15 the encapsulated packet is sent from the originating device to the target device.
 16 For example, originating device 3400 may send an encapsulated packet 3808 to
 17 target device 3500.

18 At block 3916, the encapsulated packet is received at the target device from
 19 the originating device. For example, target device 3500 may receive the
 20 encapsulated packet 3808 from originating device 3400. At block 3918, the
 21 TCP/IP 4-tuple is looked up for the connection corresponding to the packet
 22 received from the client using the flow identifier. For example, target device 3500
 23 may access an encapsulation mapping table 3810 at an encapsulation mapping
 24 entry 3810(1) that maps flow identifier 3814 to TCP/IP 4-tuple 3804.

25

132

1 At block 3920, the originating IP address and the target IP address are
2 replaced with the source IP address and the destination IP address, respectively,
3 using the looked-up TCP/IP 4-tuple. For example, target device 3500 may replace
4 the IP addresses of originating device 3400 and target device 3500 in encapsulated
5 packet 3808 with the source IP address and the destination IP address from TCP/IP
6 4-tuple 3804 as attained from encapsulation mapping table 3810.

7 At block 3922, the flow identifier is replaced with the source port and the
8 destination port of the incoming packet using the looked up TCP/IP 4-tuple. For
9 example, target device 3500 may replace flow identifier 3814 in encapsulated
10 packet 3808 with the source TCP port and the destination TCP port from TCP/IP
11 4-tuple 3804. At block 3924, the client's packet is indicated up to an application at
12 the target device. For example, a de-encapsulated version of encapsulated packet
13 3808, or incoming packet 3802, is indicated up to application 316 of target device
14 3500.

15 The actions, aspects, features, components, etc. of FIGS. 1-39 are
16 illustrated in diagrams that are divided into multiple blocks. However, the order,
17 interconnections, layout, etc. in which FIGS. 1-39 are described and/or shown is
18 not intended to be construed as a limitation, and any number of the blocks can be
19 combined, rearranged, augmented, omitted, etc. in any manner to implement one
20 or more systems, methods, devices, procedures, media, APIs, apparatuses,
21 arrangements, etc. for network load balancing. Furthermore, although the
22 description herein includes references to specific implementations (and the
23 exemplary operating environment of FIG. 40), the illustrated and/or described
24 implementations can be implemented in any suitable hardware, software,
25 firmware, or combination thereof and using any suitable network organization(s),

1 transport/communication protocols(s), application programming interface(s)
2 (APIs), client-server architecture(s), and so forth.

3 **Exemplary Operating Environment for Computer or Other Device**

4 FIG. 40 illustrates an exemplary computing (or general device) operating
5 environment 4000 that is capable of (fully or partially) implementing at least one
6 system, device, apparatus, component, arrangement, protocol, approach, method,
7 procedure, media, API, some combination thereof, etc. for network load balancing
8 as described herein. Operating environment 4000 may be utilized in the computer
9 and network architectures described below or in a stand-alone situation.

10 Exemplary operating environment 4000 is only one example of an
11 environment and is not intended to suggest any limitation as to the scope of use or
12 functionality of the applicable device (including computer, network node,
13 entertainment device, mobile appliance, general electronic device, etc.)
14 architectures. Neither should operating environment 4000 (or the devices thereof)
15 be interpreted as having any dependency or requirement relating to any one or to
16 any combination of components as illustrated in FIG. 40.

17 Additionally, network load balancing may be implemented with numerous
18 other general purpose or special purpose device (including computing system)
19 environments or configurations. Examples of well known devices, systems,
20 environments, and/or configurations that may be suitable for use include, but are
21 not limited to, personal computers, server computers, thin clients, thick clients,
22 personal digital assistants (PDAs) or mobile telephones, watches, hand-held or
23 laptop devices, multiprocessor systems, microprocessor-based systems, set-top
24 boxes, programmable consumer electronics, video game machines, game consoles,
25 portable or handheld gaming units, network PCs, minicomputers, mainframe

1 computers, network nodes, distributed or multi-processing computing
2 environments that include any of the above systems or devices, some combination
3 thereof, and so forth.

4 Implementations for network load balancing may be described in the
5 general context of processor-executable instructions. Generally, processor-
6 executable instructions include routines, programs, protocols, objects, interfaces,
7 components, data structures, etc. that perform and/or enable particular tasks and/or
8 implement particular abstract data types. Network load balancing, as described in
9 certain implementations herein, may also be practiced in distributed processing
10 environments where tasks are performed by remotely-linked processing devices
11 that are connected through a communications link and/or network. Especially in a
12 distributed computing environment, processor-executable instructions may be
13 located in separate storage media, executed by different processors, and/or
14 propagated over transmission media.

15 Exemplary operating environment 4000 includes a general-purpose
16 computing device in the form of a computer 4002, which may comprise any (e.g.,
17 electronic) device with computing/processing capabilities. The components of
18 computer 4002 may include, but are not limited to, one or more processors or
19 processing units 4004, a system memory 4006, and a system bus 4008 that couples
20 various system components including processor 4004 to system memory 4006.

21 Processors 4004 are not limited by the materials from which they are
22 formed or the processing mechanisms employed therein. For example, processors
23 4004 may be comprised of semiconductor(s) and/or transistors (e.g., electronic
24 integrated circuits (ICs)). In such a context, processor-executable instructions may
25 be electronically-executable instructions. Alternatively, the mechanisms of or for

1 processors 4004, and thus of or for computer 4002, may include, but are not
2 limited to, quantum computing, optical computing, mechanical computing (e.g.,
3 using nanotechnology), and so forth.

4 System bus 4008 represents one or more of any of many types of wired or
5 wireless bus structures, including a memory bus or memory controller, a point-to-
6 point connection, a switching fabric, a peripheral bus, an accelerated graphics port,
7 and a processor or local bus using any of a variety of bus architectures. By way of
8 example, such architectures may include an Industry Standard Architecture (ISA)
9 bus, a Micro Channel Architecture (MCA) bus, an Enhanced ISA (EISA) bus, a
10 Video Electronics Standards Association (VESA) local bus, a Peripheral
11 Component Interconnects (PCI) bus also known as a Mezzanine bus, some
12 combination thereof, and so forth.

13 Computer 4002 typically includes a variety of processor-accessible media.
14 Such media may be any available media that is accessible by computer 4002 or
15 another (e.g., electronic) device, and it includes both volatile and non-volatile
16 media, removable and non-removable media, and storage and transmission media.

17 System memory 4006 includes processor-accessible storage media in the
18 form of volatile memory, such as random access memory (RAM) 4040, and/or
19 non-volatile memory, such as read only memory (ROM) 4012. A basic
20 input/output system (BIOS) 4014, containing the basic routines that help to
21 transfer information between elements within computer 4002, such as during start-
22 up, is typically stored in ROM 4012. RAM 4010 typically contains data and/or
23 program modules/instructions that are immediately accessible to and/or being
24 presently operated on by processing unit 4004.

25

1 Computer 4002 may also include other removable/non-removable and/or
2 volatile/non-volatile storage media. By way of example, FIG. 40 illustrates a hard
3 disk drive or disk drive array 4016 for reading from and writing to a (typically)
4 non-removable, non-volatile magnetic media (not separately shown); a magnetic
5 disk drive 4018 for reading from and writing to a (typically) removable, non-
6 volatile magnetic disk 4020 (e.g., a "floppy disk"); and an optical disk drive 4022
7 for reading from and/or writing to a (typically) removable, non-volatile optical
8 disk 4024 such as a CD, DVD, or other optical media. Hard disk drive 4016,
9 magnetic disk drive 4018, and optical disk drive 4022 are each connected to
10 system bus 4008 by one or more storage media interfaces 4026. Alternatively,
11 hard disk drive 4016, magnetic disk drive 4018, and optical disk drive 4022 may
12 be connected to system bus 4008 by one or more other separate or combined
13 interfaces (not shown).

14 The disk drives and their associated processor-accessible media provide
15 non-volatile storage of processor-executable instructions, such as data structures,
16 program modules, and other data for computer 4002. Although exemplary
17 computer 4002 illustrates a hard disk 4016, a removable magnetic disk 4020, and a
18 removable optical disk 4024, it is to be appreciated that other types of processor-
19 accessible media may store instructions that are accessible by a device, such as
20 magnetic cassettes or other magnetic storage devices, flash memory, compact
21 disks (CDs), digital versatile disks (DVDs) or other optical storage, RAM, ROM,
22 electrically-erasable programmable read-only memories (EEPROM), and so forth.
23 Such media may also include so-called special purpose or hard-wired IC chips. In
24 other words, any processor-accessible media may be utilized to realize the storage
25 media of the exemplary operating environment 4000.

1 Any number of program modules (or other units or sets of
2 instructions/code) may be stored on hard disk 4016, magnetic disk 4020, optical
3 disk 4024, ROM 4012, and/or RAM 4040, including by way of general example,
4 an operating system 4028; one or more application programs 4030, other program
5 modules 4032, and program data 4034.

6 A user may enter commands and/or information into computer 4002 via
7 input devices such as a keyboard 4036 and a pointing device 4038 (e.g., a
8 "mouse"). Other input devices 4040 (not shown specifically) may include a
9 microphone, joystick, game pad, satellite dish, serial port, scanner, and/or the like.
10 These and other input devices are connected to processing unit 4004 via
11 input/output interfaces 4042 that are coupled to system bus 4008. However, input
12 devices and/or output devices may instead be connected by other interface and bus
13 structures, such as a parallel port, a game port, a universal serial bus (USB) port,
14 an infrared port, an IEEE 1394 ("Firewire") interface, an IEEE 802.11 wireless
15 interface, a Bluetooth® wireless interface, and so forth.

16 A monitor/view screen 4044 or other type of display device may also be
17 connected to system bus 4008 via an interface, such as a video adapter 4046.
18 Video adapter 4046 (or another component) may be or may include a graphics
19 card for processing graphics-intensive calculations and for handling demanding
20 display requirements. Typically, a graphics card includes a graphics processing
21 unit (GPU), video RAM (VRAM), etc. to facilitate the expeditious display of
22 graphics and performance of graphics operations. In addition to monitor 4044,
23 other output peripheral devices may include components such as speakers (not
24 shown) and a printer 4048, which may be connected to computer 4002 via
25 input/output interfaces 4042.

Computer 4002 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computing device 4050. By way of example, remote computing device 4050 may be a personal computer, a portable computer (e.g., laptop computer, tablet computer, PDA, mobile station, etc.), a palm or pocket-sized computer, a watch, a gaming device, a server, a router, a network computer, a peer device, another network node, or another device type as listed above, and so forth. However, remote computing device 4050 is illustrated as a portable computer that may include many or all of the elements and features described herein with respect to computer 4002.

Logical connections between computer 4002 and remote computer 4050 are depicted as a local area network (LAN) 4052 and a general wide area network (WAN) 4054. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, the Internet, fixed and mobile telephone networks, ad-hoc and infrastructure wireless networks, other wireless networks, gaming networks, some combination thereof, and so forth. Such networks and communications connections are examples of transmission media.

When implemented in a LAN networking environment, computer 4002 is usually connected to LAN 4052 via a network interface or adapter 4056. When implemented in a WAN networking environment, computer 4002 typically includes a modem 4058 or other means for establishing communications over WAN 4054. Modem 4058, which may be internal or external to computer 4002, may be connected to system bus 4008 via input/output interfaces 4042 or any other appropriate mechanism(s). It is to be appreciated that the illustrated network connections are exemplary and that other means of establishing communication link(s) between computers 4002 and 4050 may be employed.

451

1 Furthermore, other hardware that is specifically designed for servers may
2 be employed. For example, SSL acceleration cards can be used to offload SSL
3 computations. Additionally, especially in a network load balancing operating
4 environment, TCP offload hardware and/or packet classifiers on network
5 interfaces or adapters 4056 (e.g., on network interface cards) may be installed and
6 used at server devices.

7 In a networked environment, such as that illustrated with operating
8 environment 4000, program modules or other instructions that are depicted
9 relative to computer 4002, or portions thereof, may be fully or partially stored in a
10 remote media storage device. By way of example, remote application programs
11 4060 reside on a memory component of remote computer 4050 but may be usable
12 or otherwise accessible via computer 4002. Also, for purposes of illustration,
13 application programs 4030 and other processor-executable instructions such as
14 operating system 4028 are illustrated herein as discrete blocks, but it is recognized
15 that such programs, components, and other instructions reside at various times in
16 different storage components of computing device 4002 (and/or remote computing
17 device 4050) and are executed by processor(s) 4004 of computer 4002 (and/or
18 those of remote computing device 4050).

19 Although systems, media, devices, methods, procedures, apparatuses,
20 techniques, schemes, approaches, procedures, arrangements, and other
21 implementations have been described in language specific to structural, logical,
22 algorithmic, and functional features and/or diagrams, it is to be understood that the
23 invention defined in the appended claims is not necessarily limited to the specific
24 features or diagrams described. Rather, the specific features and diagrams are
25 disclosed as exemplary forms of implementing the claimed invention.

yest

CLAIMS

1. One or more processor-accessible media comprising processor-executable instructions that, when executed, direct a system to perform actions comprising:

receiving host status information from a plurality of hosts; and
making load balancing decisions responsive to the received host status information.

2. The one or more processor-accessible media as recited in claim 1, comprising the processor-executable instructions that, when executed, direct the system to perform a further action comprising:

receiving a request for a new connection from a client; and
wherein the action of making comprises an action of:
selecting a destination target for the new connection responsive to the received host status information.

3. The one or more processor-accessible media as recited in claim 1, wherein the host status information comprises health and/or load information that is associated with the plurality of hosts, and the action of receiving comprises an action of:

receiving the health and/or load information from the plurality of hosts.

241

1 4. The one or more processor-accessible media as recited in claim 1,
2 wherein the host status information comprises application-specific host status
3 information, and the action of making comprises an action of:

4 making load balancing decisions responsive to the received
5 application-specific host status information.

6
7 5. The one or more processor-accessible media as recited in claim 1,
8 wherein the action of receiving comprises at least one action of:

9 receiving the host status information directly from one or more hosts
10 of the plurality of hosts; and

11 receiving the host status information indirectly from one or more
12 hosts of the plurality of hosts.

13
14 6. A system comprising:
15 at least one processor; and

16 one or more media including processor-executable instructions that are
17 capable of being executed by the at least one processor, the processor-executable
18 instructions adapted to direct the system to perform actions comprising:

19 receiving host status information from a plurality of hosts; and

20 making load balancing decisions responsive to the received host
21 status information.

22
23
24
25

1 7. The system as recited in claim 6, wherein the action of receiving
2 comprises an action of:

3 receiving the host status information from the plurality of
4 hosts via at least one proxy.

5
6 8. The system as recited in claim 6, wherein host status information
7 includes at least one load balancing directive; and wherein the action of receiving
8 comprises an action of:

9 receiving the at least one load balancing directive from the
10 plurality of hosts via at least one proxy that is invoking one or more
11 application programming interfaces (APIs) to push the at least one
12 load balancing directive.

13
14 9. The system as recited in claim 6, wherein the system comprises at
15 least one of a single device and a plurality of devices.

16
17 10. A method for network load balancing, the method comprising:
18 determining host status information for a plurality of hosts;
19 sending the host status information from the plurality of hosts to at least
20 one device comprising load balancing infrastructure;
21 receiving the host status information from the plurality of hosts at the at
22 least one device; and
23 making load balancing decisions with the load balancing infrastructure
24 responsive to the received host status information.
25

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

11. One or more processor-accessible media comprising processor-executable instructions that, when executed, direct a system to perform actions comprising:

- accumulating host status information at a plurality of hosts; and
- sending the accumulated host status information from the plurality of hosts.

12. The one or more processor-accessible media as recited in claim 11, wherein the host status information comprises health and/or load information that is associated with the plurality of hosts and the action of sending comprises an action of:

- sending the health and/or load information from the plurality of hosts.

13. The one or more processor-accessible media as recited in claim 11, wherein the action of accumulating comprises an action of:

- accumulating the host status information with regard to applications executing on the plurality of hosts.

14. The one or more processor-accessible media as recited in claim 11, wherein the action of sending comprises an action of:

- sending the accumulated host status information from the plurality of hosts to at least one proxy.

4/4

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

15. The one or more processor-accessible media as recited in claim 11,
wherein the action of sending comprises an action of:
sending the accumulated host status information from the plurality of
hosts to load balancing functionality.

16. A system comprising:
a host status information determiner that is adapted to determine host status
information for a host; and
a host status information disseminator that is adapted to disseminate the
host status information to load balancing infrastructure.

17. The system as recited in claim 16, wherein the system further
comprises:
one or more applications;
wherein the host status information determiner determines the host status
information with regard to the one or more applications.

18. The system as recited in claim 16, wherein the system comprises a
device on which the host is resident.

141

1 19. The system as recited in claim 16, wherein the system further
2 comprises:

3 another host status information determiner that is adapted to determine
4 additional host status information for another host; and

5 another host status information disseminator that is adapted to disseminate
6 the additional host status information to the load balancing infrastructure;

7 wherein the system comprises a first device on which the host is resident
8 and a second device on which the other host is resident.

9
10 20. One or more processor-accessible media comprising processor-
11 executable instructions that, when executed, enable a system to implement a
12 message protocol between at least one host and one or more load balancing units,
13 the message protocol usable for communicating health and/or load information
14 between the at least one host and the one or more load balancing units.

15
16 21. The one or more processor-accessible media as recited in claim 20,
17 wherein at least a portion of the processor-executable instructions comprise
18 software that is adapted to be executed on the at least one host.

19
20 22. The one or more processor-accessible media as recited in claim 20,
21 wherein at least a portion of the processor-executable instructions comprise
22 software that is adapted to be executed on the one or more load balancing units.

1 **23.** The one or more processor-accessible media as recited in claim 20,
2 comprising the processor-executable instructions that, when executed, enable the
3 system to implement the message protocol that includes at least one message, the
4 at least one message comprising a heartbeat message that indicates to the one or
5 more load balancing units that the at least one host is functioning.

6
7 **24.** The one or more processor-accessible media as recited in claim 23,
8 wherein a format of the heartbeat message comprises an identifier for the at least
9 one host, error checking data for health and/or load information, and a domain
10 name system (DNS) name.

11
12 **25.** The one or more processor-accessible media as recited in claim 23,
13 wherein a format of the heartbeat message permits inclusion of a chunk
14 number/generation identifier (ID) pair.

15
16 **26.** The one or more processor-accessible media as recited in claim 20,
17 comprising the processor-executable instructions that, when executed, enable the
18 system to implement the message protocol that includes at least one message, the
19 at least one message comprising a goodbye message that indicates to the one or
20 more load balancing units that the at least one host is planning to shutdown.

21
22 **27.** The one or more processor-accessible media as recited in claim 26,
23 wherein a format of the goodbye message comprises an identifier for the at least
24 one host.

25

VAT

1 28. The one or more processor-accessible media as recited in claim 20,
2 comprising the processor-executable instructions that, when executed, enable the
3 system to implement the message protocol that includes at least one message, the
4 at least one message comprising a row change message that indicates to the one or
5 more load balancing units that health and/or load information for an application of
6 the at least one host has changed.

7
8 29. The one or more processor-accessible media as recited in claim 28,
9 wherein a format of the row change message comprises an identifier for the at
10 least one host, an identifier for the application, an operation to reflect the change,
11 and data for the operation.

12
13 30. The one or more processor-accessible media as recited in claim 20,
14 comprising the processor-executable instructions that, when executed, enable the
15 system to implement the message protocol that includes at least one message, the
16 at least one message comprising a get table snapshot message that is sent from the
17 one or more load balancing units to the at least one host, the get table snapshot
18 message requesting a snapshot of current health and/or load information of the at
19 least one host.

20
21 31. The one or more processor-accessible media as recited in claim 30,
22 wherein a format of the get table snapshot message comprises an identification of
23 a requesting load balancing unit of the one or more load balancing units.

1 **32.** The one or more processor-accessible media as recited in claim 20,
2 comprising the processor-executable instructions that, when executed, enable the
3 system to implement the message protocol that includes at least one message, the
4 at least one message comprising a send table snapshot message that is sent from
5 the at least one host to a requesting load balancing unit of the one or more load
6 balancing units, the send table snapshot message providing a snapshot of current
7 health and/or load information of the at least one host.

8
9 **33.** The one or more processor-accessible media as recited in claim 32,
10 wherein a format of the send table snapshot message comprises the snapshot of the
11 current health and/or load information of the at least one host.

12
13 **34.** The one or more processor-accessible media as recited in claim 20,
14 comprising the processor-executable instructions that, when executed, enable the
15 system to implement the message protocol that includes at least one message, the
16 at least one message comprising a postulate table state message that is sent from
17 the at least one host to the one or more load balancing units, the postulate table
18 state message including a load balancing state directive that indicates a current
19 load balancing state directive that is expected by the at least one host to be existing
20 at the one or more load balancing units.

21
22 **35.** The one or more processor-accessible media as recited in claim 34,
23 wherein a format of the postulate table state message comprises an identifier for
24 the at least one host and the current load balancing state directive.

25

1 **36.** The one or more processor-accessible media as recited in claim 20,
2 comprising the processor-executable instructions that, when executed, enable the
3 system to implement the message protocol that includes at least one message, the
4 at least one message comprising a postulate wrong message that is sent from a
5 load balancing unit of the one or more load balancing units to the at least one host,
6 which previously sent a postulate table state message; the postulate wrong
7 message indicating that the load balancing unit has an actual load balancing state
8 directive that differs from a postulated load balancing state directive that is
9 included in the postulate table state message.

10
11 **37.** A system comprising:
12 at least one device that is hosting one or more applications, the at least one
13 device including a health and load table that includes a plurality of entries, each
14 entry of the plurality of entries associated with an application of the one or more
15 applications; each entry of the plurality of entries comprising:
16 an application identifier for a particular application of the one or
17 more applications;
18 information characterizing at least one status of the particular
19 application; and
20 at least one load balancing directive regarding the particular
21 application.

22
23
24
25

138

1 **38.** The system as recited in claim 37, wherein the application identifier
2 uniquely identifies the particular application from among the one or more
3 applications.

4
5 **39.** The system as recited in claim 37, wherein the application identifier
6 comprises at least one of a virtual internet protocol (IP) address and port, a
7 physical IP address and port, a protocol relevant to the particular application, and
8 information that is specific to the protocol.

9
10 **40.** The system as recited in claim 37, wherein the application identifier
11 comprises at least one globally unique identifier (GUID).

12
13 **41.** The system as recited in claim 37, wherein the information
14 characterizing at least one status of the particular application comprises at least
15 one of application health, application load, and application capacity.

16
17 **42.** The system as recited in claim 41, wherein the application health
18 indicates whether a status of the particular application is healthy, failing, or
19 unknown; wherein the application load indicates how occupied the particular
20 application is; and wherein the application capacity indicates a maximum capacity
21 of the particular application.

4

1 43. The system as recited in claim 42, wherein the maximum capacity of
2 the particular application is expressed relative to a total capacity for applications
3 of a same type as that of the particular application that are executing in the system.
4

5 44. The system as recited in claim 42, wherein the maximum capacity of
6 the particular application is expressed as a unit-less and bounded number.
7

8 45. The system as recited in claim 37, wherein the at least one load
9 balancing directive may be provided to a plurality of load balancing units to
10 provide guidance for network load balancing regarding the particular application
11 and with respect to other applications of a same application type.
12

13 46. The system as recited in claim 37, wherein the at least one load
14 balancing directive comprises at least one of active, draining, and inactive.
15

16 47. The system as recited in claim 37, wherein the at least one load
17 balancing directive comprises a target load balancing state directive and a current
18 load balancing state directive.
19

20 48. The system as recited in claim 47, wherein the target load balancing
21 state directive indicates a load balancing state that health and load infrastructure,
22 which is located at the at least one device, intends for health and load
23 infrastructure, which is located at a plurality of load balancing units, to be
24 operating under.
25

1 **49.** The system as recited in claim 47, wherein the current load
2 balancing state directive indicates a load balancing state that health and load
3 infrastructure, which is located at the at least one device, believes health and load
4 infrastructure, which is located at a plurality of load balancing units, to be
5 currently operating under.

6
7 **50.** The system as recited in claim 37, wherein the system further
8 comprises:

9 a plurality of devices, with each respective device of the plurality of devices
10 including a respective health and load table.

11
12 **51.** One or more processor-accessible media comprising processor-
13 executable instructions that comprise at least part of an operating system, wherein
14 the processor-executable instructions, when executed, direct a system to perform
15 actions comprising:

16 determining when an application is started;
17 reporting, responsive to the determining, the application as started to health
18 and load infrastructure that is capable of communicating with network load
19 balancing infrastructure;

20 detecting when the application is stopped; and
21 reporting, responsive to the detecting, the application as stopped to the
22 health and load infrastructure.

23
24
25

1 **52.** The one or more processor-accessible media as recited in claim 51,
 2 wherein at least a portion of the processor-executable instructions comprise at
 3 least part of a control manager of the operating system.

4
 5 **53.** One or more processor-accessible media comprising processor-
 6 executable instructions that comprise at least part of an operating system, wherein
 7 the processor-executable instructions, when executed, direct a system to perform
 8 actions comprising:

9 identifying that an application is started as a result of starting the
 10 application;

11 reporting, responsive to the identifying, the application as started to health
 12 and load infrastructure that is capable of communicating with network load
 13 balancing infrastructure;

14 establishing that the application is stopped as a result of stopping the
 15 application; and

16 reporting, responsive to the establishing, the application as stopped to the
 17 health and load infrastructure.

18
 19 **54.** The one or more processor-accessible media as recited in claim 53,
 20 wherein at least a portion of the processor-executable instructions comprise at
 21 least part of a control manager of the operating system.

22
 23
 24
 25



1 **55.** A system comprising:

2 health and load infrastructure that is adapted to determine application-
3 specific health and load information; and

4 load balancing infrastructure that is adapted to utilize the application-
5 specific health and load information when allocating requests to a plurality of
6 applications.

7
8 **56.** The system as recited in claim 55, wherein the health and load
9 infrastructure comprises a health and load table that stores at least a portion of the
10 application-specific health and load information.

11
12 **57.** The system as recited in claim 55, wherein the health and load
13 infrastructure comprises a health and load table that stores at least a portion of the
14 application-specific health and load information; the health and load table
15 including a plurality of entries, each entry of the plurality of entries associated
16 with a particular application of the plurality of applications.

1 **58.** The system as recited in claim 55, wherein the health and load
2 infrastructure comprises a health and load table that stores at least a portion of the
3 application-specific health and load information; the health and load table
4 including a plurality of entries, each entry of the plurality of entries including: an
5 application identifier for a particular application with which the entry is
6 associated, information characterizing at least one status of the particular
7 application, and at least one load balancing directive with regard to the particular
8 application.

9
10 **59.** The system as recited in claim 55, wherein the load balancing
11 infrastructure comprises a consolidated health and load cache that stores the
12 application-specific health and load information.

13
14 **60.** The system as recited in claim 55, wherein the load balancing
15 infrastructure comprises a consolidated health and load cache that stores the
16 application-specific health and load information for the plurality of applications
17 that are executing on a plurality of hosts.

18
19 **61.** The system as recited in claim 55, wherein the application-specific
20 health and load information comprises application-endpoint-specific health and
21 load information.

22
23
24
25



1 **62.** The system as recited in claim 55, further comprising:
2 a proxy device that comprises at least part of the health and load
3 infrastructure, the at least part of the health and load infrastructure adapted to
4 determine the application-specific health and load information by performing
5 external monitoring actions.

6
7 **63.** The system as recited in claim 55, wherein:
8 the health and load infrastructure comprises a plurality of health and load
9 tables that store the application-specific health and load information; and
10 the load balancing infrastructure comprises a plurality of consolidated
11 health and load caches that store the application-specific health and load
12 information.

13
14 **64.** The system as recited in claim 63, wherein the system further
15 comprises:
16 a plurality of hosts upon which the health and load infrastructure is
17 distributed over, each host of the plurality of hosts having a health and load table
18 of the plurality of health and load tables; and
19 a plurality of load balancing units corresponding to at least a portion of the
20 load balancing infrastructure, each load balancing unit of the plurality of load
21 balancing units having a consolidated health and load cache of the plurality of
22 consolidated health and load caches.

23
24
25



1 65. The system as recited in claim 63, wherein the system further
2 comprises:

3 a plurality of hosts upon which the health and load infrastructure is
4 distributed over, each host of the plurality of hosts having a health and load table
5 of the plurality of health and load tables; and

6 a plurality of load balancing units corresponding to at least a portion of the
7 load balancing infrastructure, each load balancing unit of the plurality of load
8 balancing units having a consolidated health and load cache of the plurality of
9 consolidated health and load caches;

10 wherein each consolidated health and load cache of the plurality of health
11 and load caches includes the application-specific health and load information that
12 is stored on each health and load table of the plurality of health and load tables.

13
14 66. The system as recited in claim 63, wherein the system further
15 comprises:

16 a plurality of hosts upon which the health and load infrastructure is
17 distributed over, each host of the plurality of hosts having a health and load table
18 of the plurality of health and load tables; and

19 a plurality of load balancing units corresponding to at least a portion of the
20 load balancing infrastructure, each load balancing unit of the plurality of load
21 balancing units having a consolidated health and load cache of the plurality of
22 consolidated health and load caches;

23 wherein the plurality of applications are executing on the plurality of hosts.

24
25

13
14
15

1 67. The system as recited in claim 63, wherein the system further
2 comprises:

3 a plurality of hosts resident at a plurality of devices, the health and load
4 infrastructure distributed over the plurality of hosts, each host of the plurality of
5 hosts having a health and load table of the plurality of health and load tables; and

6 a plurality of load balancing units comprised of at least one device and
7 corresponding to at least a portion of the load balancing infrastructure, each load
8 balancing unit of the plurality of load balancing units having a consolidated health
9 and load cache of the plurality of consolidated health and load caches.

10
11 68. The system as recited in claim 63, wherein the system further
12 comprises:

13 a plurality of hosts resident at a plurality of devices, the health and load
14 infrastructure distributed over the plurality of hosts, each host of the plurality of
15 hosts having a health and load table of the plurality of health and load tables; and

16 a plurality of load balancing units comprised of at least one device and
17 corresponding to at least a portion of the load balancing infrastructure, each load
18 balancing unit of the plurality of load balancing units having a consolidated health
19 and load cache of the plurality of consolidated health and load caches;

20 wherein the health and load infrastructure comprises a remote part of the
21 load balancing infrastructure.

22
23
24
25

15/11

1 69. The system as recited in claim 63, wherein the system further
2 comprises:

3 a plurality of hosts resident at a plurality of devices, the health and load
4 infrastructure distributed over the plurality of hosts, each host of the plurality of
5 hosts having a health and load table of the plurality of health and load tables; and

6 a plurality of load balancing units comprised of at least one device and
7 corresponding to at least a portion of the load balancing infrastructure, each load
8 balancing unit of the plurality of load balancing units having a consolidated health
9 and load cache of the plurality of consolidated health and load caches;

10 wherein the at least one device is one of the plurality of devices.

11
12 70. The system as recited in claim 63, wherein the system further
13 comprises:

14 a plurality of hosts resident at a plurality of devices, the health and load
15 infrastructure distributed over the plurality of hosts, each host of the plurality of
16 hosts having a health and load table of the plurality of health and load tables; and

17 a plurality of load balancing units comprised of at least one device and
18 corresponding to at least a portion of the load balancing infrastructure, each load
19 balancing unit of the plurality of load balancing units having a consolidated health
20 and load cache of the plurality of consolidated health and load caches;

21 wherein the at least one device is not one of the plurality of devices; and

22 wherein the health and load infrastructure is further adapted to disseminate
23 the application-specific health and load information from the plurality of devices
24 to the at least one device.

25

1 71. The system as recited in claim 55, wherein the health and load
2 infrastructure and the load balancing infrastructure are capable of using a message
3 protocol for communications therebetween that relate to the application-specific
4 health and load information.

5
6 72. The system as recited in claim 71, wherein the message protocol
7 comprises one or more of the following message types: a heartbeat message type,
8 a goodbye message type, a row change message type, a get table snapshot message
9 type, a send table snapshot message type, a postulate table state message type, and
10 a postulate wrong message type.

11
12 73. The system as recited in claim 71, wherein the message protocol
13 includes a capability for communication using group membership.

14
15 74. The system as recited in claim 55, wherein the load balancing
16 infrastructure, after a failure, is capable of recovering the application-specific
17 health and load information via the health and load infrastructure using a message
18 protocol for communications therebetween.

19
20 75. The system as recited in claim 55, wherein the load balancing
21 infrastructure is further adapted to allocate requests to the plurality of applications
22 using one or more allotment schemes.

23
24
25

1 76. The system as recited in claim 75, wherein the one or more
2 allotment schemes comprise at least one of a token allotment scheme and a
3 percentage allotment scheme.

4
5 77. The system as recited in claim 75, wherein the one or more
6 allotment schemes entail using a timer expiration mechanism.

7
8 78. The system as recited in claim 75, wherein the one or more
9 allotment schemes are implemented by a health and load handler component of the
10 load balancing infrastructure.

11
12 79. One or more processor-accessible media comprising processor-
13 executable instructions that, when executed, direct a system to perform actions
14 comprising:

15 analyzing health and/or load information for a plurality of application
16 endpoints; and

17 ascertaining a token allotment for the plurality of application endpoints
18 responsive to the analyzing.

19
20
21
22
23
24
25

162

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

80. The one or more processor-accessible media as recited in claim 79, comprising the processor-executable instructions that, when executed, direct the system to perform further actions comprising:

- receiving a target application endpoint allotment request that identifies the plurality of application endpoints; and
- sending a target application endpoint allotment response that includes the token allotment.

81. The one or more processor-accessible media as recited in claim 79, comprising the processor-executable instructions that, when executed, direct the system to perform further actions comprising:

- receiving a target application endpoint allotment request that includes one or more of a virtual internet protocol (IP) address and port, a protocol, and information that is specific to the protocol; and
- sending a target application endpoint allotment response that includes a physical IP address and port.

82. The one or more processor-accessible media as recited in claim 79, wherein the action of analyzing comprises an action of:

- analyzing application-endpoint-specific health and/or load information for the plurality of application endpoints.

EB

455
155

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

83. The one or more processor-accessible media as recited in claim 79,
wherein the action of ascertaining comprises an action of:

ascertaining the token allotment for the plurality of application
endpoints based on relative available capacities between or among the
plurality of application endpoints.

84. The one or more processor-accessible media as recited in claim 79,
wherein the token allotment comprises a first number of tokens corresponding to a
first application endpoint of the plurality of application endpoints and a second
number of tokens corresponding to a second application endpoint of the plurality
of application endpoints.

85. The one or more processor-accessible media as recited in claim 79,
wherein the token allotment comprises:

- a first number of tokens corresponding to a first application endpoint
of the plurality of application endpoints;
- a second number of tokens corresponding to a second application
endpoint of the plurality of application endpoints; and
- a time limit, wherein an expiration of the time limit renders unusable
any remaining tokens of the first number of tokens and the second number
of tokens.

1 **86.** The one or more processor-accessible media as recited in claim 79,
2 comprising the processor-executable instructions that, when executed, direct the
3 system to perform a further action comprising:

4 using the token allotment to classify incoming connection requests.

5
6 **87.** One or more processor-accessible media comprising processor-
7 executable instructions that, when executed, direct a system to perform actions
8 comprising:

9 determining health and load information on a per application basis; and

10 selecting an application from among a plurality of applications responsive
11 to the health and load information.

12
13 **88.** The one or more processor-accessible media as recited in claim 87,
14 wherein the action of determining comprises an action of:

15 determining when applications of the plurality of applications start
16 and stop.

17
18 **89.** The one or more processor-accessible media as recited in claim 87,
19 wherein the action of determining comprises an action of:

20 determining when an application of the plurality of applications is
21 healthy and when the application is failing or failed.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

90. The one or more processor-accessible media as recited in claim 87,
wherein the action of determining comprises an action of:

determining a load of a given application of a particular application
type relative to loading of one or more other applications of the particular
application type.

91. The one or more processor-accessible media as recited in claim 87,
comprising the processor-executable instructions that, when executed, direct the
system to perform a further action comprising:

receiving external input regarding determination of the health and load
information on a per application basis;

wherein the action of determining comprises an action of:

determining the health and load information on a per application
basis in accordance with the external input.

92. The one or more processor-accessible media as recited in claim 87,
comprising the processor-executable instructions that, when executed, direct the
system to perform a further action comprising:

disseminating the health and load information from at least one host to one
or more load balancing units.

YB

1 **93.** The one or more processor-accessible media as recited in claim 87,
2 comprising the processor-executable instructions that, when executed, direct the
3 system to perform a further action comprising:

4 disseminating the health and load information from at least one host to one
5 or more load balancing units using a message protocol.

6
7 **94.** The one or more processor-accessible media as recited in claim 87,
8 comprising the processor-executable instructions that, when executed, direct the
9 system to perform a further action comprising:

10 disseminating the health and load information from at least one host to one
11 or more load balancing units using a membership grouping.

12
13 **95.** The one or more processor-accessible media as recited in claim 94,
14 wherein the action of disseminating comprises an action of:

15 sending a heartbeat message from the at least one host to a
16 leadership host, wherein the heartbeat message includes a forwarding
17 indication such that the leadership host is directed to forward the heartbeat
18 message to the one or more load balancing units even if no change in the
19 membership grouping is involved.

20
21
22
23
24
25

1 96. The one or more processor-accessible media as recited in claim 87,
2 comprising the processor-executable instructions that, when executed, direct the
3 system to perform a further action comprising:

4 disseminating the health and load information from at least one health and
5 load table to one or more consolidated health and load caches.

6
7 97. The one or more processor-accessible media as recited in claim 87,
8 comprising the processor-executable instructions that, when executed, direct the
9 system to perform further actions comprising:

10 receiving the health and load information from a plurality of hosts; and
11 caching the health and load information.

12
13 98. The one or more processor-accessible media as recited in claim 87,
14 comprising the processor-executable instructions that, when executed, direct the
15 system to perform further actions comprising:

16 receiving the health and load information from a plurality of hosts;
17 caching the received health and load information;
18 receiving a packet requesting a connection initiation; and
19 consulting the cached health and load information for the connection
20 initiation;

21 wherein the action of selecting comprises an action of:

22 selecting the application from among the plurality of applications
23 responsive to the consulting.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

99. The one or more processor-accessible media as recited in claim 98,
wherein the connection initiation pertains to a particular application type.

100. The one or more processor-accessible media as recited in claim 87,
wherein the action of selecting comprises an action of:
selecting an application endpoint from among a plurality of
application endpoints responsive to the health and load information.

101. The one or more processor-accessible media as recited in claim 87,
wherein the action of selecting comprises an action of:
selecting, responsive to the health and load information, an
application endpoint from among a plurality of application endpoints that
are distributed among a plurality of hosts.

102. The one or more processor-accessible media as recited in claim 87,
wherein the one or more processor-accessible media comprise at least one of (i)
one or more storage media and (ii) one or more transmission media.

103. The one or more processor-accessible media as recited in claim 87,
wherein at least a portion of the processor-executable instructions comprise at
least part of a load balancing software program.

UP

1 **104.** The one or more processor-accessible media as recited in claim 87,
2 wherein at least a portion of the processor-executable instructions are adapted to
3 run on the system wherein the system comprises a single device.

4
5 **105.** The one or more processor-accessible media as recited in claim 87,
6 wherein at least a portion of the processor-executable instructions are adapted to
7 run on the system wherein the system comprises a plurality of devices.

8
9 **106.** The one or more processor-accessible media as recited in claim 87,
10 wherein the action of selecting comprises an action of:

11 selecting, responsive to the health and load information, an allotment
12 of application endpoints from among a plurality of application endpoints
13 with regard to relative available capacities between or among the plurality
14 of application endpoints.

15
16 **107.** The one or more processor-accessible media as recited in claim
17 106, wherein the action of selecting comprises a further action of:

18 selecting the allotment of application endpoints using a token
19 allotment scheme.

20
21 **108.** The one or more processor-accessible media as recited in claim
22 106, wherein the action of selecting comprises a further action of:

23 selecting the allotment of application endpoints using a percentage
24 allotment scheme.

25

1 **109.** The one or more processor-accessible media as recited in claim
2 106, wherein the plurality of application endpoints correspond to applications of a
3 single application type.

4
5 **110.** The one or more processor-accessible media as recited in claim 87,
6 wherein the action of selecting comprises an action of:

7 selecting the application from among the plurality of applications
8 responsive to the health and load information in order to balance a network
9 load caused by incoming packets.

10
11 **111.** The one or more processor-accessible media as recited in claim 87,
12 wherein the action of selecting comprises an action of:

13 selecting the application from among the plurality of applications
14 responsive to the health and load information in order to balance a network
15 load caused by incoming connection requests.

16
17 **112.** An arrangement for network load balancing with host status
18 information, the arrangement comprising:

19 means for determining host status information for a plurality of hosts; and
20 means for making load balancing decisions responsive to the host status
21 information.

22
23
24
25

1 113. The arrangement as recited in claim 112, wherein the means for
2 determining is located at the plurality of hosts, and the means for making is
3 located at load balancing infrastructure; and

4 wherein the arrangement further comprises:
5 means for disseminating the host status information from the plurality of
6 hosts to the load balancing infrastructure.

7
8 114. The arrangement as recited in claim 112, wherein the means for
9 determining comprises means for determining application-specific host status
10 information for the plurality of hosts, and the means for making comprises means
11 for making the load balancing decisions responsive to the application-specific host
12 status information.

13
14 115. The arrangement as recited in claim 112, wherein the arrangement
15 comprises at least one system.

16
17 116. The arrangement as recited in claim 112, wherein the arrangement
18 comprises one or more processor-accessible media.

19
20 117. An arrangement comprising:
21 accumulation means for accumulating host status information at a plurality
22 of hosts; and
23 sending means for sending the accumulated host status information from
24 the plurality of hosts.

25

1 **118.** The arrangement as recited in claim 117, wherein the host status
2 information comprises health and/or load information that is associated with the
3 plurality of hosts and the sending means comprises:

4 means for sending the health and/or load information from the
5 plurality of hosts.

6
7 **119.** The arrangement as recited in claim 117, wherein the accumulation
8 means comprises:

9 means for accumulating the host status information with regard to
10 applications executing on the plurality of hosts.

11
12 **120.** The arrangement as recited in claim 117, wherein the sending
13 means comprises:

14 means for sending the accumulated host status information from the
15 plurality of hosts to load balancing functionality.

16
17 **121.** An arrangement comprising:
18 determination means for determining health and load information on a per
19 application basis; and

20 selection means for selecting an application from among a plurality of
21 applications responsive to the health and load information.

22
23
24
25

173

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

122. The arrangement as recited in claim 121, wherein the determination means comprises:

means for determining when an application of the plurality of applications is healthy and when the application is failing or failed.

123. The arrangement as recited in claim 121, wherein the determination means comprises:

means for determining a load of a given application of a particular application type relative to loading of one or more other applications of the particular application type.

124. The arrangement as recited in claim 121, wherein the selection means comprises:

means for selecting an application endpoint from among a plurality of application endpoints responsive to the health and load information.

12/24

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

ABSTRACT

In a first exemplary media implementation, one or more processor-accessible media include processor-executable instructions that, when executed, direct a system to perform actions that include: accumulating host status information at multiple hosts; and sending the accumulated host status information from the multiple hosts. In a second exemplary media implementation, one or more processor-accessible media include processor-executable instructions that, when executed, direct a system to perform actions that include: receiving host status information from multiple hosts; and making load balancing decisions responsive to the received host status information. In a third exemplary media implementation, one or more processor-accessible media include processor-executable instructions that, when executed, direct a system to perform actions that include: determining health and load information on a per application basis; and selecting an application from among multiple applications responsive to the health and load information.

167

LEANN M. SASSMAN
Notary Public
State of Washington
Commission Expires MAY 25, 2008

My appointment expires 9/29/05



UNITED STATES PATENT AND TRADEMARK OFFICE

UNDER SECRETARY OF COMMERCE FOR INTELLECTUAL PROPERTY AND
DIRECTOR OF THE UNITED STATES PATENT AND TRADEMARK OFFICE

FEBRUARY 11, 2004

PTAS

LEE & HAYES, PLLC
KEITH W. SAUNDERS
421 WEST RIVERSIDE AVENUE
SUITE 500
SPOKANE, WA 99201

**CORRECTED
NOTICE**

**UNITED STATES PATENT AND TRADEMARK OFFICE
NOTICE OF RECORDATION OF ASSIGNMENT DOCUMENT**

THE ENCLOSED DOCUMENT HAS BEEN RECORDED BY THE ASSIGNMENT DIVISION OF THE U.S. PATENT AND TRADEMARK OFFICE. A COMPLETE MICROFILM COPY IS AVAILABLE AT THE ASSIGNMENT SEARCH ROOM ON THE REEL AND FRAME NUMBER REFERENCED BELOW.

PLEASE REVIEW ALL INFORMATION CONTAINED ON THIS NOTICE. THE INFORMATION CONTAINED ON THIS RECORDATION NOTICE REFLECTS THE DATA PRESENT IN THE PATENT AND TRADEMARK ASSIGNMENT SYSTEM. IF YOU SHOULD FIND ANY ERRORS OR HAVE QUESTIONS CONCERNING THIS NOTICE, YOU MAY CONTACT THE EMPLOYEE WHOSE NAME APPEARS ON THIS NOTICE AT 703-308-9723. PLEASE SEND REQUEST FOR CORRECTION TO: U.S. PATENT AND TRADEMARK OFFICE, ASSIGNMENT DIVISION, BOX ASSIGNMENTS, CG-4, 1213 JEFFERSON DAVIS HWY, SUITE 320, WASHINGTON, D.C. 20231.

RECORDATION DATE: 02/04/2004

REEL/FRAME: 014307/0355
NUMBER OF PAGES: 4

BRIEF: ASSIGNMENT OF ASSIGNOR'S INTEREST (SEE DOCUMENT FOR DETAILS).

ASSIGNOR:

DARLING, CHRISTOPHER L.

DOC DATE: 08/26/2003

ASSIGNOR:

JOY, JOSEPH M.

DOC DATE: 08/26/2003

ASSIGNOR:

SHRIVASTAVA, SUNITA

DOC DATE: 08/26/2003

ASSIGNOR:

SUBBARAMAN, CHITTUR

DOC DATE: 08/26/2003

ASSIGNEE:

MICROSOFT CORPORATION
ONE MICROSOFT WAY
REDMOND, WASHINGTON 98052

7

014307/0355 PAGE 2

SERIAL NUMBER: 10610519
PATENT NUMBER:

FILING DATE: 06/30/2003
ISSUE DATE:

DOROTHY RILEY, PARALEGAL
ASSIGNMENT DIVISION
OFFICE OF PUBLIC RECORDS

02/04/2004
700064305

Form PTO-1095 (Rev. 03/01)		RECORDATION FORM COVER SHEET PATENTS ONLY		U.S. DEPARTMENT OF COMMERCE U.S. Patent and Trademark Office	
ONS No. 0881-0027 (exp. 8/31/2002)					
To the Honorable Commissioner of Patents and Trademarks: Please record the attached original documents or copy thereof.					
1. Name of conveying party(ies): Christopher L. Darling Joseph M. Joy Kurda Shrivastava Chittur Subbaraman			2. Name and address of receiving party(ies) Name: <u>Microsoft Corporation</u> Internal Address: _____ _____ _____ Street Address: <u>One Microsoft Way</u> _____ _____ City: <u>Redmond</u> State: <u>WA</u> Zip: <u>98052</u>		
Additional name(s) of conveying party(ies) attached? ☐ Yes ☒ No			Additional name(s) & address(es) attached? ☐ Yes ☒ No		
3. Nature of conveyance: ☒ Assignment ☐ Merger ☐ Security Agreement ☐ Change of Name ☐ Other _____			Execution Date: <u>02/04/2004</u>		
4. Application number(s) or patent number(s): <u>10/610,519</u> If this document is being filed together with a new application, the execution date of the application is: _____ A. Patent Application No.(s) _____ B. Patent No.(s) _____ Additional numbers attached? ☐ Yes ☒ No					
5. Name and address of party to whom correspondence concerning document should be mailed: Name: <u>Kath W. Saunders, Reg. No. 41,482</u> Internal Address: <u>Lee & Hayes, PLLC</u> _____ _____ Street Address: <u>421 West Riverside Avenue</u> _____ _____ Bldg 500 City: <u>Sedro Blaine</u> State: <u>WA</u> Zip: <u>98201</u>			6. Total number of applications and patents involved: <u>1</u> 7. Total fee (37 CFR 3.41).....\$ <u>40.00</u> ☐ Enclosed ☒ Authorized to be charged to deposit account 8. Deposit account number: <u>12-0768</u> (Attach duplicate copy of this page if paying by deposit account)		
DO NOT USE THIS SPACE					
9. Statement and signature. To the best of my knowledge and belief, the foregoing information is true and correct and any attached copy is a true copy of the original document. <u>Kath W. Saunders, Reg. No. 41,482</u> <u>Kath W. Saunders</u> <u>02/04/2004</u> Name of Person Signing Signature Date Total number of pages (including cover sheet, attachments, and documents): <u>4</u>					

Mail documents to be recorded with required cover sheet information to:
Mail Stop Assignment Recordation Service
P.O. Box 20, VA 22113-1400

AH

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Serial Number.....10/610,519
Filing Date.....June 30, 2003
Inventorship.....Darling et al.
Applicant.....Microsoft Corporation
Attorney's Docket No. MS1-1520US
MSDocket No.....304011.1
Title: Network Load Balancing with Host Status Information

PATENT ASSIGNMENT**PARTIES TO THE ASSIGNMENT****Assignor(s):**

Christopher L. Darling
2721 233rd Place NE
Sammamish, WA 98074

Joseph M. Joy
17219 NE 32nd Street
Redmond, WA 98052

Sunita Shrivastava
14924 NE 76th Ct.
Redmond, wa 98052

Chittur Subbaraman
5037 326th Place NE
Carnation, WA 98014

Assignee:

Microsoft Corporation
Corporation in the State of Washington
One Microsoft Way
Redmond, WA 98052

AGREEMENT

WHEREAS, ASSIGNOR(S) (listed above) are inventor(s) of an invention entitled "Network Load Balancing with Host Status Information," as described and claimed in the specification forming part of an application for United States letters patent referenced above;

10

172

WHEREAS, Microsoft Corporation (hereinafter referred to as ASSIGNEE), a corporation of the State of Washington having a place of business at One Microsoft Way, Redmond, WA 98052, is desirous of acquiring the entire right, title and interest in and to the invention and in and to any letters patent that may be granted therefor in the United States and in any and all foreign countries;

NOW, THEREFORE, in exchange for good and valuable consideration, the receipt of which is hereby acknowledged, ASSIGNOR(S) hereby sell, assign and transfer unto ASSIGNEE, the entire right, title and interest in and to said invention, said application and any and all letters patent which may be granted for said invention in the United States of America and its territorial possessions and in any and all foreign countries, and in any and all divisions, reissues and continuations thereof, including the right to file foreign applications directly in the name of ASSIGNEE and to claim priority rights deriving from said United States application to which said foreign applications are entitled by virtue of international convention, treaty or otherwise, said invention, application and all letters patent on said invention to be held and enjoyed by ASSIGNEE and its successors and assigns for their use and benefit and of their successors and assigns as fully and entirely as the same would have been held and enjoyed by ASSIGNOR(S) had this assignment, transfer and sale not been made. ASSIGNOR(S) hereby authorize and request the Commissioner of Patents and Trademarks to issue all letters patent on said invention to ASSIGNEE. ASSIGNOR(S) agree to execute all instruments and documents required for the making and prosecution of applications for United States and foreign letters patent on said invention, for litigation regarding said letters patent, or for the purpose of protecting title to said invention or letters patent therefor.

26 August 2003
Date

August 26th, 2003
Date

Christopher L. Darling
Christopher L. Darling

Joseph M. Joy
Joseph M. Joy

A

Jorney Docket No. MS1-1520US
MS Docket No. 304011.1
Serial No. 10/610,519

173

8/26/2003
Date

Sunita Shrivastava
Sunita Shrivastava

8/26/2003
Date

C. P. Subbaraman
Chittur Subbaraman

Y

SUPERINDUSTRIA Y COMERCIO
Radicacion : 84861217 00000000 Folios: 551
Fecha (IMP): 2004-06-28 16:59:25
Tramite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

MIT : 800176089-2

RECIBO OFICIAL DE CAJA : 04 - 50,354
FECHA : JUNIO 25 DE 2004

***** CONSIGNACION *****

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PGO	Vr. PAGO
CARDENAS Y CARDENAS P.I. CONSIGNACION		BANCO POPULAR	050-00110-6	28716291	25/06/2004	5,439,000.00

***** CONCEPTO *****

CANT.	RENTISTICO	CONCEPTO	TOTAL CONCEPTO
1	50005-01-01 SOLICITUDES	1 TRAMITES DE SOL. DE PATENTE DE IN	422,000.00
		TOTAL :	\$ 422,000.00

SON: CUATROCIENTOS VEINTIDOS MIL PESOS

RESPONSABLE : _____

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. _____
BALANCEO DE CARGA DE RED CON

CARDENAS Y CARDENAS
ABOGADOS

B

475

SUPERINDUSTRIA Y COMERCIO
Radicacion : 04061217 00000000 Folios: 551
Fecha (AFB): 2004-06-20 16:59:25
Tramite : 002 PATENTES D 1 REGISTRAD/ 411 PRESENTAC
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 04 - 50,361
FECHA : JUNIO 25 DE 2004

***** CONSIGNACION *****

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PAGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	050-00110-6	28716291	25/06/2004	5,439,000.00

***** CONCEPTO *****

CANT. CONTABILITICO	CONCEPTO	TOTAL CONCEPTO
1 50005-01-01 SOLICITUDES	02 INVOCACION DE UNA PRIORIDAD EN NU	118,000.00
	TOTAL :	\$ 118,000.00

SON: CIENTO DIEZ Y OCHO MIL PESOS

RESPONSABLE : 

CARDENAS & CARDENAS

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. _____

BAUNCE O DE CARGA DE RED CON
INFORMACION DE ESTADO DE ANFITRION

4

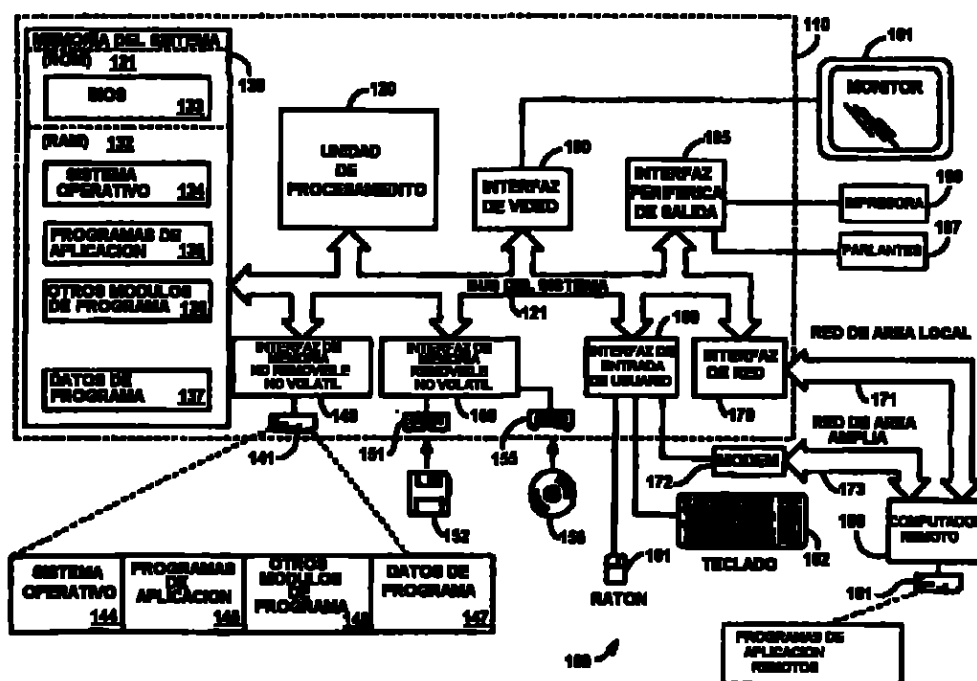


FIGURA 1

Orth para
publicar

BALANCEO DE CARGA DE RED CON INFORMACIÓN DE ESTADO DE
ANFITRIÓN

CAMPO TÉCNICO

Esta descripción se relaciona en general con balanceo de carga de red y en particular como ejemplo pero no limitación, con balanceo de carga de red con información de estado de anfitrión.

ANTECEDENTES

La comunicación, y muchas facetas de la vida que involucran la comunicación, han sido impactadas grandemente por la Internet. La Internet le posibilita a la información ser comunicada entre dos personas y/o entidades rápida y relativamente de manera fácil. La Internet incluye muchos nodos de red que están ligados juntos de tal forma, que la información puede ser transferida entre ellos. Algunos nodos de red pueden ser enrutadores que propagan un paquete desde un enlace a otro, pueden ser computadores de cliente individual, pueden ser redes personales para diferentes entidades

78

(por ejemplo, intraredes para negocios) y así sucesivamente.

Para este caso de red personal, así como también otros, los paquetes que llegan a un nodo o nodos de Internet son distribuidos a otros nodos dentro de la red personal. Tal red personal se puede formar, por ejemplo, de un conjunto de servidores que puede cada uno trabajar sobre paquetes que llegan a la red personal. Un negocio, una universidad una oficina de gobierno, etc., pueden recibir muchos paquetes en un corto periodo de tiempo en su red personal. Con el fin de responder de una manera puntual y para reducir la probabilidad de rechazo o pérdida de arribo de paquetes, la red personal puede contar con múltiples servidores que pueden trabajar cada uno en el arribo de paquetes simultáneamente.

Los paquetes que arriban son a menudo consultas que incumben a cierta información, tal como un documento, un ítem de catalogo, una página web, y así sucesivamente. Los paquetes de arribo también pueden pertenecer a una transacción económica entre un cliente y un comerciante. Otros propósitos para los paquetes de una comunicación basada en paquetes son posibles. Indiferente, los

paquetes que arriban son distribuidos entre diferentes servidores como a un conjunto de servidores para acomodar un arribo rápido de paquetes y/o intercambios de comunicación compleja.

La distribución de paquetes de arribo entre diferentes servidores de un conjunto de servidores es a menudo denominada balanceo de carga de red. En otras palabras, una operación de balanceo de carga se puede desarrollar sobre paquetes en la medida en que ellos arriban a un nodo o nodos de la Internet cuando el nodo o los nodos constituyen una red personal y/o cuando ellos conectan la red personal a la Internet.

Tal como una operación de balanceo de carga se logra utilizando hardware dedicado que enfrenta la red personal en el nodo o nodos que conectan la red personal a la Internet y/o que proporcionan una presencia para la red personal en la Internet. El hardware físico que desarrolla las operaciones de balanceo de carga es usualmente duplicado en su totalidad para realizar redundancia y mejorar la disponibilidad de la operación de balanceo de carga. Para incrementar la capacidad para las operaciones de balanceo de carga, hardware más

poderoso que replica la totalidad del equipo de hardware de balanceo de carga previa, y así la capacidad de operacional de éste, se sustituye por el hardware de balanceo de carga previa. Tal escalamiento de las capacidades operacionales de balanceo de carga es por lo tanto confinado para incrementar la potencia del hardware por vía de la sustitución de éste.

Para implementar una operación de balanceo de carga, el hardware usualmente desarrolla una distribución "round robin" de solicitudes de conexión de arribo. En otras palabras, las solicitudes de conexión de arribo son distribuidas a los servidores del conjunto de servidores de una manera repetitiva lineal con una solicitud de conexión simple que es distribuida a cada servidor. Esta distribución de conexiones de balanceo de carga "round robin" se utiliza típicamente sin importar la condición de la red personal o la naturaleza de una solicitud de conexión de arribo. Si una operación de balanceo de carga no se extiende más allá de una distribución "round robin", estos otros factores son sólo considerados en la medida en que ellos puedan ser inferidos del tráfico de red y/o de un nivel de congestión de la red personal.

De acuerdo con esto, subsiste la necesidad de esquemas y/o técnicas que mejoren el balanceo de carga de red y/o las opciones asociadas con ésta.

RESUMEN

En una primera implementación de medios de ejemplo, uno o más medios accesibles a procesador incluyen instrucciones ejecutables por procesador que, cuando se ejecutan, dirigen un sistema para desarrollar acciones que incluyen: acumulación de información de estado de anfitrión en anfitriones múltiples; y enviar la información de estado de anfitrión acumulada desde los anfitriones múltiples. En una segunda implementación de medios de ejemplo, uno o más medios accesibles al procesador incluyen instrucciones ejecutables por procesador que, cuando se ejecutan, dirigen un sistema para desarrollar acciones que incluyen: recibir información de estado de anfitrión de anfitriones múltiples; y tomar decisiones de balanceo de carga que responden a la información de estado de anfitrión recibida. En una tercera implementación de medios de ejemplo, uno o más medios accesibles al procesador incluyen instrucciones ejecutables por procesador que,

182

cuando se ejecutan, dirigen un sistema para desarrollar acciones que incluyen: determinar la salud en la información de carga en una base por aplicación; y seleccionar una aplicación de entre múltiples aplicaciones que responda a la salud y a la información de carga.

En una cuarta implementación de medios de ejemplo, uno o más medios accesibles al procesador incluyen instrucciones ejecutables por procesador que, cuando se ejecutan, dirigen un sistema para que desarrolle acciones que incluyen: analizar la salud y/o la información de carga para puntos finales de aplicación múltiple; y averiguando una parte de la muestra para los puntos finales de aplicación múltiple que responden al análisis.

En una quinta implementación de medios de ejemplo, uno o más medios accesibles por procesador incluyen instrucciones ejecutables por procesador que, cuando se ejecutan, le posibilitan al sistema implementar un protocolo de mensaje entre al menos un anfitrión y una o más unidades de balance de carga, el protocolo de mensaje utilizable para comunicar salud y/o información de carga

183

entre al menos un anfitrión y la una o más unidades de balanceo de carga.

En una implementación de sistema de ejemplo, un sistema incluye: al menos un dispositivo que está hospedando una o más aplicaciones, que al menos un dispositivo que incluye una tabla de salud y de carga que incluye múltiples entradas, cada entrada de las múltiples entradas asociadas con una aplicación de la una o más aplicaciones; cada entrada de las entradas múltiples incluyendo: un identificador de aplicación para una aplicación particular de la una o más aplicaciones; información que caracteriza al menos un estado de la aplicación particular; y al menos una directiva de balanceo de carga con relación a la aplicación particular.

Otro método, sistema, aproximación, aparato, interfaz de programación de aplicación (API), dispositivo, medio, procedimiento, disposición, etc., las implementaciones son descritas aquí.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Los mismos números son utilizados a lo largo de los dibujos para referenciar aspectos similares y/o correspondientes, características y componentes.

La FIG. 1 es un paradigma de balanceo de carga de red de ejemplo que ilustra una infraestructura d balanceo de carga y anfitriones múltiples.

La FIG. 2 es un paradigma de balanceo de carga de red de ejemplo que ilustra múltiples unidades de balanceo de carga y múltiples anfitriones.

La FIG. 3 ilustra una unidad de balanceo de carga de ejemplo que tiene funcionalidad separada y un anfitrión de ejemplo.

La FIG. 4 ilustra infraestructura de balanceo de carga de red de ejemplo que tiene funcionalidad de clasificación y de envío separado.

La FIG. 5 es un diagrama de flujo que ilustra un método de ejemplo para escalar infraestructura de balanceo de carga de red en diferentes configuraciones.

La FIG. 6 ilustra una primera configuración de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de dispositivo.

La FIG. 7 ilustra una segunda configuración de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de dispositivo.

Las FIGS. 8A y 8B ilustran primeras y segundas configuraciones de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de componente.

Las FIGS. 9A y 9B ilustran primeras y segundas configuraciones de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de recurso.

La FIG. 10 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de estado de anfitrión.

La FIG. 11 es un diagrama de flujo que ilustra un método de ejemplo para balanceo de carga de red que involucra información de estado de anfitrión.

La FIG. 12 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra salud e información de carga.

La FIG. 13A es una tabla de salud y carga de ejemplo como se ilustró en la FIG. 12.

La FIG. 13B es un caché de salud y de carga consolidado de ejemplo como se ilustró en la FIG. 12.

La FIG. 14 es un diagrama de flujo que ilustra un método de ejemplo para balanceo de carga de red que involucra salud e información de carga.

La FIG. 15 ilustra un protocolo de mensaje de ejemplo para comunicaciones entre los anfitriones y las unidades de balanceo de carga que son ilustradas en la FIG. 12.

La FIG. 16 ilustra un esquema de transmisión de mensaje de ejemplo para comunicaciones entre los anfitriones y

las unidades de balanceo de carga que son ilustradas en la FIG. 12.

Las FIG. 17A y 17B ilustran escenarios de almacenamiento Proxy de información de salud y carga para las tablas de salud y carga de la FIG. 13A y para los cachees de salud y carga consolidados de la FIG. 13B, respectivamente.

La FIG. 18 ilustra un procedimiento de parte anfitrión objetivo de ejemplo que utiliza información de salud y carga.

La FIG. 19 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de sesión.

La FIG. 20 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de sesión comunicante utilizando notificaciones y mensajes.

La FIG. 21 es un diagrama de flujo que ilustra un método de ejemplo para balanceo de carga de red que involucra información de sesión comunicante utilizando notificaciones y mensajes.

La FIG. 22 ilustra una aproximación de ejemplo para manejar información de sesión en unidades de balanceo de carga múltiple.

La FIG. 23A es una tabla de sesión de ejemplo como se ilustra en la FIG. 20.

La FIG. 23B es una tabla (DAMT) de manejador de átomo distribuido (DAM) de ejemplo como se ilustra en la FIG. 22.

La FIG. 24 es un diagrama de flujo que ilustra un método de ejemplo para manejar información de sesión en unidades de balanceo de carga múltiple.

La FIG. 25 ilustra una infraestructura de balanceo de carga de red de ejemplo que tiene solicitudes de funcionalidad de enrutamiento.

La FIG. 26 es un diagrama de flujo que ilustra un método de ejemplo para enrutar paquetes entrantes con relación a (i) información de sesión y (ii) salud e información de carga.

189

La FIG. 27 ilustra un flujo de enrutamiento de tráfico de ejemplo en ausencia de fallas.

La FIG. 28 ilustra un flujo de enrutamiento de tráfico de ejemplo en la presencia de falla (s).

La FIG. 29 ilustra procedimientos de migración tras error de ejemplo adicionales para alta estabilidad de la infraestructura de balanceo de carga de red.

La FIG. 30 ilustra una implementación operacional de ejemplo de interacción de enrutamiento de tráfico con información de salud y carga.

La FIG. 31 ilustra mecanismos de disponibilidad altos de ejemplo para infraestructura de balanceo de carga de red.

La FIG. 32 una aproximación de ejemplo de balanceo de carga de red a nivel de aplicación con migración de conexión.

La FIG. 33 es un diagrama de flujo que ilustra un método de ejemplo para migrar la conexión desde un dispositivo a un segundo dispositivo.

La FIG. 34 ilustra una aproximación de ejemplo de migración de conexión desde la perspectiva de un dispositivo de origen.

La FIG. 35 ilustra una aproximación de ejemplo a la migración de conexión desde la perspectiva de un dispositivo objetivo.

La FIG. 36 ilustra una aproximación de ejemplo de un procedimiento de descarga para una migración de conexión.

La FIG. 37 ilustra una aproximación de ejemplo a un procedimiento de carga para una migración de conexión.

La FIG. 38 ilustra una aproximación de ejemplo a canalización de paquete entre un enviador y un anfitrión.

La FIG. 39 es un diagrama de flujo que ilustra un método de ejemplo para canalización de paquete entre un primer dispositivo y un segundo dispositivo.

La FIG. 40 ilustra un ambiente operativo de cómputo de ejemplo (o dispositivo general) que es capaz de

implementar (total o parcialmente) al menos un aspecto del balanceo de carga de red como se describió aquí.

DESCRIPCIÓN DETALLADA

Paradigmas del Balanceo de Carga de Red de Ejemplo

Esta sección describe paradigma de ejemplo para balanceo de carga de red y se utiliza para suministrar fundaciones, ambientes, contextos, etc., para las descripciones en las siguientes secciones. Esta sección referencia primariamente a las FIGS. 1-3.

La FIG. 1 es un paradigma de balanceo de carga de red de ejemplo 100 que ilustra una infraestructura de balanceo de carga 106 y anfitriones múltiples 108. El paradigma de balanceo de carga de red de ejemplo 100 incluye clientes múltiples 102(1), 102(2)...102(m) y anfitriones múltiples 108(1), 108(2)...108(n), así como también la red 104 y la infraestructura de balance de carga 106.

Cada uno de los clientes 102 puede ser cualquier dispositivo que es capaz de comunicación de red, tal como un computador, una estación móvil, un dispositivo de

entretenimiento, otra red, y así sucesivamente. Los clientes 102 también pueden relacionarse con una persona y/o entidad que está operando un dispositivo de cliente. En otras palabras, los clientes 102 pueden comprender clientes lógicos que son usuarios y/o máquinas. La red 104 se puede formar de una o más redes, tal como la Internet, una intrared, una red telefónica alámbrica o inalámbrica, y así sucesivamente. Ejemplos adicionales de dispositivos para clientes 102 y tipos/topologías de red para redes 104 se describen adelante con referencia a la FIG. 40 en la sección titulada "Ambiente Operativo de Ejemplo para Computador u otro Dispositivo".

Los clientes individuales 102 son capaces de comunicarse con uno o más anfitriones 108, y viceversa, a través de la red 104 por vía de la infraestructura de balanceo de carga 106. Los anfitriones 108 hospedan una o más aplicaciones para interacción/comunicación con los clientes 102, para uso de los clientes 102, y así sucesivamente. Cada anfitrión 108 puede corresponder a un servidor y/o un dispositivo, múltiples servidores y/o múltiples dispositivos, parte de un servidor y/o parte de un dispositivo, alguna combinación de esto, y así sucesivamente. Las implementaciones particulares para

193

anfitriones 108 se describen adicionalmente adelante en el contexto de diferentes situaciones de balanceo de carga de red. (Sin embargo, el soporte de servicio para anfitriones 108 generalmente no se muestra por claridad). Además, ejemplos adicionales de dispositivos para anfitriones 108 también se describen adelante con referencia a la FIG. 40 de la sección titulada "Ambiente Operativo de Ejemplo para Computador u otro Dispositivo".

La infraestructura de balanceo de carga 106 es alcanzable o localizable a través de la red 104 en una o más direcciones de protocolo de Internet virtuales (IP). Las comunicaciones de cliente 102 (u otros nodos) que son dirigidas a la dirección IP virtual de la infraestructura de balanceo de carga 106 son recibidas allí y enviadas a un anfitrión 108. La infraestructura de balanceo de carga 106 está comprendida de un hardware y/o componentes de software (no explícitamente mostrados en la FIG. 1).

Aunque la infraestructura de balanceo de carga 106 es mostrada como una elipse integral, la infraestructura para efectuar balanceo de carga puede también ser distribuida a otros aspectos del paradigma de balanceo de carga de red de ejemplo 100. Por ejemplo, el o los

componentes de software de la infraestructura de balanceo de carga 106 se puede localizar en uno o más anfitriones 108 como se describe adicionalmente adelante. Ejemplos de arquitecturas para infraestructuras de balanceo de carga 106 se describen adelante con referencia a la FIG. 40 en la sección titulada "Ambiente Operativo de Ejemplo para Computador y otro Dispositivo".

Como se indicó en (1), uno o más anfitriones 108 pueden suministrar información de estado de anfitrión desde los anfitriones 108 a la infraestructura de balanceo de carga 106. Esta información de estado de anfitrión puede ser aplicación específica. Ejemplos de tal información de estado de anfitrión se describen más adelante e incluyen información de salud y/o carga, información de sesión, etc., para anfitriones 108. Una implementación particular que incluye proporcionar información de salud y/o carga de los anfitriones 108 a la infraestructura de balanceo de carga 106 se describe adelante en la sección titulada "Manejo de Salud y Carga de Ejemplo".

En (2), se envía una solicitud desde el cliente 102(1) a través de la red 104 a la infraestructura de balanceo de carga 106 en la dirección IP virtual de esta. El

contenido, formato, etc., de una solicitud de un cliente 102 puede depender de la aplicación a la cual está dirigida la solicitud, y el término "solicitud" puede incluir implícitamente una respuesta o respuestas del o los anfitriones 108 dependiendo del contexto. Las clases de solicitudes del cliente se incluyen, pero no están limitadas a:

1. Protocolo de transferencia de hipertexto (HTTP)
GET requiere de un cliente utilizar un programa de navegación. Dependiendo de la aplicación (y más específicamente, del localizador de recurso uniforme (URL) de las solicitudes), puede ser mejor servir las solicitudes por diferentes conjuntos de anfitriones, y la existencia de un estado de "sesión" de cliente en los anfitriones puede militar esas solicitudes de clientes específicos que sean enrutadas a anfitriones específicos. Las solicitudes pueden terminar una conexión de capa de sockets segura (SSL) (u otra encriptada).
2. Las conexiones de red privada virtuales (VPN) (por ejemplo, los anfitriones son un conjunto de

XFA

servidores VPN). En este caso, la "solicitud" puede ser considerada por ser un protocolo de canalización capa-2 (L2TP) o una "conexión" de protocolo de canalización punto a punto (PPTP) (la última es una combinación de una conexión de control de protocolo de control de transmisión (TCP) y el tráfico de datos de encapsulación de enrutamiento genérico asociado (GRE)).

3. Las conexiones de servidor de terminal (por ejemplo, los anfitriones son un conjunto de servidores de terminal).
4. Las solicitudes de propietario en la forma de conexiones TCP individuales (una por solicitud) que emplean un protocolo específico de aplicación de propietario.
5. Solicitudes de protocolo de acceso de objeto simple (SOAP).
6. Solicitudes de comunicación en tiempo real que involucran control de la información sobre una

KS

conexión TCP y medios sensibles a latencia que corren sobre protocolo en tiempo real (RTP).

Así, las solicitudes pueden tomar muchas diversas formas específicas de aplicación. En ciertas implementaciones descritas, la infraestructura de balanceo de carga 106 puede tomar decisiones de envío específica de aplicación.

En (3), la infraestructura de balanceo de carga 106 envía una solicitud de 102(1) al anfitrión 108(2) (en este ejemplo). La infraestructura de balanceo de carga 106 puede considerar uno o más de muchos factores cuando selecciona un anfitrión 108 al cual la solicitud le va a ser enviada, dependiendo de que implementación o implementaciones descritas aquí están siendo empleadas. Por ejemplo, la infraestructura de balanceo de carga 106 puede tener en cuenta: la información de salud y/o carga de la aplicación de cada anfitrión 108, la información de sesión que relaciona al cliente 102(1) como almacenado en un anfitrión 108, y así sucesivamente.

La FIG. 2 es un paradigma de balanceo de carga de red de ejemplo 200 que ilustra unidades de balanceo de carga múltiples 106 y anfitrión múltiple 108. Específicamente,

198

la infraestructura de balanceo de carga 106 es mostrada como unidades de balanceo de carga múltiple 106(1), 106(2)...106(u) en el paradigma de balanceo de carga de red de ejemplo 200. Adicionalmente, dos enrutadores y/o conmutadores 202(1) y 202(2) son ilustrados.

El enrutador/conmutadores 202, si están presentes, pueden ser considerados como parte de o separados de la infraestructura de balanceo de carga 106 (de la FIG. 1). El enrutador/conmutadores 202 son responsables de dirigir las solicitudes totales y los paquetes individuales que son recibidos de la red 104 a las direcciones IP virtuales compartidas (VIP) de unidades de balanceo de carga 106. Si un primer enrutador/conmutador 202 falla, el segundo enrutador/conmutador 202 toma control por el primero. Aunque se ilustran dos enrutadores/conmutadores 202, uno o más de dos enrutadores/conmutadores 202 pueden ser alternativamente empleados.

El enrutador/conmutadores 202 puede ser ignorante de la infraestructura de balanceo de carga o conocer el balanceo de carga. Si el enrutador/conmutador 202 no están enterados del balanceo de carga, una de las dos opciones de ejemplo se puede emplear: para una primera

opción, a una unidad de balanceo de carga 106 le es "asignada" la dirección de IP compartida, y todo el tráfico de red es enviado a ésta. Esta una unidad de balanceo de carga 106 entonces redistribuye homogéneamente el tráfico a través de otras unidades de balanceo de carga 106. Sin embargo, existen cuellos de botella y temas de migración tras error con esta primera opción (que puede ser mitigada si las direcciones VIP múltiples son compartidas y son repartidas entre unidades de balanceo de carga múltiple 106). Para una segunda opción, el enrutador/conmutadores 202 son "timados" en tráfico de red directo a todas unidades de balanceo de carga 106, las cuales individualmente deciden que tráfico debe ser aceptado para el balanceo de carga. Sin embargo, existe duplicación de esfuerzo ineficiente y temas de desempeño/compatibilidad de conmutador con esta segunda opción.

Si, de otra parte, el enrutador/conmutadores 202 están enterados del balanceo de carga, el enrutador/conmutadores 202 pueden ser hechos para distribuir el tráfico de red entrante entre unidades de balanceo de carga múltiple 106 (por ejemplo, en una forma "round-robin"). Se debe entender que tales

208

enrutadores/conmutadores 202 enterados del balanceo de carga son capaces de desarrollar funciones de balanceo de carga a un nivel rudimentario (por ejemplo, en hardware). Por ejemplo, los enrutadores/conmutadores 202 enterados del balanceo de carga pueden desarrollar afinidad de sesión basada en dirección IP simple de tal forma que todos los paquetes provenientes de una dirección IP de fuente específica son dirigidos a la misma unidad de balanceo de carga 106.

Cada unidad de balanceo de carga ilustrada separadamente 106 de unidades de balanceo de carga 106 pueden representar un dispositivo físico, dispositivos físicos múltiples, o parte de un dispositivo físico simple. Por ejemplo, la unidad de balanceo de carga 106(1) puede corresponder a un servidor, dos servidores, o más. Alternativamente, la unidad de balanceo de carga 106(1) y la unidad de balanceo de carga 106(2) pueden juntas corresponder a un servidor simple. Una unidad de balanceo de carga de ejemplo 106 se describe además adelante desde una perspectiva funcional con referencia a la FIG. 3.

Dos sendas de solicitud de ejemplo [1] y [2] son ilustrados en la FIG. 2. Para sendas de solicitud [1], el

cliente 102(2) transmite una solicitud sobre la red 104 que alcanza el enrutador/conmutador 202(1). El enrutador/conmutador 202(1) dirige el o los paquetes de la solicitud que se originó del cliente 102(2) a la unidad de balanceo de carga 106(1). La unidad de balanceo de carga 106(1) entonces envía el o los paquetes de la solicitud del anfitrión 108(1) de acuerdo con la misma funcionalidad de balanceo de carga (por ejemplo, política). Para la senda de solicitud [2], el cliente 102(m) transmite una solicitud sobre la red 104 que alcanza al enrutador/conmutador 202(2). El enrutador/conmutador 202(2) dirige el o los paquetes de la solicitud que se originó del cliente 102(m) a la unidad de balanceo de carga 106(u). La unidad de balanceo de carga 106(u) entonces envía el o los paquetes de la solicitud del anfitrión 108(n) de acuerdo con la misma funcionalidad de balanceo de carga. Funcionalidad de balanceo de carga de ejemplo se describe adicionalmente adelante con referencia a la FIG. 3.

La FIG. 3 ilustra una unidad de balanceo de carga de ejemplo 106 que tiene funcionalidad separada y un anfitrión de ejemplo 108. La unidad de balanceo de carga 106 incluye (7) bloques funcionales 302-314. Estos

bloques funcionales de unidad de balanceo de carga 106 se pueden comprender al menos parcialmente utilizando software. El anfitrión 108 incluye una o más aplicaciones 316. En una implementación descrita, la unidad de balanceo de carga 106 incluye un enviador 302, un clasificador 304, un enrutador de solicitud 306, un rastreador de sesión 308, un migrador de conexión 310, un canalizador 312 y un manejador de salud y carga 314.

El manejador de salud y carga 314 está localizado parcialmente en los anfitriones 108 y parcialmente sobre los dispositivos de las unidades de balanceo de carga 106. El manejador de salud y carga 314 monitorea la salud y/o carga (o más generalmente el estado) de los anfitriones 108 de tal forma que la información de salud y/o carga de ésta se puede utilizar para la funcionalidad de balanceo de carga (por ejemplo cuando se toman decisiones de balanceo de carga). Implementaciones de ejemplo para el manejador de salud y carga 314 son descritas adicionalmente adelante, particularmente en la sección titulada "Manejo de Salud y Carga de Ejemplo".

El rastreador de sesión 308 también puede estar localizado en los anfitriones 108 y parcialmente en los

ψ

dispositivos de las unidades de balanceo de carga 106. El rastreador de sesión 308 monitorea las sesiones que son establecidas por los clientes 102 de tal forma que las reconexiones/continuaciones de las sesiones previamente establecidas se pueden facilitar por la funcionalidad de balanceo de carga. Por ejemplo, algunas aplicaciones mantienen los datos de sesión de cliente específicos de la aplicación en los anfitriones (lo cual es también un tipo de información de estado de anfitrión). Estas aplicaciones típicamente esperan que los clientes utilicen el mismo anfitrión para la duración de cualquier sesión dada. Tipos de ejemplo de sesiones incluyen: (i) una conexión TCP (la cual es, estrictamente hablando, una sesión); (ii) una sesión SSL (iii) una sesión IP segura (IPsec); (iv) una sesión basada en "cookie" HTTP; y así sucesivamente.

Aunque el rastreador de sesión 308 se ilustra como un bloque discreto en una unidad de balanceo de carga 106, la funcionalidad de rastreo de sesión del rastreador de sesión 308 puede ser actualmente implementada a un nivel global. En otras palabras, la afinidad de sesión está soportada a través de las unidades de balanceo de carga múltiple 106. Los rastreadores de sesión 308 incluyen una

205

base de datos centralizada y/o una base de datos distribuida de información de sesión con el fin de preservar la afinidad de sesión. Las implementaciones de ejemplo para rastreadores de sesión 308, con un énfasis en una aproximación de base de datos distribuida, son descritas adicionalmente adelante, particularmente en la sección titulada "Rastreo de Sesión de Ejemplo".

El clasificador 304 utiliza los datos adquiridos y mantenidos por el manejador de salud y carga 314 y/o el rastreador de sesión 308, posiblemente en conjunto con otros factores, para clasificar solicitudes entrantes. En otras palabras, el clasificador 304 selecciona un anfitrión objetivo 108 para cada solicitud entrante de un cliente 102. El enviador 302 envía solicitudes de cliente (y/o los paquetes de estos) de acuerdo con el anfitrión objetivo 108 como seleccionado por el clasificador 304. El enviador 302 y el clasificador 304 pueden operar una base por paquete. Las implementaciones de ejemplo para el enviador 302 y el clasificador 304 son descritas adicionalmente adelante, particularmente en la sección titulada "Aproximación de Ejemplo a Balanceo de Carga de Red Flexible" y "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud".

206

El enrutador de solicitud 306, en contraste con las implementaciones por paquete del enviador 302 y el clasificador 304, puede actuar como un Proxy para un corrimiento de aplicación sobre un anfitrión 108. Por ejemplo, el enrutador de solicitud 306 puede terminar las conexiones TCP, analizar sintácticamente (quizás parcialmente) cada solicitud lógica de un cliente 102, y resometer cada solicitud lógica al anfitrión objetivo 108. Consecuentemente, cada solicitud lógica de un cliente 102 puede ser dirigida a un anfitrión diferente 108, dependiendo de las decisiones hechas por el enrutador de solicitud 306. Adicionalmente, el enrutador de solicitud 306 puede desarrollar pre-procesamiento sobre una conexión (por ejemplo, descripción SSL), puede escoger absorber ciertas solicitudes (por ejemplo, porque el enrutador de solicitud 306 mantiene un caché de respuestas), puede modificar arbitrariamente las solicitudes antes de enviarlas a los anfitriones 108, y así sucesivamente. Las implementaciones de ejemplo para el enrutador de solicitud 306 también son descritas adicionalmente adelante, particularmente en las secciones tituladas "Aproximación de Ejemplo a Balanceo de Carga de

Red Flexible" y "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud".

El migrador de conexión 310 posibilita que una conexión sea inicialmente terminada en la unidad de balanceo de carga 106 y luego migrada de tal forma que la conexión es subsecuentemente terminada en el anfitrión 108. Esta migración de conexión puede facilitar el balanceo de carga a nivel de aplicación. El migrador de conexión 310 es capaz de migrar una conexión desde una unidad de balanceo de carga 106 a un anfitrión 108 de tal manera que aquella de la terminación original en la unidad de balanceo de carga 106 es transparente a un cliente solicitante 102 y a las aplicaciones 316 del anfitrión nuevamente terminante 108. El canalizador 312 puede utilizar un esquema de encapsulación para canalizar los paquetes que no introducen una sobrecarga a cada paquete entonelado.

La funcionalidad del canalizador 312 también se puede utilizar en situaciones que no involucran una migración de conexión. Adicionalmente, el migrador de conexión 310 y/o el canalizador 312 pueden adicionalmente ser utilizados en implementaciones de balanceo sin carga. Las

implementaciones de ejemplo para migrador de conexión 310, así como también para el canalizador 312, son descritas adicionalmente adelante, particularmente en la sección titulada "Conexión de Ejemplo que Migra con Canalización Opcional y/o Balanceo de Carga con Nivel de Aplicación".

Cualquier implementación dada de una unidad de balanceo de carga 106 puede incluir una o más de las funciones ilustradas. Aunque ilustradas separadamente, cada una de las funciones de los bloques 302-314 pueden ser actualmente interrelacionadas con, traslapadas con, y/o inclusive de otras funciones. Por ejemplo, la salud y/o la información de la carga del manejador de salud y carga 314 se puede utilizar por el clasificador 304. También, el migrador de conexión 310 y el canalizador 312 trabajan en conjunto con el enviador 302 y el clasificador 304. Ciertos otros traslapos e interacciones de ejemplo son descritas aquí adelante.

En una implementación descrita, el anfitrión 108 corre y proporciona acceso a una o más aplicaciones 316. De manera general, las aplicaciones 316 incluyen programas de suministro de archivo, programas de manejo/servidor de

sitio web, programas de acceso remoto, programas de correo electrónico, programas de acceso de base de datos, y así sucesivamente. Específicamente, las aplicaciones 316 pueden incluir, pero no están limitadas a, servidores de red tales como Internet Information Server® (IIS) de Microsoft® Corporation, los servidores de terminal tales como Microsoft® Terminal Server™, y productos de servidor de seguridad y Proxy tal como Internet Security y Acceleration Server™ (ISA). Aunque los ejemplos de aplicación específica 316 en la sentencia precedente se relacionan con productos Microsoft®, el balanceo de carga de red como se describe aquí no está limitado a ningún vendedor(es), aplicación(es), o sistema(s) operativo(s).

Ejemplo de Aproximación a Balanceo de Carga de Red Flexible

Esta sección ilumina como las implementaciones de balanceo de carga de red descritas en esta y otras secciones suministran una aproximación flexible a un balanceo de carga de red. Esta sección primariamente referencia las FIGs. 4-9B.

208

Como se anotó anteriormente, la funcionalidad de balanceo de carga de red se puede escalar al reemplazar un primer balanceador de carga de red con un segundo, mayor y más potente balanceador de carga de red. Las capacidades de hardware del segundo balanceador de carga de red replican la totalidad de las capacidades de hardware del primer balanceador de carga de red, excepto que se suministra una mayor capacidad. Esta es una aproximación inflexible que puede ser muy ineficiente, especialmente cuando solamente una característica de balanceo de carga de red está limitando el desempeño y precipitando una actualización de un balanceador de carga de red.

La FIG. 4 ilustra una infraestructura de balanceo de carga de red de ejemplo que tiene funcionalidad de clasificación y envío separadas. La funcionalidad de clasificación separada y la funcionalidad de envío son representadas por el clasificador 304 y el enviador 302, respectivamente. Aunque clasificar y enviar las funciones se describe adicionalmente adelante, especialmente en la sección titulada "Enrutamiento de Clasificación, de Envío, y Solicitud de Ejemplo", una descripción inicial se presenta aquí como un ejemplo de interacción entre la

25%

funcionalidad de infraestructura de balanceo de carga de red y los anfitriones 108.

En una implementación descrita, el enviador 302 corresponde a, y es el punto de extremo de red para, la dirección IP virtual (VIP) (o direcciones). El enviador 302 es un componente de relativamente bajo nivel que toma decisiones de política simplificada y/o elemental, si existe alguna, cuando enrutan los paquetes a un destino adicional o final. El enviador 302 consulta una tabla de enrutamiento para determinar este destino. El clasificador 304 rellena las tablas de enrutamiento basada en uno o más factores (por ejemplo, información de estado de anfitrión), que son descritas adicionalmente en otras secciones aquí.

Los clientes 102 y los anfitriones 108 también corresponden a direcciones de red indicadas. Específicamente, el cliente 102(1) corresponde a direcciones C1, el cliente 102(2) corresponde a la dirección C2...el cliente 102(m) corresponde a la dirección Cm. También, el anfitrión 108(1) corresponde a la dirección H1, el anfitrión 108(2) corresponde a la

dirección H2...el anfitrión 108(n) corresponde a la dirección Hn.

Cinco sendas de comunicación (1)-(5) son mostradas en la FIG. 4. La senda de comunicación (1) está entre el cliente 102(1) y el enviador 302, y la senda de comunicación (5) está entre el enviador 302 y el anfitrión 108(1). Las sendas de comunicación (2)-(4) están entre el enviador 302 y el clasificador 304. Para simplicidad en este ejemplo, la conexión asociada con las sendas de comunicación (1)-(5) es una conexión TCP HTTP. Adicionalmente, el balanceo de carga en este ejemplo se relaciona con conexiones entrantes de enrutamiento al anfitrión menos cargado 108, al menos sin ninguna consideración explícita de balanceo de carga con nivel de aplicación.

Las sendas de comunicación (1)-(5) indican como el enviador 302 y el clasificador 304 balancean la carga de una conexión HTTP TCP del cliente 102(1). En (1), el cliente 102(1) inicia la conexión TCP al enviar un paquete TCP SYN direccionado a la dirección de IP. La infraestructura de enrutamiento de la red 104 enruta este paquete al enviador 302 por vía del enrutador/conmutador

202(1), el cual es el enrutador/conmutador 202 "más cercano" al emisor 302.

En (2), el emisor 302 consulta una tabla de enrutamiento, la cual puede ser interna al emisor 302 o accesible de otra forma desde éste, con el fin de mirar esta conexión. Esta conexión se puede identificar en la tabla de enrutamiento por las 4-tuplas TCP/IP (es decir, dirección IP fuente, puerto TCP fuente, dirección IP de destino, puerto TCP de destino). En razón a que este es el primer paquete de la conexión, no existe entrada en la tabla de enrutamiento. El emisor 302 por lo tanto aplica la acción de "ruta por defecto" la cual es para enviar este paquete al clasificador 304.

En (3), el clasificador 304 consulta su caché (por ejemplo, consolidado) de la información de estado del anfitrión para los anfitriones 108(1), 108(2)...108(n). El clasificador 304 concluye que el anfitrión 108(1) está disponible y el anfitrión menos cargado 108 en este instante para este ejemplo. El clasificador 304 también "aploma" una ruta en la tabla de enrutamiento consultada por el emisor 302 para esta conexión TCP. Por ejemplo, el clasificador 304 agrega una entrada de ruta o instruye

al enviador 302 para agregar una entrada de ruta a la tabla de enrutamiento que mapea la conexión TCP (por ejemplo, identificada por el 4-tuplas TCP) a un anfitrión con destino específico 108, que es el anfitrión 108(1) en este ejemplo. Más particularmente, la entrada de ruta especifica la dirección de red H1 del anfitrión 108(1).

En (4), el clasificador 304 envía el paquete TCP SYN de regreso al enviador 302. Alternativamente, el clasificador 304 puede enviar este paquete TCP SYN inicial al anfitrión 108(1) sin utilizar el enviador 302. Otras opciones disponibles para el clasificador 304 son descritas adicionalmente adelante.

En (5), el enviador 302 puede acceder a una entrada de ruta para la conexión representada por el paquete SYN, de tal forma que este envía el paquete al anfitrión 108(1) a la dirección H1. El enviador 302 también envía todos los paquetes subsecuentes del cliente 102(1) para esta conexión directamente al anfitrión 108(1). En otras palabras, el enviador 302 puede evitar la interacción adicional con el clasificador 304 para esta conexión. Una o una combinación de mecanismos, que son descritos

adicionalmente adelante, se pueden utilizar para suprimir la entrada de ruta cuando cesa la conexión.

Para la senda de comunicación (5) en muchos ambientes de protocolo, el enviador 302 no puede simplemente enviar los paquetes del cliente 102(1) como están al anfitrión 108(1) en la dirección de red H1 porque estos paquetes están dirigidos a la dirección VIP, que está hospedada por el enviador 302 mismo. En su lugar, el enviador 302 puede emplear una o más de las siguientes opciones de ejemplo:

1. El enviador 302 desarrolla una Traslación de Dirección de Red (NAT) al (i) sobrescribir la dirección IP fuente (cliente 102(1)) (C1) y el número de puerto con la dirección IP y el número de puerto generado NAT del enviador 302 y (ii) sobrescribir la dirección IP de destino (VIP) con la dirección IP (H1) del anfitrión (108(1)).
2. El enviador 302 desarrolla "Media-NAT" al sobre escribir la dirección IP de destino (VIP) con la dirección IP (H1) del anfitrión (108(1)) de tal

7

forma que la dirección IP fuente (cliente 102(1)) (C1) y el número de puerto son preservados.

3. El enviador 302 "canaliza" los paquetes recibidos del cliente 102(1) desde el enviador 302 al anfitrión 108(1). Específicamente en este ejemplo, la canalización se puede efectuar al encapsular cada paquete dentro de un nuevo paquete IP que está dirigido al anfitrión 108(1). El software enterado del balanceo de carga de red en el anfitrión 108(1) reconstruye el paquete original como se recibió en el enviador 302 del cliente 102(1). Este paquete original es luego indicado en la interfaz virtual en el anfitrión 108(1) (por ejemplo, la dirección VIP que corresponde al enviador 302 está unida a esta interfaz virtual en el anfitrión 108(1)). Las implementaciones de ejemplo de tal canalización son descritas adicionalmente adelante con referencia al canalizador 312, específicamente para escenarios de migración de conexión y particularmente en la sección titulada "Conexión de Ejemplo que Migra con Canalización Opcional y/o Balanceo de Carga con Nivel de Aplicación".

2/16

Aunque las FIGS. 4-9B muestran dos funciones específicas separadas, a saber clasificar y enviar, se debe entender que otras funciones, tales como aquellas del enrutador de solicitud 306, el servidor de sesión 308, el migrador de conexión 310, y el manejador de salud y carga 314, también pueden ser escalados independientemente (por ejemplo, factorizado independientemente), como se describe adicionalmente adelante. Adicionalmente, se debe notar que una o más de dos funciones se pueden separar y escalar independientemente en diferentes momentos y/o simultáneamente. También, aunque el TCP/IP se utiliza por motivos de claridad en muchos ejemplos en esta y otras secciones, los principios de balanceo de carga de red descritos aquí son aplicables a otros protocolos de transmisión y/o comunicaciones.

A manera de ejemplo en la FIG. 4, las funciones de balanceo de carga de red (tal como aquellas mostradas en la FIG. 3) se pueden separar una de la otra con propósitos de escalabilidad. Ellas también se pueden separar y duplicar en varias configuraciones para disponibilidad incrementada. Las configuraciones de ejemplo para escalabilidad y/o disponibilidad son

descritas adelante con referencia a la FIGS. 6-9B después de que se describe el método de la FIG. 5.

La FIG. 5 es un diagrama de flujo 500 que ilustra un método de ejemplo para escalar la infraestructura de balanceo de carga de red en diferentes configuraciones. El diagrama de flujo 500 incluye tres bloques 502-506. Aunque las acciones del diagrama de flujo 500 se pueden desarrollar en otros ambientes y con una variedad de esquemas de software, las FIGS. 1-4 y 6-9B son utilizadas en particular para ilustrar ciertos aspectos y ejemplos del método.

En el bloque 502, la infraestructura de balanceo de carga de red se opera en una primera configuración. Por ejemplo, cada configuración puede relacionar una o más de una selección, proporción y/o inter-relación de diferentes funcionalidades de balanceo de carga; un número de y/o tipo(s) de diferentes dispositivos; una organización y/o disposición de diferentes componentes; una distribución y/o localización de recursos; y así sucesivamente. En el bloque 504, la infraestructura de balanceo de carga de red es escalada. Por ejemplo, las funcionalidades de balanceo de carga separadas se pueden

216

expandir y/o concomitantemente contraer en un individuo y/o base independiente. En el bloque 506, la infraestructura de balanceo de carga de red escalada se opera en una segunda configuración.

Como se anota anteriormente, el balanceador de carga de red monolítico se puede escalar al incrementar la funcionalidad de balanceo de carga de red en su totalidad al suplantar el hardware de balanceo de carga de red previo con hardware de balanceo de carga de red más potente. En contra distinción, la infraestructura de balanceo de carga de red escalada puede posibilitarle balancear (sub-) funciones de carga de red a ser escaladas individualmente y/o independientemente. También se puede posibilitar funciones de balanceo de carga de red a ser escaladas juntas o individualmente entre diferentes números de dispositivos. Ejemplos de dispositivo, componente, y escalamiento orientado a recurso son suministrados adelante.

La FIG. 6 ilustra una primera configuración de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de dispositivo. En esta primera configuración de infraestructura de balanceo de carga de

red orientada a dispositivo, se ilustran tres dispositivos 602(1), 602(2), y 602(3). Sin embargo, uno, dos, o más de tres dispositivos 602 pueden emplearse alternativamente.

Como se ilustró, un emisor 302(1), un clasificador 304(1), y un anfitrión 108(1) son residentes y se ejecutan en el dispositivo 602(1). Un emisor 302(2), un clasificador 304(2), y un anfitrión 108(2) son residentes en y se ejecutan en el dispositivo 602(2). También, un emisor 302(3), un clasificador 304(3), un anfitrión 108(3) son residentes en y se ejecutan en el dispositivo 602(3). Así, en esta primera configuración de infraestructura de balanceo de carga de red orientada por dispositivo, un emisor 302, un clasificador 304, y un anfitrión 108 respectivos comparten los recursos de cada uno de los dispositivos respectivos 602.

En operación, los emisores 302 son los puntos finales de red para la o las direcciones VIP. Cualquier clasificador 304 puede aplomarse una ruta para una conexión a cualquier anfitrión 108, dependiendo de la información del estado del anfitrión. Por ejemplo, el clasificador 304(2) puede aplomarse una ruta para una nueva conexión

entrante al anfitrión 108(3). De acuerdo con una nueva entrada de ruta para esta conexión, el enviador 302(2) envía paquetes subsecuentes al anfitrión 108(3).

En una configuración de infraestructura de balanceo de carga de red orientada por dispositivo alternativa a la cual el primero ilustrado se puede escalar, un cuarto dispositivo 602(4) (no explícitamente mostrado en la FIG. 6) se puede agregar que incluye un enviador 302(4), un clasificador 304(4), y un anfitrión 108(4). Si, de otra parte, la funcionalidad de clasificación suficiente ya está presente con los clasificadores 304(1-3) pero la funcionalidad de envío puede beneficiar el manejo de solicitud de anfitriones 108, un cuarto dispositivo 602(4) se puede agregar el cual incluye un enviador 302(4) y opcionalmente un anfitrión 108(4). Para esta configuración escalada, otro clasificador 304(1,2, o 3) puede aplomar rutas para el enviador 302(4) a cualquiera de los anfitriones 108(1, 2, o 3) y el anfitrión 108(4), si está presente.

La primera configuración de infraestructura de balanceo de carga de red de ejemplo orientada por dispositivo de la FIG. 6 puede ser especialmente apropiada para

situaciones de hospedaje más pequeño en la cual los dispositivos separados para la infraestructura de balanceo de carga de red no son técnica y/o económicamente valiosos o viables. Sin embargo, como los deberes de hospedaje se expanden a un mayor número (y/o una mayor demanda en el mismo número) de anfitriones 108 o si la carga de red sobre los anfitriones 108 es significativa, la primera configuración de infraestructura de balanceo de carga de red de ejemplo orientada por dispositivo se puede escalar para acomodar esta expansión, representada por una segunda configuración de infraestructura de balanceo de carga de red de ejemplo orientada por dispositivo de la FIG. 7.

La FIG. 7 ilustra una segunda configuración de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva del dispositivo. En esta segunda configuración de infraestructura de balanceo de carga de red orientada por dispositivo, los tres dispositivos 602(1), 602(2), y 602(3) también se ilustran. De nuevo, uno, dos, o más de tres dispositivos 602 pueden ser alternativamente empleados.

fo

Como se ilustró, el emisor 302(1) y el clasificador 304(1) son residentes en y se ejecutan en el dispositivo 602(1). El emisor 302(2) y el clasificador 304(2) son residentes y se ejecutan en el dispositivo 602(2). También, el emisor 302(3) y el clasificador 304(3) son residentes en y se ejecutan en el dispositivo 602(3). Así, en esta segunda configuración de infraestructura de balanceo de carga de red orientada por dispositivo, cada emisor respectivo 302 y clasificador 304 no están compartiendo los recursos de cada uno de los dispositivos respectivos 602 con un anfitrión 108. Adicionalmente, la infraestructura de balanceo de carga de red puede estar sirviendo cualquier número de anfitriones 108.

En operación, los emisores 302 son de nuevo los puntos finales de red para la o las direcciones VIP. También, cualquier clasificador 304 puede aplomar una ruta para una conexión a cualquier anfitrión 108, dependiendo de la información del estado del anfitrión. Por ejemplo, el clasificador 304(3) puede aplomar una ruta para una nueva conexión entrante al anfitrión 108(2). De acuerdo con una nueva entrada de ruta para esta conexión, el emisor 302(3) envía paquetes subsecuentes al anfitrión 108(2).

De esta manera, la infraestructura de balanceo de carga de red como se efectúa en el software, por ejemplo, puede ser escalada al mover la infraestructura de balanceo de carga de red (o parte de ésta) de los dispositivos que son compartidos con los anfitriones 108 a los dispositivos que no son compartidos con los anfitriones 108. También, como se aludió anteriormente para la FIG. 6 otro dispositivo 602(4) se puede agregar a la infraestructura de balanceo de carga de red para proporcionar funcionalidad de envío adicional, funcionalidad de clasificación adicional, funcionalidad adicional de ambos tipos, y así sucesivamente.

Las FIGS. 8A y 8B ilustran las primeras y segundas configuraciones de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de componente. Como se ilustró, la primera configuración de infraestructura de balanceo de carga de red de ejemplo orientada por componente 800 incluye cuatro componentes. La segunda configuración de infraestructura de balanceo de carga de red de ejemplo orientada por componente 850 incluye seis componentes. Una segunda configuración alternativa 850 incluye un séptimo componente como se

7/22

indica por el bloque punteado, que se describe adicionalmente adelante.

Específicamente, la primera configuración de infraestructura de balanceo de carga de red de ejemplo orientada por componente 800 (o la primera configuración 800) incluye (i) dos envióadores 302(1) y 302(2) y (ii) dos clasificadores 304(1) y 304(2). La segunda configuración de infraestructura de balanceo de carga de red orientada por componente de ejemplo 850 (o la segunda configuración 850) incluye (i) cuatro envióadores 302(1), 302(2), 302(3), y 302(4) y (ii) dos clasificadores 304(1) y 304(2). Así, la primera configuración 800 es escalada a la segunda configuración 850 al agregar dos componentes, que son componentes de envío en este ejemplo.

En una implementación descrita, cada componente funcional relacionado con el balanceo de carga de red respectivo corresponde a un dispositivo respectivo (no explícitamente mostrado en las FIGS. 8A o 8B); sin embargo, cada componente puede corresponder alternativamente a parte de un dispositivo o a más de un dispositivo. Por ejemplo, los envióadores 302(1) y 302(2) se pueden distribuir a través de tres dispositivos. O el

enviador 302(1) y el clasificador 304(1) pueden corresponder a un primer dispositivo, y el enviador 302(2) y el clasificador 304(2) pueden corresponder a un segundo dispositivo.

Dos componentes funcionales relacionados con el balanceo de carga de red son agregados para escalar la primera configuración 800 a la segunda configuración 850. Sin embargo, un componente (o más de dos) pueden ser alternativamente escalados a la infraestructura de balanceo de carga de red. Adicionalmente, dos o más diferentes tipos de componentes funcionales se pueden escalar "simultáneamente". Por ejemplo, como se ilustró por el bloque en línea punteada, otro componente de clasificación (por ejemplo, el clasificador 304(3)) también se puede agregar cuando se escala la primera configuración 800 a la segunda configuración 850.

Más aún, el escalamiento por dos o más diferentes tipos de componentes funcionales se puede desarrollar en proporciones similares (por ejemplo, equivalentes) o no similares una a la otra. Como se ilustró, los componentes del enviador agregados 302(3) y 302(4) aunque no agregando ningún componente clasificador 304 o aunque

2/4

agregando un componente clasificador simple 304(3) representa un escalamiento a proporciones no similares. Sin embargo, dos componentes clasificadores 304(3) y 304(4) (el último de los cuales no está explícitamente ilustrado en la FIG. 8B) se puede agregar aunque los dos componentes envidadores 302(3) y 302(4) son agregados para escalar a proporciones similares. Sin importar esto, cada componente funcional relacionado con el balanceo de carga de red individual puede consumir una cantidad diferente de recursos de infraestructura de balanceo de carga de red disponibles, como se describe con referencia a las FIGS. 9A y 9B.

Las FIGS. 9A y 9B ilustran primeras y segundas configuraciones de infraestructura de balanceo de carga de red de ejemplo desde una perspectiva de recurso. La primera configuración de infraestructura de balanceo de carga de red de ejemplo orientada por recurso 900 (o la primera configuración 900) incluye una primera distribución o localización de recurso para una unidad de balanceo de carga 106. La segunda configuración de infraestructura de balanceo de carga de red de ejemplo orientada por recurso 950 (o la segunda configuración

950) incluye una segunda distribución de recurso para la unidad de balanceo de carga 106.

Como se ilustró, la primera configuración 900 incluye una distribución de recurso de 70%-30%, y la segunda configuración 950 incluye una distribución de recurso de 40%-60%. Tales recursos pueden incluir recursos de dispositivo totales (por ejemplo, número de dispositivos), recursos de procesamiento (por ejemplo, número de ciclos de procesador), recursos de memoria (por ejemplo, proporción de caché, memoria principal, etc.), ancho de banda de red y/o recursos de interfaz (por ejemplo, bits por segundo y/o tarjetas de interfaz de red física (NIC)), y así sucesivamente.

Específicamente para la primera configuración 900, el emisor 302 consume 70% de los recursos de la unidad de balanceo de carga 106 mientras el clasificador 304 consume el 30% de estos recursos. Después de la relocalización durante el procedimiento de escalamiento para producir la segunda configuración 950, el emisor 302 consume 40% de los recursos de la unidad de balanceo de carga 106 mientras el clasificador 304 consume el 60% de estos recursos.

726

En una situación de ejemplo, la primera configuración 900 podría facilitar mejor desempeño del balanceo de carga de red cuando más pocas pero más largas transacciones están siendo manejadas por los anfitriones asociados (no mostrados en las FIGS. 9A y 9B) porque la funcionalidad de clasificación se utiliza luego de la comunicación inicial para una conexión y para funcionalidad de envío se utiliza posteriormente. La segunda configuración 950, de otra parte, podría facilitar mejor desempeño de balanceo de carga de red cuando más, pero más cortas transacciones están siendo manejadas por el anfitrión asociado porque la funcionalidad de clasificación es utilizada para un mayor porcentaje de número total de paquetes puestos en embudo a través de la infraestructura de balanceo de carga de red. En esta situación, si la funcionalidad de enrutamiento de solicitud está también siendo empleada, entonces el o los enrutadores de solicitud 306 también son localizados como porcentaje de los recursos de cómputo totales. La distribución de recursos entre las tres funcionalidades se puede ajustar aunque las conexiones de manejo (por ejemplo, ajustadas "sobre el vuelo") dependen del consumo de recurso corriente y/o los déficit.

Como se indicó anteriormente con referencia a las FIGS. 2 y 3, cada unidad de balanceo de carga 106 puede corresponder a toda o a una parte de la infraestructura de balanceo de carga de red total 106. Para cualquier unidad de balanceo de carga físicamente, lógicamente, arbitrariamente, etc., definida o estipulada dada 106, los recursos de ésta pueden ser re-localizados durante el procedimiento de escalada. Más específicamente, una distribución de recurso entre diferentes funciones separadas relacionadas con el balanceo de carga de red diferentes de la unidad de balanceo de carga 106 se pueden alterar en un procedimiento de escalada. Adicionalmente, más de dos diferentes funciones, así como también otras funciones relacionadas con el balanceo de carga de red que no son específicamente ilustradas en la FIGS. 9A y 9B, pueden ser localizadas con diferentes porcentajes de recurso.

El porcentaje de los recursos totales del sistema localizados para todas las funciones de balanceo de carga también se pueden alterar en el procedimiento de escalada. Como un ejemplo de potencia de procesamiento general, el porcentaje de la potencia de procesamiento

total que está dedicada al balanceo de carga puede ser gradualmente incrementado en la medida en que la cantidad de tráfico que necesita ser balanceado en carga se incrementa.

El software de balanceo de carga de red opcionalmente desarrolla monitoreo para analizar y determinar si los recursos se deben re-localizar. Por ejemplo, el software de balanceo de carga de red puede monitorear la utilización del procesador de diferentes funciones relacionadas con el balanceo de carga de red. La presente re-localización también puede ser opcionalmente desarrollada automáticamente por el software de balanceo de carga de red en un modo fuera de línea o en línea.

Se debe entender que una capacidad de escalada de la infraestructura de balanceo de carga de red (por ejemplo, como la comprendida al menos parcialmente en el software) como se describió aquí puede relacionar diferentes instalaciones y no necesariamente un cambio a una instalación simple. En un ejemplo orientado a recurso, la infraestructura de balanceo de carga de red como se describe aquí se puede configurar de acuerdo con una distribución de recurso en un ambiente de instalación y

se puede configurar de acuerdo con otra distribución de recurso diferente en otro ambiente de instalación que tiene diferentes parámetros operacionales. Adicionalmente, las capacidades, características, opciones, etc., descritas anteriormente con relación al escalamiento también son aplicables para "desescalamiento". En otras palabras, los recursos dedicados a la infraestructura de balanceo de carga de red (o las subfunciones de ésta) también se pueden reducir.

Manejo de Salud y Carga de Ejemplo

Esta sección describe como la información de estado de anfitrión, tal como la información de salud y/o carga, se puede recolectar y utilizar en el balanceo de carga de red. Esta sección principalmente hace referencia a las FIGS. 10-18 e ilumina la funcionalidad de salud y carga tal como la suministrada por el manejador de salud y carga 314 (de la FIG. 3). Como se describió anteriormente con referencia a la FIG. 3, cada anfitrión 108 hospeda una o más aplicaciones 316. El manejador de salud y carga 314 utiliza información de salud y/o carga que se relaciona con las aplicaciones 316 y/o los anfitriones

108 para ciertas implementaciones descritas del balanceo de carga de red.

La FIG. 10 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra la información de estado de anfitrión (HSI) 1006. Cada anfitrión 108(1), 108(2)...108(n) incluye una o más aplicaciones 316(1), 316(2)...316(n) respectivamente. Estos anfitriones 108 generalmente y estas aplicaciones 316 específicamente pueden cambiar los estados de vez en cuando.

Por ejemplo, los anfitriones 108 y las aplicaciones 316 pueden estar aceptando nuevas conexiones o no aceptar nuevas conexiones. También, ellas pueden manejar rápidamente solicitudes de cliente o manejar lentamente solicitudes de cliente. Adicionalmente, ellas pueden tener muchos recursos en reversa o pocos recursos no utilizados. Todos o alguna parte de dichos datos, o de otros datos, pueden comprender información de estado de anfitrión 1006. Generalmente, la información de estado de anfitrión 1006 proporciona una indicación del estado de algún aspecto de los anfitriones 108 y/o las aplicaciones 316 que están corriendo sobre ésta.

23

En una implementación descrita, cada anfitrión 108(1), 108(2)...108(n) incluye un determinador de información de estado de anfitrión (HSI) 1002(1), 1002(2)... y 1002(n), respectivamente. Cada anfitrión 108(1), 108(2)...108(n) también incluye un diseminador de información de estado de anfitrión (HSI) 1004(1), 1004(2)... y 1004(n), respectivamente. Cada determinador de información de estado de anfitrión 1002 y/o diseminador de información de estado de anfitrión 1004 puede ser parte de infraestructura de balanceo de carga (LBI) 106.

Cada determinador de información de estado de anfitrión 1002 determina la información de estado de anfitrión 1006 para su respectivo anfitrión 108 y/o aplicaciones 316 que están corriendo sobre ésta. Las técnicas de ejemplo para Determinar tal información de estado de anfitrión 1006 son descritas adelante con referencia a las FIGS. 12-14, y particularmente la FIG. 13A. Cada diseminador de información de estado de anfitrión 1004 disemina la información de estado de anfitrión 1006 para su respectivo anfitrión 108 y/o las aplicaciones 316 para cargar infraestructura de balanceo 106 (por ejemplo, aquella porción o porciones de infraestructura de balanceo de carga 106 que no están localizadas en los

anfitriones 108). Las técnicas de ejemplo para diseminar tal información de estado de anfitrión 1006 son descritas adelante con referencia a las FIGS. 12-17 y particularmente las FIGS. 13B y 15-17.

Específicamente, cada diseminador de información de estado de anfitrión 1004 disemina información de estado de anfitrión 1006 (directa o indirectamente) a cada unidad de balanceo de carga (LBU) 106 de la infraestructura de balanceo de carga 106 que incluye al menos un manejador de salud y de carga 314 y/o clasificador 304. La infraestructura de balanceo de carga 106 se refiere a información de estado de anfitrión 1006 cuando se implementa el balanceo de carga de red. Por ejemplo, como se indicó por el lógico 1008, la infraestructura de balanceo de carga 106 es capaz de tomar decisiones de balanceo de carga en respuesta a información de estado de anfitrión 1006.

En operación en (1), los determinadores de información de estado de anfitrión 1002 determina la información de estado de anfitrión 1006 para los anfitriones respectivos 108 y/o las aplicaciones 316. En (1) y (2), los diseminadores de información de estado de anfitrión 1004

235

diseminan la información de estado de anfitrión 1006 desde los anfitriones 108 a la infraestructura de balanceo de carga 106. Por ejemplo, la información de estado de anfitrión 1006 se puede diseminar a unidades de balanceo de carga individuales 106. En (3), el lógico 1008 toma decisiones de balanceo de carga de red que responden a la información de estado de anfitrión 1006. En (4), las conexiones para enviar a los anfitriones objetivo 108 basadas en estas decisiones de balanceo de carga de red.

La FIG. 11 es un diagrama de flujo 1100 que ilustra un método de ejemplo para balancear carga de red que involucra información de estado de anfitrión. El diagrama de flujo 1100 incluye tres bloques 1102-1106. Aunque las acciones del diagrama de flujo 1100 se pueden desarrollar en otros ambientes y con una variedad de esquemas de software, las FIGS. 1-3 y 10 son utilizadas en particular para ilustrar ciertos aspectos y ejemplos el método.

En el bloque 1102, la información de estado de anfitrión es enviada desde los anfitriones a las unidades de balanceo de carga. Por ejemplo, la información del estado de anfitrión 1006 se puede enviar desde los anfitriones

108 a las unidades de balanceo de carga 106. En el bloque 1104, la información de estado de anfitrión es recibida de los anfitriones de las unidades de balanceo de carga. Por ejemplo, las unidades de balanceo de carga 106 pueden recibir información de estado de anfitrión 1006 de los anfitriones 108. En el bloque 1106, las decisiones de balanceo de carga son hechas en respuesta a la información de estado de anfitrión recibida. Por ejemplo, el lógico 1008 en las unidades de balanceo de carga 106 pueden tomar decisiones para el balanceo de carga de red en respuesta a la información de estado de anfitrión 1006.

Así, en la FIG. 10, la infraestructura de balanceo de carga 106 recolecta información de estado de anfitrión 1006 desde los anfitriones 108 (y/o las aplicaciones 316 de estos) y las solicitudes entrantes de balances de carga están dirigidas a los anfitriones 108 en respuesta a la información de estado de anfitrión 1006. Como se describe adicionalmente adelante con referencia a las FIGS. 12-18, esta información de estado de anfitrión 1006 puede ser de aplicación específica. Como también se describe adicionalmente adelante, ejemplos de la

7/35

información de estado de anfitrión 1006 incluyen información de salud y/o carga.

La FIG. 12 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de salud y/o carga (HLI) 1206. Los anfitriones 108(1), 108(2)...108(n) están acoplados a las unidades de balanceo de carga 106(1), 106(2)...106(u) por vía de un enlace de comunicación 1210 tal como una red.

Como se ilustró, los anfitriones 108 comunican la información de salud y carga 1206 a las unidades de balanceo de carga 106 utilizando un enlace de comunicación 1210. La comunicación bi-direccional de la información de salud y carga 1206, como se indicó por la flecha de doble punta, se refiere a comunicaciones de doble vía desde las unidades de balanceo de carga 106 a los anfitriones 108 que suministran cierto completamiento, coherencia, corrección, etc., de tal forma que los anfitriones 108 y/o las unidades de balanceo de carga 106 puedan fallar independientemente una de la otra. Tales comunicaciones de dos vías desde las unidades de balanceo de carga 106 a los anfitriones

238

108 son descritas adicionalmente adelante con referencia particular a la FIG. 15.

La información de salud refleja si un anfitrión dado y/o aplicación es capaz de manejar solicitudes de cliente. La información de carga refleja el número, cantidad, y/o nivel de solicitudes de cliente que el anfitrión y/o aplicación dada es capaz de manejar en un momento particular. En otras palabras, la carga puede reflejar directamente y/o inversamente un número disponible, cantidad, y/o nivel de capacidad total del anfitrión y/o la aplicación dada. Como se anotó anteriormente, las implementaciones descritas con referencia a las FIGS. 12-18 se enfocan en la información de salud y/o carga; sin embargo, aquellas implementaciones también son aplicables a la información de estado general para los anfitriones (incluyendo las aplicaciones de estos).

En una implementación descrita, cada anfitrión 108(1), 108(2)...108(n) incluye un componente de infraestructura de salud y carga (H&LI) 1202(1), 1202(2)...1202(n). Cada componente de infraestructura de salud y carga 1202 puede opcionalmente ser una porción de una infraestructura de balanceo de carga 106 que es residente en y ejecutante en

cada anfitrión 108. La información de salud y carga 1206 se puede efectuar en el software. Cuando funciona, cada infraestructura de salud y carga 1202(1), 1202(2)...1202(n) crea y mantiene una tabla de salud y carga respectiva (H&L) 1204(1), 1204(2)...1204(n).

Estas tablas de salud y carga 1204 pueden incluir entradas específicas de aplicación. La información de salud y carga 1206 que está almacenada en las tablas de salud y carga 1204 puede ser independiente de la infraestructura de balanceo de carga 106. Por ejemplo, los administradores, diseñadores, etc., pueden especificar criterios para información de salud y carga 1206 en el momento de la configuración. Adicionalmente, las entidades externas al dispositivo que son o que tienen un anfitrión 108 pueden contribuir a determinar la información de salud y carga 1206 para aplicaciones 316 en el dispositivo. Una tabla de salud y carga de ejemplo 1204 se describe adicionalmente adelante con referencia a la FIG. 13A.

Cada unidad de balanceo de carga 106(1), 106(2)...106(u) incluye una caché de salud y carga consolidada (H&L) 1208(1), 1208(2)...1208(u). Cada caché de salud y carga

consolidada 1208 incluye la información de cada tabla de salud y carga 1204(1), 1204(2)...1204(n). Consecuentemente, cada unidad de balanceo de carga 106 está provista con un acceso rápido a la información de salud y carga 1206 para cada anfitrión 108 para lo cual las unidades de balanceo de carga 106 son tráfico de red de balanceo de carga.

En operación, las infraestructuras de salud y carga 1202 empuja la información de salud y carga 1206 desde las tablas de salud y carga 1204 a cachés de salud y carga consolidados 1208. El mecanismo para suministrar información de salud y carga 1206 es impulsado por evento de tal forma que los cambios a las tablas de salud y carga 1204 son suministrados a cachés de salud y carga consolidados 1208 de una manera precisa y escalable.

La FIG. 13A es una tabla de salud y carga de ejemplo 1204 como se ilustró en la FIG. 12. En una implementación descrita, la tabla de salud y carga 1204 incluye entradas múltiples 1302 que están cada una asociadas con una aplicación diferente 316. Cada entrada 1302 puede corresponder a una hilera en la tabla de salud y carga 1204 que tiene tres columnas. Estas columnas corresponden

al identificador de aplicación (ID) 1302(A), la caracterización del estado de aplicación 1302(B), y la directiva del balanceador de carga 1302(C).

En razón a que cada entrada 1302 está asociada con una aplicación particular 316, se agrega una hilera en la medida en que cada aplicación es rotada (por ejemplo, por un administrador). De manera similar, una hilera es suprimida/removida cada vez que una aplicación es cerrada. Similarmente, los campos individuales en las columnas 1302(A), 1302(B), y/o 1302(C) son modificadas/actualizadas cuando cambia un valor de estas. Por ejemplo, cuando el valor de la caracterización del estado cambia para una aplicación dada 316, un valor en un campo de la caracterización del estado de la aplicación 1302(B) para la entrada 1302 de la aplicación dada 316 es actualizada.

Las adiciones y supresiones de entradas 1302 para las aplicaciones 316 se pueden efectuar con la entrada de un manejador de control en el anfitrión 108. Por ejemplo, una porción el manejador de control de un sistema operativo sabe cuando una aplicación 316 es arrancada y detenida porque ésta está activamente involucrada en el

arranque y parada de las aplicaciones 316. De esta manera, un manejador de control puede identificar que ésta ha, al menos en parte, arrancado una aplicación 316, y el manejador de control puede establecer que ésta ha, al menos en parte, parado la aplicación 316. La infraestructura de salud y carga 1202 puede por lo tanto ser informada del arranque y parada de las aplicaciones 316 por el manejador de control. De esta manera, ninguna comunicación explícita de las aplicaciones 316 tiene que ser suministrada a la infraestructura de salud y carga 1202. Un ejemplo de un manejador de control es el Service Control Manager (SCM) de Windows® Operating System de Microsoft® Corporation.

El identificador de aplicación 1302(A) incluye información que es utilizada para identificar únicamente la aplicación 316 a la cual está asociada la entrada 1302. El identificador de aplicación 1302(A) puede incluir uno o más de los siguientes para la aplicación asociada 316: la dirección y el puerto IP virtual, la dirección y el puerto IP físico, el protocolo utilizado, y cualquier información específica de protocolo. El protocolo puede ser HTTP, IPsec, SOAP, y así

sucesivamente. La información específica de protocolo puede ser un patrón o cuerda URL para delinear adicionalmente la aplicación asociada con la entrada 1302. Así, el identificador de aplicación 1302(A) más particularmente se refiere a un punto de extremo de aplicación específica sobre un anfitrión particular 108.

Otros identificadores de aplicación pueden ser empleados alternativamente. Por ejemplo, para reducir el ancho de banda de comunicación, el identificador de aplicación 1302(A) puede ser un número de 32 bits que mapea la información de ejemplo anterior en la infraestructura de salud y carga 1202 y en las unidades de balanceo de carga 106. Más aún, cualquiera de los campos en la entrada 1302 puede contener actualmente un identificador globalmente único (GUID) que es utilizado como una clave para mirar la información real para el campo.

La caracterización del estado de aplicación 1302(B) incluye información que refleja el estado de la aplicación 316 al cual está asociado la entrada 1302. La caracterización del estado de aplicación 1302(B) incluye lo siguiente para la aplicación asociada 316: salud de aplicación, carga de aplicación, y capacidad de

244

aplicación. La salud de aplicación es un valor cuasi-Booleano que indica si una aplicación está funcionando. La salud de la aplicación puede ser saludable, fallar, o desconocida. La salud de la aplicación es un valor relativamente instantáneo y es comunicado con latencia relativamente baja (por ejemplo, de aproximadamente un segundo o unos pocos segundos) para unidades de balanceo de carga 106 cuando el valor de salud de aplicación cambia.

La carga de aplicación es un valor que indica que tan ocupada está una aplicación dada y así, directa o inversamente, cuanta carga adicional puede manejar la aplicación dada. La carga de aplicación es un valor de cambio relativamente lento o promediado que puede ser suavizado con un mecanismo inductor de histéresis, si se desea, para eliminar picos transitorios de carga creciente o decreciente. Se comunica de manera relativamente infrecuente a las unidades de balanceo de carga 106 (por ejemplo, aproximadamente una a cuatro veces por minuto). El valor de la carga de aplicación le es dado un significado con relación a la capacidad de aplicación.

246

La capacidad de aplicación es un valor que indica la capacidad máxima de la aplicación. Esta se selecciona de manera genérica para ser significativa para un contexto dado pero aún suficientemente flexible para otros contextos. La capacidad de aplicación es un número ligado, no unitario (por ejemplo, 0-99) que es determinable en el momento de la configuración. Este puede estar basado en potencia de procesamiento, tamaño/velocidad de memoria, acceso de red, algunas combinaciones de estos, y así sucesivamente. La capacidad de aplicación expresa capacidades relativas entre otras aplicaciones del mismo tipo en un conjunto de anfitriones 108(1, 2...n).

Así, con relación a la capacidad de aplicación, la carga de aplicación gana significado. La carga de aplicación para una aplicación dada es un porcentaje de la capacidad de aplicación para la aplicación dada. Alternativamente, la carga de aplicación se puede expresar como un número no unitario del cual el porcentaje puede ser constatado en conjunto con el valor de la capacidad de aplicación.

La directiva del balanceador de carga 1302(C) incluye información que refleja el estado deseado y/o esperado de

247

la directiva establecida por la infraestructura de salud y carga 1202 para las unidades de balanceo de carga 106 con respecto a una aplicación 316 al cual está asociada la entrada 1302. La directiva del balanceador de carga 1302(C) incluye lo siguiente para la aplicación asociada 316: estado de balanceo de carga objetivo y estado de balanceo de carga corriente.

El estado de balanceo de carga objetivo refleja el estado de la directiva en las unidades de balanceo de carga 106 como se desea por la infraestructura de salud y carga 1202. El estado de balanceo de carga corriente refleja lo que la infraestructura de salud y carga 1202 entiende el estado corriente de la directiva en las unidades de balanceo de carga 106 para ser grabadas en las unidades de balanceo de carga 106. El estado de balanceo de carga corriente refleja así la directiva de balanceo de carga que la infraestructura de salud y carga 1202 espera de las unidades de balanceo de carga 106 estén habitualmente operativas como se dictó utilizando un protocolo de comunicación. Tal protocolo de comunicación de ejemplo se describe adicionalmente adelante con referencia a la FIG. 15. La interacción y relación entre el estado de balanceo de carga objetivo y el estado de balanceo de carga

247

corriente también es clarificado adicionalmente con la descripción de la FIG. 15.

El estado de balanceo de carga objetivo y el estado de balanceo de carga corriente puede cada uno tomar un valor de activo, inactivo, o drenado. Una directiva activa indica que nuevas solicitudes/conexiones son bienvenidas y pueden ser el objetivo en la aplicación que está asociada con la entrada 1302. La directiva inactiva indica que no deben ser enviados paquetes adicionales a la aplicación asociada. Una directiva drenada indica que no deben ser enviados nuevos paquetes para nuevas solicitudes/conexiones a la aplicación asociada pero que los paquetes para las solicitudes/conexiones existentes deben continuar siendo enviados a la aplicación asociada.

En una implementación descrita, la versión definitiva de la información de salud y carga respectiva 1206 se almacena en las tablas de salud y carga 1204 que están localizadas en cada anfitrión respectivo 108 de anfitriones múltiples 108. Con esta implementación, si un anfitrión 108 se estrella, la información de salud y carga 1206 que se pierde pertenece a aquellas aplicaciones 316 que también se estrellan. Una medida de

alta disponibilidad se gana por lo tanto automáticamente sin duplicación de datos. Sin embargo, la versión definitiva de la información de salud y carga 1206 puede ser alternativamente almacenada en cualquier otra parte. Otra de tales opciones de almacenamiento incluyen unidades de balanceo de carga 106 mismas, un anfitrión 108 que (como su sola tarea o junto con deberes de anfitrión) almacena y mantiene la información de salud y carga 1206 para otros múltiples anfitriones (incluyendo todos los otros) 108, otro dispositivo separado y/o externo, y así sucesivamente.

Si la versión definitiva de la información de salud y carga 1206 se almacena y se mantiene en cualquier otra parte además de ser distribuida a través de los anfitriones 108(1, 2...n), tal información de salud y carga 1206 se puede almacenar redundantemente (por ejemplo, también almacenada en un dispositivo duplicativo de almacenamiento, etc.) para propósitos de alta disponibilidad. Los escenarios proxy de ejemplo para almacenamiento de información de salud y carga 1206 son descritos adelante con referencia a las FIGS. 17A y 17B. La FIG. 17A está dirigida a un escenario proxy para las tablas de salud y carga 1204, y la FIG. 17B está dirigida

2/17

a un escenario proxy para los cachés de salud y carga consolidados 1208.

La FIG. 13B es un caché de salud y carga consolidado de ejemplo 1208 como se ilustra en la FIG. 12. En una implementación descrita, cada caché de salud y carga consolidado 1208 en cada unidad de balanceo de carga 106 incluyen al menos parte de la información almacenada en cada tabla de salud y carga 1204 para cada infraestructura de salud y carga 1202 en cada anfitrión 108. La información de salud y carga cacheada puede ser organizada de cualquier manera en el caché de salud y carga consolidado 1208.

Como se ilustró, el caché de salud y carga consolidado 1208 incluye un caché para cada anfitrión 108(1), 108(2)...108(n) que replica parte o toda la información en la tabla de salud y carga 1204 de cada anfitrión respectivo 108(1, 2...n). Específicamente, el caché de salud y carga consolidado 1208 incluye un caché para el anfitrión #1 1304(1), un caché para el anfitrión #2 1304(2)...un caché para el anfitrión #n 1304(n). Así, el caché de salud y carga consolidado ilustrado 1208 se

organiza a un nivel amplio por el anfitrión 108(1, 2...n), con cada caché individual 1304 incluyendo entradas específicas de aplicación para los correspondientes anfitriones respectivos 108(1, 2...n). Alternativamente, el caché de salud y carga consolidado 1208 se puede organizar a un amplio nivel por el tipo de aplicación 316, con bloques individuales que están dirigidos a un tipo de aplicación específica dividida adicionalmente por el anfitrión 108(1, 2...n). Otros formatos de estructuras de datos también se pueden emplear.

La FIG. 14 es un diagrama de flujo que ilustra un método de ejemplo para el balanceo de la carga de red que involucra información de salud y carga. El diagrama de flujo 1400 incluye ocho bloques 1402-1416. Aunque las acciones del diagrama de flujo 1400 se pueden desarrollar en otros ambientes y con una variedad de esquemas de software, las FIGS. 1-3 y 12-13B son utilizadas en particular para ilustrar ciertos aspectos y ejemplos del método. Por ejemplo, las acciones de los dos bloques 1402-1404 son desarrolladas por un anfitrión 108, y las acciones de los seis bloques 1406-1416 son desarrolladas por una unidad de balanceo de carga 106.

251

En el bloque 1402, se determina la información de salud y carga en el anfitrión. Por ejemplo, la información de salud y carga 1206 para la aplicación 316(2) se puede indagar por la infraestructura de salud y carga 1202(2) y almacenada en la tabla de salud y carga 1204(2) en el anfitrión 108(2). En el bloque 1404, la información de salud y carga determinada se disemina a las unidades de balanceo de carga. Por ejemplo, la infraestructura de salud y carga 1202(2) puede enviar información de salud y carga 1206 para las aplicaciones 316(2) a las unidades de balanceo de carga 106(1, 2...u). Como se indica por la flecha 1418, las acciones de los bloques 1402 y 1404 son repetidas de tal forma que (la aplicación) la salud y carga pueden ser continuamente monitoreadas y actualizadas en la medida en que ocurren los cambios.

En el bloque 1406, la información de salud y carga es recibida de los anfitriones. Por ejemplo, la unidad de balanceo de carga 106(1) puede recibir información de salud y carga 1206 de anfitriones múltiples 108(1, 2...n), que incluyen información de salud y carga 1206 para las aplicaciones 316(2) del anfitrión 108(2). En el bloque 1408, la información de salud y carga recibida es

cacheada. Por ejemplo, la unidad de balanceo de carga 106(1) puede almacenar información de salud y carga 1206 de los anfitriones 108(1, 2...n) en caché de salud y carga consolidado 1208(1). Con referencia a la FIG. 13B la implementación de un caché de salud y carga consolidado 1208(1), la información de salud y carga 1206 para las aplicaciones 316(2) del anfitrión 108(2) se pueden almacenar en caché para el anfitrión #2 1304(2). Como se indicó por la flecha 1420, las acciones de los bloques 1406 y 1408 son repetidos de tal forma que (la aplicación) la información de salud y carga puede ser continuamente recibida y actualizada en la medida en que ocurren los cambios.

Como se indicó por la flecha punteada 1422, las unidades de balanceo de carga 106 son también comunicaciones de manejo de los clientes 102 aunque manejando (aplicación) temas de salud y carga. En el bloque 1410, se recibe un paquete que solicita una nueva conexión. Por ejemplo, la unidad de balanceo de carga 106(1) puede recibir un paquete TCP SYN del cliente 102(2) a través de la red 104. En el bloque 1412, se consulta la información de salud y carga cacheada. Por ejemplo, la unidad de

251

balanceo de carga 106(1) puede consultar el caché de salud y carga consolidado 1208(1). Más particularmente, la unidad de balanceo de carga 106(1) puede consultar entradas que están asociadas con la aplicación a la cual está dirigido el paquete TCP SYN a través de los cachés para los anfitriones #1, #2...#n 1304(1, 2...n).

En el bloque 1414, un anfitrión es seleccionado que responde a la información de salud y carga cacheada. Por ejemplo, la unidad de balanceo de carga 106(1) puede seleccionar el anfitrión 108(2) que tiene aplicación o aplicaciones 316(2) que responden a la información de salud y carga 1206 que es cacheada en el caché de salud y carga consolidado 1208(1). La aplicación seleccionada 316 (y el anfitrión 108) deben ser saludables y ser capaces de aceptar carga adicional (por ejemplo, posiblemente la aplicación menos cargada entre aquellas aplicaciones que son del tipo de aplicación al cual el paquete TCP SYN está dirigido).

La consulta de la información de salud y carga cacheada (en el bloque 1412) y la respuesta seleccionada por el anfitrión a la información de salud y carga cacheada (en

el bloque 1414) se puede desarrollar antes de la recepción de un paquete específico que solicita nueva conexión y/o utilizar un esquema de tandas. También, la selección puede ser de acuerdo con cualquiera de muchos esquemas. Por ejemplo, se puede emplear un esquema basado en señal o basado en "round-robin". Con cada esquema, la selección puede involucrar un peso de cargas relativas entre las opciones de aplicación. Esta consulta y selección, junto con los esquemas basados en señal y "round-robin", son descritos adicionalmente adelante con referencia a la FIG. 18 y en la sección titulada "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud", especialmente con relación a la funcionalidad de clasificación.

Después de que el anfitrión objetivo es seleccionado del bloque 1414, el paquete con solicitud de nueva conexión se puede enviar a éste. En el bloque 1416, el paquete recibido del cliente es enviado al anfitrión seleccionado. Por ejemplo, el paquete TCP SYN es enviado a la unidad de balanceo de carga 106(1) para seleccionar el anfitrión 108(2). El envío de este paquete inicial se puede efectuar directamente por un clasificador 304 o por un enviador 302, como también se describe adicionalmente

adelante en la sección titulada "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud".

Para una implementación descrita, la infraestructura de salud de carga 1202 es residente en y distribuida a través de anfitriones múltiples 108 así como también estando localizada en unidades de balanceo de carga 106 (como se representó por el manejador de salud y carga 314). La infraestructura de salud y carga 1202 tiene tres responsabilidades. Primero, ésta expone el o los puntos de escucha para lograr estados de aplicación actualizados para las caracterizaciones de estados de aplicación 1302(B) de las tablas de salud y carga 1204. Segundo, ésta sintetiza la información de estado de aplicación para determinar que deben hacer las unidades de balanceo de carga 106, cual es la modalidad en la directiva del balanceador de carga 1302(C). Tercero, la infraestructura de salud y carga 1202 comunica la directiva de los anfitriones 108 a las unidades de balanceo de carga 106.

El contenido directivo de la directiva del balanceador de carga 1302(C) es efectivamente una versión digerida de la información para las caracterizaciones del estado de aplicación 1302(B). Sin embargo, las unidades de balanceo

de carga 106 también pueden recibir la información bruta de las caracterizaciones del estado de aplicación 1302(B) así como también esta directiva procesada. La comunicación del contenido de estos y otros campos de las tablas de salud y carga 1204 se logran utilizando un protocolo de mensaje que es descrito adelante con referencia a la FIG. 15.

La FIG. 15 ilustra un protocolo de mensaje de ejemplo 1500 para las comunicaciones relacionadas con salud y carga que son ilustradas en la FIG. 12 entre los anfitriones 108 y las unidades de balanceo de carga 106. Generalmente, un mecanismo de activación de evento se utiliza para empujar el cambio a las tablas de salud y carga 1204 de los anfitriones 108 a las unidades de balanceo de carga 106. En otras palabras, para una implementación descrita, la información es transmitida de los anfitriones 108 a las unidades de balanceo de carga 106 cuando las tablas de salud y carga 1204 están actualizadas. Esto evita enviar periódicamente una instantánea de cada una de las tablas de salud y carga 1204, que reducen el consumo de ancho de banda de red mediante la infraestructura de salud y carga 1202.

5/2

El protocolo de mensaje 1500 se puede implementar utilizando cualquier mecanismo de transporte de mensajes disponible. Tales mecanismos incluyen transmisión multidifusión confiable, transmisión punto a punto (por ejemplo, protocolo de datagrama de usuario (UDP)), y así sucesivamente. Como se ilustró, el protocolo de mensaje 1500 incluye siete tipos de mensaje 1502-1514: un mensaje de latido de corazón 1502, un mensaje de despedida 1504, un mensaje de cambio de hilera 1506, un mensaje instantáneo de consecución de tabla 1508, un mensaje instantáneo de envío de tabla 1510, un mensaje de estado de postulación de tabla 1512, y un mensaje equivocado de postulación 1514.

Se debe entender que, con excepción de las flechas 1516 y 1518, ninguna relación temporal entre los diferentes tipos de mensajes 1502-1514 está implicada por la ilustración. Por ejemplo, un mensaje de cambio de hilera 1506 no sigue típicamente un mensaje de adiós 1504.

El mensaje de latido de corazón 1502 indica que un anfitrión particular 108 está funcionando y proporciona alguna revisión de error para el contenido de una tabla de salud y carga correspondiente particular 1204 con

respecto al caché particular correspondiente para el anfitrión particular 1304 en caché de salud y carga consolidado 1208. Cada infraestructura de salud y carga 1202 en cada anfitrión 108 envía un mensaje de latido de corazón directamente o indirectamente a cada caché de salud y carga consolidado 1208 en cada unidad de balanceo de carga 106.

Los mensajes de latido de corazón 1502 de envejecimiento para los datos en caché de salud y carga consolidado 1208 que surge, en parte, porque una instantánea de la totalidad de cada tabla de salud y carga 1204 no es periódicamente transmitida a cada una de las unidades de balanceo de carga 106. Un esquema de transmisión para el mensaje de latido de corazón 1502 se describe adicionalmente adelante con referencia a la FIG. 16.

Los mensajes de latido de corazón 1502 incluyen un identificador para el anfitrión, datos de revisión de error, y opcionalmente un nombre DNS. El identificador del anfitrión puede ser un número único (por ejemplo, 32-bit) que es seleccionado en el momento de la configuración. Los datos de revisión de error pueden ser una suma de revisión, un número de secuencia con cambio

de estado, un número de generación, un valor CRC, etc., lo que le posibilita a la unidad de balanceo de carga receptora 106 validar aquellos contenidos de su caché de salud y carga consolidado 1208 que se comporta con los contenidos de la tabla de salud y carga 1204 del anfitrión que transmite 108. Si una aproximación del número de generación es empleada, entonces la generación múltiple de los ID se puede utilizar con cada ID de generación asignado a un "trozo" de aplicaciones. Los mensajes pueden entonces referirse a un número de trozo o a un par de ID de número/generación de trozo, dependiendo del contexto.

Los datos de revisión de error pueden ser un valor simple para la tabla de salud y carga 1204 total, o pueden ser valores múltiples determinados sobre una base por entrada 1302. El nombre DNS puede opcionalmente ser enviado (por ejemplo, cada "x" latido de corazón) para verificar o actualizar la dirección de la red correcta habitual para el anfitrión.

El mensaje de despedida 1504 es enviado desde un anfitrión particular 108 a unidades de balanceo de carga 106 para indicar que el anfitrión particular 108 está

254

planeando apagar. Los mensajes de despedida 1504 incluyen un identificador de anfitrión que puede ser indexado/mapeado a una dirección de red para el anfitrión particular 108. El mensaje de despedida 1504 se utiliza para limpiar, las apagadas intencionales de los anfitriones 108 para precipitar una "limpieza rápida". Sin embargo, si el mensaje de despedida se pierde, los cachés eventualmente envejecerán las entradas de anfitrión particular 108 porque los mensajes de latido de corazón 1502 ya no serán enviados.

El mensaje de cambio de hilera 1506 es enviado desde un anfitrión particular 108 a unidades de balanceo de carga 106 para indicar que la salud y/o carga para una aplicación dada 316 de un anfitrión particular 108 ha cambiado. El mensaje de cambio de hilera 1506 incluye un identificador de anfitrión, un identificador de aplicación, una operación, y datos para la operación. Los identificadores de anfitrión de ejemplo son descritos anteriormente con relación a los mensajes de latido de corazón 1502 y los mensajes de despedida 1504. Los identificadores de aplicación de ejemplo son descritos anteriormente con relación al identificador de aplicación

1302(A) de una entrada asociada con aplicación 1302 de las tablas de salud y carga 1204.

La operación de cambio de hilera se puede agregar, suprimir, o actualizar. En otras palabras, los datos para la operación se pueden agregar a (para una operación de suma) o un reemplazo para (para una operación de actualización) información ya presente en los cachés de salud y carga consolidados 1208 en las unidades de balanceo de carga 106. Para una operación de supresión, no se requiere suministrar datos. El protocolo de mensajes 1500 se define de tal forma que se puedan estipular múltiples operaciones a ser desarrolladas por un mensaje de cambio de hilera simple 1506. De esta manera para un identificador de anfitrión particular, los conjuntos de un identificador de aplicación, operación, y datos de operación se pueden repetir para múltiples aplicaciones 316 del anfitrión 108 que es identificado por el identificador de anfitrión particular.

El mensaje instantáneo de consecución de tabla 1508 es enviado desde una unidad de balanceo de carga particular 106 para un caché de carga y salud consolidado particular 1208 a un anfitrión individual 108 o a anfitriones 108.

Este mensaje instantáneo de consecución de tabla 1508 solicita que la infraestructura de salud y carga 1202 en el anfitrión 108 suministre una instantánea de la tabla de salud y carga respectiva 1204 para el respectivo anfitrión 108. Este mensaje incluye una identificación de la unidad de balanceo de carga solicitante 106 y se puede utilizar por una unidad de balanceo de carga 106 (i) después de que ésta ha fallado y se ha recuperado; (ii) después de que un anfitrión 108 falla, se recupera e inicia el envío de mensajes de latido de corazón 1502 de nuevo; (iii) si un mensaje de cambio de hilera 1506 es enviado a la unidad de balanceo de carga 106, pero el mensaje se cae, de tal forma que su caché de salud y carga consolidado 1208 está por fuera del sync con la respectiva tabla de salud y carga 1204 para el respectivo anfitrión 108; y (iv) así sucesivamente.

Para la tercera (iii) situación, la falta de sincronización entre el caché de salud y carga consolidado 1208 y la respectiva tabla de salud y carga 1204 para el respectivo anfitrión 108 se descubre mediante un mensaje de latido de corazón subsecuente 1502 del respectivo anfitrión 108 porque la "revisión de error" indicará que el caché de salud y carga consolidado

1208 está fuera de fecha. La unidad de balanceo de carga 106 puede entonces enviar un mensaje instantáneo de consecución de tabla 1508 de tal forma que esta pueda actualizar su caché de salud y carga consolidado 1208. Así, para cualquiera de las tres situaciones de ejemplo (i, ii, iii), la unidad de balanceo de carga 106 subsecuentemente reconstituye su caché de salud y carga consolidado 1208 utilizando una instantánea de consecución de tabla 1508. La instantánea de consecución de tabla 1508 se puede enviar repetidamente a cada anfitrión 108 de una manera punto a punto o se puede enviar una vez a muchos anfitriones 108 de una manera multidifusión.

El mensaje instantáneo de envío de tabla 1510 es enviado desde un anfitrión individual 108 a una unidad de balanceo de carga 106 después de que el anfitrión individual 108 ha recibido el mensaje instantáneo de consecución de tabla 1508 desde una unidad de balanceo de carga particular 106 como se indica por la flecha 1516. Los contenidos de un mensaje instantáneo de tabla de envío 1510 se preparan por la infraestructura de salud y carga 1202 e incluyen todo o al menos múltiples hileras de la tabla de salud y carga 1204 del anfitrión

individual 108 de tal forma que la unidad de balanceo de carga particular 106 puede reconstruir su caché de salud y carga consolidado 1208. El mensaje instantáneo de tabla de envío 1510 puede ser un mensaje separadamente diseñado o puede ser equivalente a una secuencia de operaciones de suma en un mensaje de cambio de hilera 1506.

Los mensajes de estado de tabla postulado 1512 y los mensajes de equivocación postulados 1514 están relacionados con el estado de balanceo de carga objetivo y el estado de balanceo de carga habitual de la directiva balanceadora de carga 1302(C) de una entrada 1302 en una tabla de salud y carga 1204. El estado de balanceo de carga objetivo es la directiva que la infraestructura de salud y carga 1202 con que se desea que las unidades de balanceo de carga 106 sean operadas. El estado de balanceo de carga habitual es la directiva bajo la cual la infraestructura de salud y carga 1202 espera o cree que las unidades de balanceo de carga 106 estarán habitualmente operando. De manera general, los dos estados de balanceo de carga son idénticos.

Sin embargo, el estado de balanceo de carga objetivo puede diferir del estado de balanceo de carga habitual

durante el periodo transicional para un cambio de directiva de estado. Por ejemplo, el estado de balanceo de carga objetivo y el estado de balanceo de carga habitual son ambos inicialmente ajustados para la activación. Un problema con el anfitrión 108 y/o una aplicación 316 de ésta es detectada y la directiva de estado de balanceo de carga objetivo es conmutada a drenaje. Esta directiva de drenaje es comunicada a las unidades de balanceo de carga 106 utilizando un mensaje de cambio de hilera 1506.

Existe un retraso antes de que este cambio de directiva se note en todos los cachés de salud y carga consolidados 1208 de todas las unidades de balanceo de carga 106. Durante este periodo de transición, el estado de balanceo de carga objetivo es drenado mientras que el estado de balanceo de carga habitual está aún activo en la tabla de salud y carga 1204 del anfitrión 108. Antes de cambiar el estado de balanceo de carga habitual al drenado, la infraestructura de salud y carga 1202 quiere asegurarse que los cachés de salud y carga consolidados 1208 hayan sido de hecho actualizados a la nueva directiva del estado de drenaje.

Para verificar que los cachés de salud y carga consolidados 1208 de las unidades de balanceo de carga 106 se han actualizado a la nueva directiva de estado, la infraestructura de salud y carga 1202 envía un mensaje de estado de tabla postulada 1512 a las unidades de balanceo de carga 106. El mensaje de estado de tabla postulada 1512 se envía algún tiempo después (por ejemplo, un periodo de retraso predeterminado) de la transmisión de un mensaje de cambio de hilera 1506 indica que la directiva de estado debe ser cambiada. El mensaje de estado de la tabla postulada 1512, en este ejemplo, indica que el estado de tabla se debe drenar. Como se indicó por la flecha punteada 1518, una unidad de balanceo de carga 106 responde a este mensaje de estado de tabla postulada 1512 si su caché de salud y carga consolidado 1208 difiere de la directiva de estado postulada.

Si la directiva en el caché de salud y carga consolidado 1208 no difiere de la directiva de estado postulada, entonces esa unidad de balanceo de carga 106 envía un mensaje equivocado postulado 1514 a la infraestructura de salud y carga 1202 del anfitrión 108 que emite el mensaje de estado de tabla postulada 1512. Esta infraestructura

de salud y carga 1202 entonces reenvía periódicamente mensajes de estado de tabla postulada 1512 hasta que no son recibidos mensajes equivocados postulados adicionales 1514 desde los cachés de salud y carga consolidados 1208. En ese punto, la infraestructura de salud y carga 1202 envía un mensaje de cambio de hilera 1506 con un nuevo estado de balanceo de carga habitual. En este sentido, los cachés de salud y carga consolidados 1208 son los determinadores definitivos del estado de balanceo de carga habitual, y la infraestructura de salud y carga 1202 es el determinador definitivo del estado de balanceo de carga objetivo.

La FIG. 16 ilustra un esquema de transmisión de mensaje de ejemplo para las comunicaciones que son ilustradas en la FIG. 12 entre los anfitriones 108 y las unidades de balanceo de carga 106. El esquema de transmisión de mensaje de ejemplo puede reducir el ancho de banda consumido por los mensajes de latido de corazón 1502 sobre el enlace de comunicación 1210. El esquema de transmisión de mensaje de la FIG. 16 está particularmente adaptado a los mensajes de latido de corazón 1502, pero se puede utilizar para otros mensajes de protocolo de mensaje 1500.

Un grupo de anfitriones 108(1), 108(2), 108(3)...108(11), y 108(12) son ilustrados junto con las unidades de balanceo de carga 106(1), 106(2)...106(u). Cada línea representa enlaces de membresía o inclusión entre los grupos de anfitriones 108(1, 2...12). El grupo de anfitriones 108(1, 2...12) forma una membresía de nodos que trabajan juntos para propagar la información de latido de corazón a las unidades de balanceo de carga 106. Aunque los doce anfitriones son mostrados, más o menos pueden ser parte de cualquier grupo dado de anfitriones. También, el conjunto total de anfitriones 108 que están siendo servidos por la infraestructura de balanceo de carga 106 pueden ser divididos en uno, dos, tres, o más grupos de anfitriones.

En una implementación descrita, la membresía de nodos para grupos de anfitriones 108(1, 2...12) eligen un líder que es responsable de transmitir los mensajes de latido de corazón 1502 a las unidades de balanceo de carga 106. Cada anfitrión (no líder) 108 en el grupo de anfitriones 108(1, 2...12) envía sus mensajes de latido de corazón 1502 al líder elegido. El anfitrión 108(4) es el líder elegido en este ejemplo.

Con la membresía de nodos, la información de latido de corazón para cada anfitrión 108 en el grupo de anfitriones 108(1, 2...12) se propaga al anfitrión líder del grupo 108(4). El anfitrión 108(4) recolecta la información de latido de corazón y la consolida en un mensaje de latido de corazón consolidado 1602. Los mensajes de latido de corazón consolidados 1602(1), 1602(2)...1602(u) son entonces enviados a las respectivas unidades de balanceo de carga 106(1), 106(2)...106(u). Estos mensajes de latido de corazón consolidados 1602 pueden opcionalmente ser comprimidos para reducir adicionalmente el consumo de ancho de banda.

Como otra alternativa, el anfitrión líder 108(4) puede solamente enviar cambios en membresía de grupos a los cachés de salud y carga consolidados 1208. En otras palabras, en este modo, los cachés de salud y carga consolidados deben 1208 tratan primeramente si no solamente con cambios de estado a la membresía. Es responsabilidad del anfitrión líder 108(4) asegurarse que el primer hola es enviado cuando el anfitrión 108 entra en línea, y que un mensaje de adiós 1504 se consigue enviar cuando el anfitrión 108 sale de línea.

Adicionalmente, un anfitrión 108 puede especificar periódicamente que el mensaje de latido de corazón 1502 va a ser "enviado". Esto le indica al anfitrión líder 108(4) enviarlo a los cachés de salud y carga consolidados 1208 aunque éste no represente un cambio de membresía.

Los mensajes de latido de corazón 1502 (incluyendo los mensajes de latido de corazón consolidados 1602) son utilizados por unidades de balanceo de carga 106 cuando sus cachés de salud y carga consolidados 1208 no están sincronizados con las tablas de salud y carga 1204. Esta falta de sincronización puede surgir, por ejemplo, de una estrellada u otra falla del caché de salud y carga consolidado 1208 y/o de la unidad de balanceo de carga 106. Como se describió anteriormente, cada mensaje de latido de corazón 1502 incluye datos de revisión de error que son utilizables para verificar la equivalencia entre un caché de salud y carga consolidado 1208 y las tablas de salud y carga 1204. Si no se descubre equivalencia con relación a un anfitrión particular 108 y/o una aplicación 316 de éste, el nombre DNS del anfitrión particular 108 se adquiere de los mensajes de latido de corazón 1502.

~~6769~~

El nombre DNS es utilizado por el caché de salud y carga consolidado 1208 para enviar un mensaje instantáneo de tabla de consecución 1508 al anfitrión particular 108 con el fin de actualizar la información de salud y carga 1206 en la forma de un mensaje instantáneo de tabla de envío 1510. Un mensaje diferente o igual instantáneo de tabla de consecución 1508 es enviado a cada anfitrión 108 para lo cual no se ha descubierto equivalencia. Eventualmente, la información de salud y carga 1206 en el caché de salud y carga consolidado 1208 es equivalente a la información de salud y carga 1206 en las tablas de salud y carga 1204 como verificables por los nuevos mensajes de latido de corazón 1502. De esta manera, el caché de salud y carga consolidado con falla 1208 puede ser retro-iniciado en operación sin omisión manual utilizando protocolo de mensaje 1500 y un esquema de revisión de equivalencia.

La FIG. 17A y la FIG. 17B ilustran escenarios de almacenamiento proxy de información de salud y carga de ejemplo para las tablas de salud y carga 1204 y para los cachés de salud y carga consolidados 1208, respectivamente. En las implementaciones descritas anteriormente con referencia a las FIGS. 12-16, los anfitriones 108 incluyen infraestructura de salud y carga

1202. Sin embargo, otras implementaciones pueden implicar anfitriones que no incluyen infraestructura de salud y carga 1202.

Por ejemplo, un anfitrión puede estar corriendo una versión de aplicación o aplicaciones y/o un sistema operativo para el cual la infraestructura de salud y carga no está implementada o por razones de política pueden no se instalados en el anfitrión. Consecuentemente, estos tipos de anfitriones no tienen infraestructura de salud y carga 1202 que se ejecute sobre ésta. El anfitrión 1702 es un anfitrión tal que no ejecute la infraestructura de salud y carga 1202. Sin embargo, el anfitrión 1702 puede utilizar infraestructura de salud y carga 1202 que esté ejecutando sobre uno o más proxys, tal como el proxy 1704.

El proxy 1704 tiene residente allí y la ejecución sobre este una infraestructura de salud y carga 1202, que incluye una tabla de salud y carga 1204. El anfitrión 1702 puede utilizar la funcionalidad de la infraestructura de salud y carga 1202 al suministrar información de salud y carga 1206 a la tabla de salud y carga 1204 para aplicaciones que están corriendo sobre el

anfitrión 1702. Alternativamente, el proxy 1704 puede deducir la salud y carga sobre el anfitrión 1702 al desarrollar acciones de monitoreo externas. El proxy 1704 se ilustra como el proxy 1704(1) y 1704(2) para redundancia y alta disponibilidad resultante.

En implementaciones descritas anteriormente con referencia a las FIGS. 12-16 y adelante con referencia a la FIG. 18, el balanceo de carga es efectuado con unidades de balanceo de carga 106 que incluyen cachés de salud y carga consolidados 1208. Sin embargo, otras implementaciones pueden implicar balanceo de carga que no incluye cachés de salud y carga consolidados 1208.

Por ejemplo, el balanceo de carga se puede efectuar mediante hardware de balanceo de carga monolítico u otra infraestructura de balanceo de carga que no almacene o no pueda almacenar o de otra manera incluir un caché de salud y carga consolidado 1208. El balanceador de carga 1706 refleja tal dispositivo o dispositivos de balanceo de carga que no tengan un caché de salud y carga consolidado 1208. Sin embargo, el balanceador de carga 1706 puede utilizar un caché de salud y carga consolidado

1208 que exista en uno o más proxys, tal como el proxy 1708.

El proxy 1708 incluye un caché de salud y carga consolidado 1208, que almacena la información de salud y carga 1206 para las aplicaciones hospedadas que están siendo servidas por el balanceador de carga 1706. El balanceador de carga 1706 puede utilizar la información de salud y carga 1206 del caché de salud y carga consolidado 1208 cuando desarrolla funciones de balanceo de carga al acceder tal información utilizando interfaces de programación de aplicación (API) nativa a y soportadas por el balanceador de carga 1706. Alternativamente, el caché de salud y carga consolidado 1208 puede invocar los API para que empujen la información de salud y carga 1206, incluyendo las directivas, al balanceador de carga 1706. El proxy 1708 se ilustra como el proxy 1708(1) y 1708(2) para redundancia y alta disponibilidad resultante.

La FIG. 18 ilustra un procedimiento de lote de punto de extremo de aplicación de objetivo de ejemplo que involucra un clasificador 304 y un manejador de salud y carga 314 de una unidad de balanceo de carga 106. Después

de que el manejador de salud y carga 314 ha adquirido un caché de salud y carga consolidado 1208, la información de salud y carga 1206 de ésta es utilizada en la selección de puntos finales de aplicación para nuevas solicitudes/conexiones.

Como se describió anteriormente con referencia a la FIG. 13B, el caché de salud y carga consolidado 1208 incluye información de salud y carga cacheada 1206 para múltiples anfitriones 108. Para facilitar la creación y actualización de cachés de salud y carga consolidados 1208 de la información de salud y carga 1206 que se origina de anfitriones múltiples 108, la información de salud y carga 1206 allí es organizada de tal forma que ésta pueda ser accedida por el identificador de cada anfitrión 108. Sin embargo, la información de salud y carga 1206 allí también es organizada de tal forma que ésta pueda ser acesada por el tipo de aplicación 316 con el fin de facilitar la selección de punto de extremo de la aplicación.

En otras palabras, el manejador de salud y carga 314 es capaz de acceder información de salud y carga 1206 sobre una base por aplicación 316 a través de la información de

salud y carga 1206 para anfitriones múltiples 108. Una vez que la información de salud y carga 1206 para una aplicación dada 316 se ha accedido por cada anfitrión 108, la localización de las solicitudes de conexión entrante se pueden desarrollar de acuerdo con tal información de salud y carga 1206. Por ejemplo, los puntos finales posibles para la aplicación dada 316 se pueden localizar en solicitudes de conexión entrantes mediante la selección de los puntos finales de la aplicación dada 316 con consideración de la capacidad de carga relativa disponible entre los puntos finales saludables para la aplicación dada 316.

En una implementación descrita, el clasificador 304 hace la solicitud de lote de punto de extremo de aplicación objetivo 1802 al manejador de salud y carga 314. Como se ilustró, la solicitud de lote de punto de extremo de aplicación objetivo 1802 incluye (i) una dirección y puerto IP virtuales, (ii) un protocolo, y (iii) información de especificación de protocolo. La solicitud de lote de punto de extremo de aplicación objetivo 1802 identifica por lo tanto un tipo de aplicación 316 al cual están dirigidas las solicitudes de conexión entrantes.

El manejador de salud y carga 314 recibe solicitud de lote de punto de extremo de aplicación objetivo 1802 y selecciona al menos un punto de extremo físico que corresponde al tipo identificado de aplicación 316 utilizando una cualquiera o más de muchos mecanismos de selección. Para reducir la latencia, el manejador de salud y carga 314 selecciona un lote de puntos finales de aplicación a ser utilizados sobre un número de solicitudes de conexión entrantes. Este lote es suministrado del manejador de salud y carga 314 para el clasificador 304 utilizando la respuesta de lote de punto de extremo de aplicación objetivo 1804. Como se ilustró, la respuesta del lote de punto de extremo de aplicación objetivo 1804 incluye un lote de direcciones y puertos IP físicos (tales como los puntos finales IP1, IP2, e IP3) para el tipo identificado de aplicación 316.

El lote para la respuesta de lote de punto de extremo de aplicación 1804 puede ser completado utilizando uno o más esquemas de lote. Por vía de ejemplo, son ilustrados un esquema de lote de señal 1806 y un esquema de lote de porcentaje 1808. El esquema de lote de señal 1806 es un esquema de lote basado en unidad, y el esquema de lote de porcentaje 1808 es un esquema de lote basado en tiempo.

El esquema de lote de señal 1806 localiza las señales para cada punto de extremo saludable IP1, IP2, e IP3 que responden a sus índices de carga y capacidad relativa. Para el ejemplo ilustrado, de la capacidad disponible total, el IP1 tiene 40% de capacidad disponible, el IP2 tiene 35% de capacidad disponible, y el IP3 tiene 25% de capacidad disponible. De esta manera, el número total de señales se divide entre estos porcentajes. El número total de señales se puede suministrar como parte de una solicitud de lote de punto de extremo de aplicación objetivo 1802 o se puede determinar por un manejador de salud y carga 314.

Cualquier valor para el número total de señales se puede utilizar, tal como 10, 45, 100, 250, 637, 1000, y así sucesivamente. Este valor puede ser establecido en dependencia del número de solicitudes de conexión por segundo y la velocidad/frecuencia en la cual la aplicación saludable y/o la carga está cambiando. El clasificador 304 "utiliza"/consume una señal cuando responde a cada solicitud de conexión con una localización de punto de extremo de aplicación hasta que las señales se han agotado; el clasificador 304 entonces

solicita otro lote de señal que utiliza solicitud de lote de punto de extremo de aplicación objetivo 1802.

El esquema de lote de porcentaje 1808 determina la capacidad relativa disponible de manera similar. Sin embargo, en lugar de las señales, estas capacidades relativas disponibles determinadas por punto de extremo de aplicación son suministradas al clasificador 304 junto con un temporizador de duración 1810. El clasificador 304 localiza los puntos finales de aplicación objetivo en las solicitudes de conexión entrantes de acuerdo con estos porcentajes de capacidad relativa disponibles hasta la expiración del temporizador de duración 1810.

Para el esquema de lote de porcentaje 1808, el clasificador 304 mantiene un registro de corrimiento de las localizaciones de punto de extremo de aplicación para adherir a los porcentajes de lote y mantener la pista del tiempo durante la duración del temporizador 1810. Cuando el temporizador expira, el clasificador 304 solicita entonces otro lote de porcentaje utilizando solicitud de lote de punto de extremo de aplicación de objetivo 1802.

Se debe notar que el esquema de lote de señal 1806 puede también utilizar un tiempo límite. Si las señales de lote son demasiado viejas, ellas se deben descartar y se adquieren nuevas. De otra manera, el clasificador 304 puede consumir señales anticuadas que fueron previamente localizadas con base en la información de salud y carga que es habitualmente demasiado desactualizada. El uso de lotes de punto de extremo de aplicación por el clasificador 304 se describe adicionalmente adelante en la sección titulada "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud".

Rastreo de Sesión de Ejemplo

Esta sección describe como la información de estado de anfitrión, tal como la información de sesión, se puede recolectar y utilizar en el balanceo de carga de red. Esta sección referencia primariamente las FIGS. 19-24 e ilumina la funcionalidad de preservación de afinidad de sesión tal como aquella suministrada por el rastreador de sesión 308 (de la FIG. 3). Como se describió anteriormente con referencia a las FIGS. 1-3, cada anfitrión 108 hospeda una o más aplicaciones 316 que suministra el o los servicios a los clientes 102. El

rastreador de sesión 308 utiliza información de sesión que se relaciona con los contextos para las conexiones establecidas entre las aplicaciones 316 y los clientes 102 para ciertas implementaciones descritas de balanceo de carga de red.

La FIG. 19 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de sesión 1902. En la conexión [1], el cliente 102(1) es mostrado haciendo una nueva conexión con el anfitrión 108(2) por vía de la infraestructura de balanceo de carga 106.

La infraestructura de balanceo de carga 106 puede estar comprendida de una o más unidades de balanceo de carga 106. Cuando la solicitud de conexión llega a la infraestructura de balanceo de carga 106, la solicitud es típicamente enrutada a un anfitrión 108 que utiliza funcionalidad de balanceo de carga de red que responde a la información de salud y/o carga de los anfitriones 108 y/o las aplicaciones 316 (no explícitamente mostradas en la FIG. 19) de ésta.

Cuando se hace la conexión [1], se establece una sesión entre el cliente 102(1) y la aplicación de servicio 316

que está en el anfitrión 108(2) en este ejemplo. La sesión suministra un contexto para el intercambio de comunicación entre el cliente 102(1) y el anfitrión 108(2). La información para el contexto de la sesión se almacena en el anfitrión 108(2). Cuando se completa la conexión [1], el contexto de la sesión no se puede utilizar de nuevo. De otra parte, el contexto de la sesión puede ser útil de nuevo si el cliente 102(1) intenta iniciar otra conexión con los anfitriones 108 para el servicio suministrado por la aplicación 316. Si esta otra conexión no está enrutada al mismo anfitrión 108(2) que almacena ese contexto de sesión, el cliente 102(1) tiene que establecer un nuevo contexto de sesión, que puede consumir tiempo, ser intensivo en procesamiento de datos, y/o frustrar al usuario humano del cliente 102(1). Con el balanceo de carga de red basado en información de salud y/o carga, no existe mayor probabilidad que la oportunidad aleatoria de que una segunda conexión sea enrutada a 108(2).

Sin embargo, si la infraestructura de balanceo de carga 106 tiene acceso a un mapeo entre información de sesión y anfitriones 108, la infraestructura de balanceo de carga 106 puede enrutar solicitudes de conexión que se

relacionan con sesiones previamente establecidas al anfitrión apropiado 108. Alguna información de sesión se puede inferir de los contenidos de los paquetes que fluyen a través de la infraestructura de balanceo de carga 106. Sin embargo, esta aproximación es imprecisa y riesgosa por un número de razones. Primero, el establecimiento de la sesión y la terminación es simplemente inferido. Segundo, algunas sesiones no son oficialmente terminadas con una indicación apropiada que está incluida en un paquete. Por ejemplo, algunas sesiones simplemente expiran. Tercero, los paquetes que están siendo transmitidos desde el anfitrión 108(2) al cliente 102(1) pueden tomar una senda que no incluye la infraestructura de balanceo de carga 106, que precluye cualquier entrometida de tales paquetes mediante la infraestructura de balanceo de carga 106 para la información de sesión.

Como se muestra en la FIG. 19, los anfitriones 108 proporcionan información de sesión (SI) 1902 a la infraestructura de balanceo de carga 106. Utilizando la información de sesión 1902 de los anfitriones 1080, un preservador de afinidad de sesión 1904 puede preservar la afinidad entre una sesión establecida y el anfitrión 108

sobre la cual se ha establecido la sesión. La información de sesión 1902 incluye un enlace entre o un mapeo de cada sesión establecida entre un cliente 102 y un anfitrión particular 108 a ese anfitrión particular 108. Este mapeo es accesible al preservador de afinidad de sesión 1904 como parte del mapeo de información de la sesión del anfitrión 1906. Ejemplos más específicos de información de sesión 1902 son suministrados adelante especialmente con referencia a las FIGS. 20, 22, 23A, y 23B.

En ciertas implementaciones descritas, para rastreo de sesión, la naturaleza lógica de los clientes 102 es pertinente. Como se anotó anteriormente con referencia a la FIG. 1, un cliente 102 puede ser un dispositivo específico y/o un usuario específico de un dispositivo. Consecuentemente, la afinidad de sesión para un cliente usuario 102 que está accediendo a los anfitriones 108 desde diferentes dispositivos puede aún ser preservado. Las continuaciones de la sesión utilizando información de sesión 1902 pueden por lo tanto ser aún efectuadas en escenarios proxy (por ejemplo, aquellos de algunos proveedores de servicio de Internet (ISP)).

Continuando con el ejemplo de conexión [1], la sesión establecida en el anfitrión 108(2) es provista con infraestructura de balanceo de carga 106 como la información de sesión 1902. Específicamente, un enlace/mapeo entre (i) el contexto de la sesión del cliente 102(1) y el anfitrión 108(2) y (ii) un identificador para el anfitrión 108(2) se crea en el mapeo de información de sesión de anfitrión 1906. Cuando una solicitud de conexión para la conexión [2] subsecuentemente arriba para el mismo contexto de sesión, el preservador de afinidad de sesión 1904 localiza este contexto de sesión en el mapeo de información de sesión de anfitrión 1906 e indaga que el anfitrión 108(2) está asociado con este contexto de sesión del enlace/mapeo.

En respuesta del mapeo del anfitrión 108(2) al contexto de la sesión solicitada como indagada por el preservador de afinidad de sesión 1904 del mapeo de información de sesión de anfitrión 1906, la conexión [2] es enrutada al anfitrión 108(2). En este sentido, preservar la afinidad de sesión es una prioridad mayor para la infraestructura de balanceo de carga 106 que las decisiones de balanceo de carga de red basadas en salud y carga de la aplicación. Sin embargo, la salud y/o carga pueden ser un

286

factor de balanceo de carga de red más importante que el rastreo de la sesión cuando, por ejemplo, la carga es extremadamente pesada o cuando la aplicación relevante de sesión y/o el anfitrión está en condición de fallo.

Muchos tipos de conexiones pueden estar relacionadas con la sesión. Ejemplos incluyen: una conexión TCP, una sesión de seguridad de capa de transporte (TLS)/SSL, una sesión PPTP, una sesión IPsec/L2TP, una sesión ISA, una sesión basada en cookie HTTP, una sesión de Terminal Server, una sesión definida por administrador, y así sucesivamente. Por vía de clarificación, la conexión TCP es considerada por ser una sesión de paquetes TCP. También, un modelo para definir sesiones por un administrador se pueden enumerar y soportar. Adicionalmente, las sesiones basadas en dirección IP del cliente que son delineadas por tiempos de espera también pueden ser soportadas. Este es un soporte de sesión relativamente no inteligente, pero es esperado por algunos usuarios.

Una solicitud de conexión de un cliente 102 varía por el tipo de sesión deseada. Por ejemplo, las sesiones del tipo "conexión TCP", la solicitud de conexión comprende

287

un paquete TCP. Para las sesiones del tipo "sesión SSL", la solicitud de conexión comprende una conexión TCP. Otras de tales solicitudes de conexión corresponden a otros tipos de sesión. Estos ejemplos también muestran como puede haber capas de sesión. En un nivel de sesión más bajo, un contexto de sesión para una conexión TCP puede incluir un TCP 4-tuplas, un número de sesión, un número de bytes enviados/recibidos, y así sucesivamente. A un nivel de sesión mayor, el contexto de sesión para la sesión SSL puede incluir un ID de sesión de 32 -byte, una clave pública del cliente 102 que es suministrada al anfitrión 108, y así sucesivamente.

La FIG. 20 ilustra una aproximación de balanceo de carga de red de ejemplo que involucra información de sesión de comunicación utilizando notificaciones 2006 y mensajes 2008. Las unidades de balanceo de carga múltiples 106(1), 106(2)...106(u) y anfitriones múltiples 108(1), 108(2)...108(n) son mostradas. Cada anfitrión respectivo 108(1), 108(2)...108(n) incluyen una o más de las respectivas aplicaciones 316(1), 316(2)...316(n) que son residentes allí y se ejecutan sobre éste. Las notificaciones 2006 son utilizadas para suministrar información de sesión de las aplicaciones 316, y los

mensajes 2008 son utilizados para suministrar información de sesión de los anfitriones 108 a las unidades de balanceo de carga 106.

Como se ilustró, cada anfitrión respectivo 108(1), 108(2)...108(n) incluye infraestructura de rastreo de sesión respectiva (STI) 2002(1), 2002(2)...2002(n). Cada infraestructura de rastreo de sesión respectiva 2002(1), 2002(2)...2002(n) incluye una tabla de sesión respectiva 2014(1), 2014(2)...2014(n) (aunque solamente la tabla de sesión 2014(1) es explícitamente ilustrada en la FIG. 19).

Cada unidad de balanceo de carga respectiva 106(1), 106(2)...106(u) incluye funcionalidad de enrutamiento de tráfico respectivo (TRF) 2012(1), 2012(2)...2012(u). La funcionalidad de enrutamiento de tráfico 2012 puede comprender, por ejemplo, clasificar y/o solicitar funcionalidad de enrutamiento, tal como aquella suministrada por el clasificador 304 y el enrutador de solicitud 306, respectivamente. Distribuida a través de las unidades de balanceo de carga 106(1), 106(2)...106(u) está un manejador de rastreo de sesión distribuida 2010.

289

En una implementación descrita, la funcionalidad de enrutamiento de tráfico 2012 y el manejador de rastreo de sesión distribuida 2010 son parte de la infraestructura de balanceo de carga 106. La infraestructura de rastreo de sesión 2002 también puede ser (por ejemplo, un remoto) parte de la infraestructura de balanceo de carga 106.

Un API 2004 se emplea para suministrar información de sesión de las aplicaciones 316 a la infraestructura de rastreo de sesión 2002. Utilizando el API 2204, las aplicaciones 316 son potenciadas para notificar la infraestructura de rastreo de sesión 2002 de la información de sesión, incluyendo varios cambios a ésta. Más específicamente, cada aplicación 316 es capaz de suministrar, y la infraestructura de rastreo de sesión 2002 es capaz de aceptar, notificaciones 2006.

Una notificación de que una sesión se ha establecido (o la notificación de establecimiento de sesión 2006(E)) se suministra de la aplicación 316 cuando una sesión es nuevamente establecida o abierta. La notificación de establecimiento de sesión 2006(E) incluye un identificador de sesión y opcionalmente un identificador de aplicación 316. Una notificación de que la sesión se

ha terminado (o notificación de terminación de sesión 2006(T)) se suministra de la aplicación 316 cuando una sesión es terminada o cerrada. La notificación de terminación de sesión 2006(T) también incluye un identificador de sesión y opcionalmente el identificador de aplicación 316.

Cuando la infraestructura de rastreo de sesión 2002 acepta una notificación de establecimiento de sesión 2006(E), esta inserta una entrada en la tabla de sesión 2014 para la nueva sesión. Una tabla de sesión de ejemplo 2014 se describe adicionalmente adelante con referencia a la FIG. 23A. Cuando la infraestructura de rastreo de sesión 2002 acepta una notificación de terminación de sesión 2006(T), esta remueve la entrada en la tabla de sesión 2014 de la sesión vieja.

La tabla de sesión 2014(1) es la fuente autorizada para la información de sesión 1902 con respecto a las aplicaciones 316(1) en el anfitrión 108(1). Existe generalmente demasiada latencia, sin embargo, para requerir la funcionalidad de enrutamiento de tráfico 2012 para contactar los anfitriones 108 para acceso a las tablas de sesión 2014 luego de recibir cada una de las

~~291~~

solicitudes de conexión entrantes que tienen una referencia de sesión. La información de sesión 1902 es por lo tanto cacheada en las unidades de balanceo de carga 106.

En las unidades de balanceo de carga 106, el manejador de rastreo de sesión distribuida 2010 cachea la información de sesión 1902 como parte de sus responsabilidades de manejo de rastreo de sesión. De manera general, el manejador de rastreo de sesión distribuida 2010 es una aplicación distribuida y/o servicio virtual que reside parcialmente en cada unidad de balanceo de carga 106. Para cada sesión lógica, el manejador de rastreo de sesión distribuida 2010 mantiene al menos una copia cacheada de la información de sesión por lo tanto de una manera confiable y escalable que puede ser fácilmente utilizada para enrutar tráfico como solicitudes de conexión entrantes que tienen una referencia de sesión son recibidas por la infraestructura de balanceo de carga 106.

Las comunicaciones entre los anfitriones 108 y las unidades de balanceo de carga 106 son efectuadas con un protocolo confiable que asegura que los mensajes 2008

enviados desde un anfitrión 108 lleguen a la unidad de balanceo de carga pretendida 106. Cada anfitrión 108 está unido al menos a una unidad de balanceo de carga específica 106 que es la unidad de balanceo de carga pretendida 106 para los mensajes 2008. Esta unión es creada al asignar una dirección IP de una unidad de balanceo de carga específica 106 de cada anfitrión 108 para enviar mensajes de rastreo de sesión 2008 entre la infraestructura de rastreo de sesión 2002 y el manejador de rastreo de sesión distribuida 2010. Para facilitar la alta disponibilidad de la infraestructura de balanceo de carga 106, si una unidad de balanceo de carga 106 falla, otra unidad de balanceo de carga 106 asume la dirección IP de la unidad de balanceo de carga fallida 106. La detección de falla para la dirección IP asume que se puede lograr utilizando un latido de corazón u otro esquema de monitoreo de vida.

Así, los mensajes 2008 comunican información de sesión 1902 de la infraestructura de rastreo de sesión 2002 al manejador de rastreo de sesión distribuida 2010. Por ejemplo, cuando la infraestructura de rastreo de sesión 2002 acepta una notificación de establecimiento de sesión 2006(E), esta también envía un mensaje de sesión arriba

293

2008(U) al manejador de rastreo de sesión distribuida 2010. El mensaje de sesión arriba 2008(U) incluye el identificador de sesión, un identificador de anfitrión, u opcionalmente otra información. Los contenidos para un mensaje de sesión arriba 2008(U) son descritos adicionalmente adelante con referencia a la FIG. 23B con respecto a la información que se puede almacenar para cada sesión por una implementación del manejador de rastreo de sesión distribuida 2010. Cuando la infraestructura de rastreo de sesión 2002 acepta una notificación de terminación de sesión 2006(T), esta también envía un mensaje de sesión abajo 2008(D) al manejador de rastreo de sesión distribuida 2010. Los mensajes 2008 se pueden enviar antes, durante, o después de que la infraestructura de rastreo de sesión 2002 modifica apropiadamente la tabla de sesión 2014 en respuesta a las notificaciones 2006.

La FIG. 21 es un diagrama de flujo 2100 que ilustra un método de ejemplo para el balanceo de carga de red que involucra comunicar la información de sesión utilizando notificaciones y mensajes. El diagrama de flujo 2100 incluye quince bloques 2102-2130. Aunque las acciones del diagrama de flujo 2100 se pueden desarrollar en otros

ambientes y con una variedad de esquemas, las FIGS. 1-3, y 19-20 son utilizadas en particular para ilustrar ciertos aspectos y ejemplos del método.

Por ejemplo, las acciones de cuatro bloques 2102-2104 y 2118-2120 son desarrolladas por una aplicación 316, las acciones de seis bloques 2106-2110 y 2122-2126 son desarrolladas por la infraestructura de rastreo de sesión 2002, y las acciones de los cinco bloques 2112-2116 y 2128-2130 son desarrolladas por un manejador de rastreo de sesión distribuida 2010. Las acciones de ocho de estos bloques 2102-2116 son primariamente dirigidas a la apertura de una sesión, y las acciones de siete de estos bloques 2118-2130 son primariamente dirigidas al cierre de una sesión.

En el bloque 2102, se abre una sesión. Por ejemplo, la aplicación 316 puede abrir una sesión con un cliente 102. En el bloque 2104, se proporciona una notificación de establecimiento de sesión. Por ejemplo, la aplicación 316 puede proporcionar una notificación de establecimiento de sesión 2006(E) a la infraestructura de rastreo de sesión 2002 utilizando API 2004 como una consecuencia de y/o en conjunto con la apertura de la sesión.

En el bloque 2106, la notificación de establecimiento de sesión es aceptada. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede aceptar notificación de establecimiento de sesión 2006(E) de la aplicación 316 de acuerdo con el API 2004. En el bloque 2108, se inserta una entrada en la tabla de sesión. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede insertar una entrada en la tabla de sesión 2014 para la sesión abierta. Los ejemplos de tal inserción son descritos adicionalmente adelante especialmente con referencia a la FIG. 23A. En el bloque 2110, se envía un mensaje de sesión arriba. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede enviar un mensaje de sesión arriba 2008(U) al manejador de rastreo de sesión distribuida 2010 utilizando un protocolo de comunicación confiable.

En el bloque 2112, el mensaje de sesión arriba es recibido. Por ejemplo, el manejador de rastreo de sesión distribuida 2010 puede recibir un mensaje de sesión arriba 2008(U) de la infraestructura de rastreo de sesión 2002 de acuerdo con el protocolo de comunicación confiable. En el bloque 2114, se crea una entrada de información de sesión. Por ejemplo, el manejador de

rastreo de sesión distribuida 2010 puede crear una entrada de información de sesión para la información de sesión cacheada 1902 en una o más unidades de balanceo de carga 106. Ejemplos de tal creación y subsecuente agregado son descritos adicionalmente adelante especialmente con referencia a las FIGS. 22 y 23B.

En el bloque 2116, el tráfico de red es enrutado con la información de sesión. Por ejemplo, la funcionalidad de enrutamiento de tráfico 2012 en conjunto con el manejador de rastreo de sesión distribuida 2010 puede utilizar información de sesión cacheada 1902, incluyendo la entrada de información de sesión creada, para enrutar solicitudes de conexión entrantes que tienen una referencia de sesión. Un ejemplo de tal enrutamiento de tráfico se describe adicionalmente adelante especialmente con referencia a la FIG. 24. Ejemplos adicionales se describen adelante en la sección titulada "Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud".

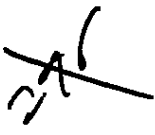
En el bloque 2118, se cierra la sesión. Por ejemplo, la aplicación 316 puede cerrar la sesión con el cliente 102. En el bloque 2120, se suministra una notificación de terminación de sesión. Por ejemplo, la aplicación 316

~~297~~

puede suministrar una notificación de terminación de sesión 2006(T) a la infraestructura de rastreo de sesión 2002 utilizando API 2004 como un consecuencia de y/o en conjunto con el cierre de la sesión.

En el bloque 2122, la notificación de terminación de sesión se acepta. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede aceptar la notificación de terminación de sesión 2006(T) de la aplicación 316 de acuerdo con el API 2004. En el bloque 2124, se remueve la entrada en la tabla de sesión. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede remover la entrada en la tabla de sesión 2014 para la sesión cerrada. En el bloque 2126, se envía un mensaje de sesión caída. Por ejemplo, la infraestructura de rastreo de sesión 2002 puede enviar un mensaje de sesión caída 2008(D) al manejador de rastreo de sesión distribuida 2010 utilizando el protocolo de comunicación confiable.

En el bloque 2128, se recibe el mensaje de sesión caída. Por ejemplo, el manejador de rastreo de sesión distribuida 2010 puede recibir el mensaje de sesión caída 2008(D) de la infraestructura de rastreo de sesión 2002 de acuerdo con el protocolo de comunicación confiable. En



el bloque 2130, la entrada de información de sesión es destruida. Por ejemplo, el manejador de rastreo de sesión distribuida 2010 puede destruir la entrada de información de sesión en la información de sesión cacheada 1902 en cualquiera de las unidades de balanceo de carga 106 que tienen la entrada de información de sesión. Los ejemplos de tal destrucción y subsecuente supresión son descritos adicionalmente adelante especialmente con referencia a las FIGS. 22 y 23B.

La FIG. 22 ilustra una aproximación de ejemplo a la información de sesión de manejo en unidades de balanceo de carga múltiple 106. Cada unidad de balanceo de carga respectiva 106(1), 106(2)...106(u) incluye una parte respectiva 2202(1), 2202(2)...2202(u) de un manejador de átomo distribuido (DAM) 2202. El DAM 2202 es una implementación de ejemplo de un manejador de rastreo de sesión distribuida 2010. Cada porción DAM respectiva 2202(1), 2202(2)...2202(u) incluye una parte respectiva 2206(1), 2206(2)...2206(u) de una tabla DAM (DAMT) 2206.

El DAM 2202 es una aplicación distribuida o un servicio virtual que maneja información de sesión 1902 de una manera confiable y escalable de tal forma que la

funcionalidad de enrutamiento de tráfico 2012 puede ser utilizada para preservar la afinidad de sesión. Por ejemplo, la funcionalidad de enrutamiento de tráfico 2012 puede acceder al DAM 2202 utilizando un API (no específicamente mostrado) para buscar o haber buscado el DAMT 2206. Las llamadas de función 2204, la operación del DAM 2202, y otros aspectos de la FIG. 22 son descritos adicionalmente adelante después de la descripción de las FIGS. 23A y 23B.

La FIG. 23A es una tabla de sesión de ejemplo 2014 como se ilustra en la FIG. 20. La tabla de sesión 2014 incluye "v" entradas 2302(1), 2302(2)...2302(v). Cada entrada 2302 se inserta en la infraestructura de rastreo de sesión 2202 en respuesta a una notificación de establecimiento 2006(E) que es aceptada de una aplicación 316. Cada entrada 2302 es removida por la infraestructura de rastreo de sesión 2002 en respuesta a una notificación de terminación de sesión 2006(T) que es aceptada de la aplicación 316.

Como se describió anteriormente, cada notificación de establecimiento de sesión 2006(E) incluye un identificador de sesión y opcionalmente un identificador

de aplicación 316. Cada entrada respectiva 2302(1), 2302(2)...2302(v) en la tabla de sesión 2014 incluye campos respectivos de (i) el identificador de sesión 2302(1I), 2302(2I)...2302(vI) y (ii) el tipo de sesión y/o la aplicación 2302(1T), 2302(2T)...2302(vT).

El tipo de sesión y/o la aplicación 2302(T) puede ser "TCP", "IPSEC", "Servidor Terminal", "cookie HTTP", un tipo de aplicación como se anotó anteriormente, y así sucesivamente. El identificador de sesión 2302(I) puede ser "<dirección IP fuente, puerto TCP fuente, dirección IP de destino, puerto TCP de destino>", "Cliente IP = 172.30.189.122", "User = 'joe_user'", "Cookie = '{b7595cc9-e68b-4eb0-9bf1-bb717b31d447} '", otro ejemplo, la identificación específica de aplicación para una sesión, y así sucesivamente. Para los tipos de conexión/sesión TCP, el identificador de sesión 2302(I) puede alternativamente ser una versión canónica del TCP 4-tuplas (para IPv4 o IPv6). Otros valores para los campos del identificador de sesión 2302(I) y el tipo de aplicación/sesión 2302(T) pueden ser alternativamente utilizados.

279

La FIG. 23B es una tabla de manejador de átomo distribuido de ejemplo (DAM) (DAMT) 2206 como se ilustró en la FIG. 22. La tabla DAM 2206 incluye "w" entradas 2304(1), 2304(2)...2304(w). Cada entrada de información de sesión 2304 es creada por un DAM 2202 en respuesta a un mensaje de sesión arriba 2008(U) que es recibido de la infraestructura de rastreo de sesión 2002. Cada entrada de información de sesión 2304 es destruida en respuesta a un mensaje de sesión caída 2008(D) que es recibida de la infraestructura de rastreo de sesión 2002. Como se describe adicionalmente adelante, las entradas de información de sesión 2304 de las tablas DAM 2006 pueden de hecho ser manipuladas por el DAM 2202 utilizando llamadas de función 2204.

Como se describió anteriormente, el mensaje de sesión arriba 2008(U) incluye el identificador de sesión, un identificador anfitrión, y opcionalmente otra información. Cada entrada de información de sesión respectiva 2304(1), 2304(2)...2304(w) en la tabla DAM 2206 incluye campos respectivos de (i) clave 2304(1K), 2304(2K)...2304(wK), (ii) datos 2304(1D), 2304(2D)...2304(wD), y (iii) metadatos 2304(1M), 2304(2M)...2304(wM). Por ejemplo, los valores para los

300

campos clave 2304(K) pueden ser cuerdas alfanuméricas, y valores para los campos de datos 2304(D) pueden ser bits binarios. Los valores para la clave 2304(K) pueden ser bits binarios, también.

La clave 2304(K) puede corresponder al identificador de sesión 2302(I). Los datos 2304(D) pueden corresponder al identificador anfitrión, tal como una dirección de red del anfitrión 108 en la cual existe el contexto de sesión. Los metadatos 2304(M) pueden corresponder a otra, información adicional. Ejemplos de tales metadatos 2304(M) incluyen datos que son utilizados internamente por el DAM 2202 para resolver las colisiones de átomo y seguir la vida del átomo (por ejemplo, por vía de un mecanismo de terminación de tiempo). (Esta caracterización de entradas 2304 por ser atómicas se describen más completamente en el párrafo siguiente). Más específicamente, los metadatos 2304(M) incluyen, entre otras cosas, la identidad de la entidad (por ejemplo, la instancia de la funcionalidad de enrutamiento de tráfico 2012) que se agregó a la entrada de información de sesión 2304 a la tabla DAM 2206.

En una implementación descrita, cada entrada de información de sesión 2304 es atómica en el sentido de que el DAM 2202 puede agregar, suprimir, copiar, etc., las entradas 2304 como un todo, pero el DAM 2202 no modifica ordinariamente una porción de cualquier entrada completa 2304. Así, las entradas atómicas 2304 son agregadas, suprimidas, copiadas, o manipuladas de otra manera, etc., a través de las tablas DAM 2206 y DAM 2202 con el fin de implementar la disponibilidad y escalabilidad para la implementación de preservación de afinidad de sesión.

Las llamadas de función 2204 (de la FIG. 22) son utilizables por el DAM 2202 para manipular las entradas atómicas 2304 de la tabla DAM 2206. Las llamadas de función 2204 se pueden comunicar desde una unidad de balanceo de carga 106 a una o más de otras unidades de balanceo de carga 106 de una manera punto a punto o multidifusión. Estas llamadas de función incluyen el átomo de agregar 2204(A), el átomo de suprimir 2204(D), el átomo de pregunta 2204(Q), y el átomo de regreso 2204(R).

El átomo de agregar 2204(A) toma la forma de AddAtom(clave, dato) y es utilizada para agregar una entrada atómica 2304^a una o más tablas DAM 2206. De esta manera, la llamada de función de átomo de agregar 2204(A) puede ser formulada como AddAtom(<session identifier>, dirección IP de anfitrión). El átomo de supresión 2204(D) toma la forma de DeleteAtom(clave) y es utilizada para suprimir una entrada atómica 2304 en una o más tablas DAM 2206. Las llamadas de función del átomo de supresión 2204(D) pueden ser dirigidas a aquellas tablas DAM 2206 conocidas por tener una copia de la sesión que es identificada por la clave 2304(K) o pueden ser multidifusión a todas las tablas DAM 2206 para asegurar que cualquier copia sea suprimida.

El átomo de pregunta 2204(Q) toma la forma de QueryAtom(clave) y es utilizada por una porción DAM particular 2202 cuando un identificador de sesión como se referenció por una solicitud de conexión entrante no es localizada en la tabla DAM local particular 2206 de la porción DAM particular 2202. Las llamadas de función del átomo de pregunta 2204(Q) son enviadas a una o más (incluyendo posiblemente todas) las otras porciones DAM 2202. En respuesta, cada una de las otras porciones DAM

2202 revisan su tabla DAM local 2206 para el identificador de clave/sesión. Si la clave es localizada por otra porción DAM 2202, esta otra porción DAM 2202 replica con un átomo de regreso 2204(R).

El átomo de regreso 2204(R) toma la forma de ReturnAtom(clave, datos) y es utilizada para responder a una llamada de función de átomo de pregunta 2204(Q). Las llamadas de función de átomo de regreso 2204(R) son utilizadas cuando una porción DAM 2202 tiene una entrada atómica solicitada 2304 en su tabla DAM local 2206 como identificada por una clave 2304(K) especificada en la llamada de función de átomo de pregunta 2204(Q). Las llamadas de función de átomo de regreso 2204(R) pueden ser dirigidas de regreso a la porción DAM 2202 que emite la llamada de función del átomo de pregunta 2204(Q).

Las llamadas de la función del átomo de agregado 2204(A) son utilizadas en respuesta a mensajes de sesión arriba 2008(U) y/o para replicar una entrada atómica 2304 a una o más de otras tablas DAM 2206. Tal replicación puede ser por redundancia y/o escalabilidad.

304

Las llamadas de función del átomo se supresión 2204(D) son utilizadas en respuestas a mensajes de sesión caída 2008(D) y también se pueden enviar a una o más de otras tablas DAM 2206. Después de que se suprime la entrada atómica 2304, la entrada atómica 2304 puede entrar en un estado de "zombie" de tal manera que este permanece con el DAM 2202 y opcionalmente de tal forma que esta es de hecho aún almacenada con la tabla DAM 2206 con una indicación zombie en los campos de metadatos 2304(M) de la entrada atómica 2304.

Así, una vez que es suprimida la entrada atómica 2304, esta puede permanecer en el DAM 2202 y en la tabla DAM 2206 en un estado zombie de tal forma que los paquetes para esta sesión (ahora muerta y cerrada) son dirigidos al anfitrión 108 del contexto de sesión para un tratamiento adecuado específico de protocolo. Por ejemplo, los paquetes TCP recibidos después de que una conexión TCP se ha caído son dirigidos al anfitrión 108 que termina la conexión. El anfitrión 108 puede responder apropiadamente -quizás al enviar un RST o al reenviar un FIN-ACK. El tiempo que la entrada atómica 2304 gasta en este estado zombie casa con el tiempo muerto específico de protocolo. (tan cercano como razonablemente sea

posible) del protocolo de comunicación confiable que es empleado.

Una llamada de función de átomo de pregunta 2204(Q) es utilizada para lograr una entrada atómica 2304 cuando una primera unidad de balanceo de carga 106 recibe una solicitud de conexión entrante que referencia una sesión que no está almacenada en la tabla DAM 2206 del DAM 2202 de la primera unidad de balanceo de carga 106. Se debe notar que otras porciones DAM 2202 pueden ser consultadas simultáneamente en una llamada de función de átomo de pregunta radiodifundida 2204(Q) o secuencialmente hasta que una llamada de función de átomo de regreso positivo 2204(R) es recibida.

La llamada de función de átomo de regreso 2204(R) es utilizada por una porción DAM 2202 de una segunda unidad de balanceo de carga 106 para suministrar una entrada atómica 2304 a la porción DAM 2202 de la primera unidad de balanceo de carga 106, donde la entrada atómica 2304 tiene una clave 2304(K) que es especificada por el identificador de clave/sesión en una llamada de función de átomo de pregunta 2204(Q), que fue previamente emitida por la porción DAM 2202 de la primera unidad de balanceo

~~6~~
C

de carga 106. Se debe anotar que otros componentes tales como la funcionalidad de enrutamiento de tráfico 2012, también pueden ser capaces de llamar funciones 2204, especialmente una llamada de función de átomo de pregunta 2204(Q), de acuerdo con un API o similar.

Las porciones DAM 2202 y las tablas DAM 2206 se pueden organizar y manejar en una miríada de maneras. Las maneras de ejemplo relacionan el cacheo local de replicación/redundancia luego de adquisición, aplicando un algoritmo hash para selección de sitio, y así sucesivamente. Los niveles cero, uno, dos, o más de replicación hasta la replicación completa se pueden emplear. Con un nivel cero de replicación, cada entrada atómica 2304 se almacena en el DAM 2202 que recibe un mensaje de sesión arriba 2008(U) por lo tanto sin replicación a otras porciones DAM 2202.

Con un primer nivel de replicación, cada entrada atómica 2304 se almacena en el DAM 2202 que recibe un mensaje de sesión arriba 2008(U) para éste, y es también agregado (copiado) a una otra porción DAM 2202 utilizando una llamada de función de átomo de agregar 2204(A). Este maneja un nivel de falla para una unidad de balanceo de

carga 106. Similarmente, con un segundo nivel de replicación, cada entrada atómica 2304 es almacenada en el DAM 2202 que recibe un mensaje de sesión arriba 2008(U) para éste, y es también agregado a otras dos porciones DAM 2202. Generalmente, la una, dos, etc., de otras porciones DAM 2202 a las cuales una porción DAM dada 2202 copia las entradas atómicas 2304 es predeterminada o seleccionada aleatoriamente. Terceros, cuartos, etc., niveles de replicación también se pueden emplear.

Adicionalmente, la replicación completa se puede emplear al tener cada entrada atómica 2304 que es almacenada en el DAM 2202 que recibe un mensaje de sesión arriba 2008(U) para ésta que también es agregada a cada una de las otras porciones DAM 2202. Varios factores son impactados por la selección del nivel de replicación: en la medida en que el nivel de replicación se incrementa, la disponibilidad se incrementa y la latencia se disminuye. De otra parte, el tráfico de red y el uso de memoria se incrementan ambos en la medida en que el nivel de replicación se incrementa.

Cuando la replicación completa no se emplea, se puede emplear el caché local luego de adquisición. Por ejemplo, cuando una porción DAM 2202 no localiza un identificador de sesión referenciado en su parte de la tabla DAM 2206, la porción DAM 2202 emite una llamada de función de átomo de pregunta 2204(Q) para lograr la entrada atómica 2304 asociada con el identificador de sesión referenciada por vía de la llamada de función de átomo de regreso 2204(R). En lugar de lanzar la entrada atómica lograda 2304 después del uso de ésta, la porción de DAM 2202 cachea la entrada atómica lograda 2304 en su parte de la tabla DAM 2206. Esta opción ofrece un compromiso entre los factores anteriormente enumerados.

Como otra opción cuando la replicación completa no se emplea, la aplicación del algoritmo hash para la selección de localización se puede emplear. La primera entrada atómica 2304 para una sesión se almacena en la porción DAM 2202 que recibe el mensaje de sesión arriba 2008(U). Las copias replicadas o las copias son enviadas por vía de unas llamadas de función de átomo agregado 2204(A) a la o las porciones DAM específicas 2202 utilizando una función de aplicación del algoritmo hash. De un rango total de valores hash posibles, cada porción

DAM 2202 le es asignado un subconjunto de estos. Cada identificador de sesión le es asignado un algoritmo hash utilizando alguna función de aplicación de algoritmo hash para llegar a un valor de aplicación de algoritmo hash. Este valor de aplicación del algoritmo hash es mapeado a las porciones DAM asignadas 2202. La porción DAM 2202 que primero agregó la entrada atómica 2304 replica entonces la entrada atómica 2304 a la porción o porciones DAM asignadas 2202.

Con la aplicación del algoritmo hash para la selección de localización, al menos una porción DAM 2202 que tiene una entrada atómica deseada 2304 localmente cacheada en su tabla DAM 2206 es conocible del identificador de sesión. Una llamada de función de átomo de pregunta 2204(Q) puede por lo tanto ser dirigida al lado de las porciones DAM conocidas 2202. Esto usualmente reduce el tráfico de red y/o latencia.

Esta aplicación del algoritmo hash para selección de localización se puede utilizar con una, dos, tres, o más niveles de replicación con cada uno de los rangos de valores de aplicación del algoritmo hash mapeando a uno, dos, tres, etc., diferentes porciones DAM 2202,

respectivamente. Adicionalmente, la aplicación del algoritmo hash para la selección de aplicación se puede utilizar con cacheo local luego de adquisición.

La FIG. 24 es un diagrama de flujo 2400 que ilustra un método de ejemplo para la información de sesión de manejo en las unidades de balanceo de carga múltiples. El diagrama de flujo 2400 incluye ocho bloques 2402-2416. Aunque las acciones del diagrama de flujo 2400 se pueden desarrollar en otros ambientes y con una variedad de esquemas de software, las FIGS. 1-3, 19, 20, 22, y 23 B son utilizadas en particular para ilustrar ciertos aspectos y ejemplos del método.

En el bloque 2402, una solicitud de conexión entrante con una referencia de sesión es analizada. Por ejemplo, la funcionalidad de enrutamiento de tráfico 2012 puede recibir una solicitud de conexión entrante que referencia una sesión abierta/establecida previamente de un tipo particular. En el bloque 2404, una tabla DAM local es buscada utilizando la referencia de sesión. Por ejemplo, para una unidad de balanceo de carga dada 106 y una funcionalidad de enrutamiento de tráfico 2012, la porción

~~31~~

DAM 2202 de ésta puede buscar su tabla correspondiente DAM 2206 buscando la referencia de sesión.

En el bloque 2406, se determina si la referencia de sesión casa con una clave de la tabla DAM local. Por ejemplo, la porción DAM 2202 puede buscar campos clave 2304(K) de entradas múltiples 2304 de la tabla DAM 2206 para determinar si la referencia de sesión casa con cualquiera de los valores de los campos clave 2304(K). Si es así, el diagrama de flujo 2400 continúa en el bloque 2412.

Si, de otra parte, la referencia de sesión no casa con ninguna clave, el diagrama de flujo 2400 continúa en el bloque 2408. En el bloque 2408, se hace una llamada de función de átomo de pregunta. Por ejemplo, la porción DAM 2202 puede hacer una llamada de función de átomo de pregunta 2204(Q) que incluye la referencia/identificador de sesión como la clave. La llamada de función de átomo de pregunta 2204(Q) puede ser enviada al menos a otra porción DAM 2202. El número, selección, orden, etc., de las porciones DAM de destino posible 2202 para el átomo de pregunta 2204(Q) pueden depender de las opciones (por ejemplo, nivel de replicación, aplicación del algoritmo

hash para selección de localización, cacheo local luego de adquisición, punto a punto versus multidifusión, etc.) empleada por el DAM 2202.

En el bloque 2410, se recibe un átomo retornado. Por ejemplo, puede ser recibida información proveniente de una llamada de función del átomo retornado 2204(R) que es emitida por otra porción DAM 2202. La otra porción DAM 2202 localizada exitosamente en una entrada atómica 2304 en su correspondiente tabla DAM 2206, con la entrada atómica localizada 2304 que tiene una clave que casa con la referencia de sesión. La información proveniente de la llamada de función del átomo retornado 2204(R) incluye valores provenientes del campo clave 2304(K) y el campo de datos 2304(D) para la entrada atómica localizada 2304. Estos valores corresponden al identificador de sesión de la sesión y la dirección de red del anfitrión 108 que es afinizada a la sesión.

En el bloque 2412, se extrae una entrada atómica. La entrada atómica se extrae de la tabla DAM local si un case fue encontrado localmente (en los bloques 2404 y 2406) o desde el átomo retornado si se encontró un case en cualquier otra parte (en los bloques 2408 y 2410). Por

ejemplo, una entrada atómica 2304 se puede extraer de la tabla DAM 2206 de la porción DAM 2202 o desde la información recibida por una llamada de función de átomo regresado 2204(R). La entrada atómica extraída 2304 se puede cachear en la tabla DAM local 2206 si es recibida como un resultado de una llamada de función de átomo retornado 2204(R).

En el bloque 2414, el anfitrión que tiene la afinidad de sesión con la sesión referenciada es indagado por la entrada atómica. Por ejemplo, un valor de los campos de datos 2304(D) de la entrada atómica extraída 2304 se puede indagar para indagar de esta manera una dirección de red del anfitrión afinizado 108. En el bloque 2416, la solicitud de conexión entrante es enrutada al anfitrión indagado. Por ejemplo, la funcionalidad de enrutamiento de tráfico 2012, y/o la funcionalidad de envío puede enrutar la solicitud de conexión entrante que tiene la referencia de sesión al anfitrión indagado y afinizado 108. La clasificación de ejemplo, enrutamiento de solicitud, y las funcionalidades de envío son descritas en la siguiente sección.

Clasificación de Ejemplo, Envío, y Enrutamiento de Solicitud

Esta sección describe como el enrutamiento de tráfico puede ser implementado para el balanceo de carga de red, que incluye con respecto a la alta disponibilidad de tal funcionalidad de enrutamiento de tráfico. La funcionalidad de enrutamiento de tráfico puede incluir clasificar y/o solicitar la funcionalidad de enrutamiento, especialmente en conjunto con la funcionalidad de envío. Esta sección primeramente hace referencia a las FIGS. 25-31. Esto ilumina la funcionalidad de una solicitud de enrutador 306 (de la FIG. 3), una interrelación entre las sesiones de rastreo y que utilizan la información de salud y carga cuando enrutan tráfico, implementaciones operacionales para interacciones de enrutamiento de tráfico con sesiones de información y/o información de salud y carga, procedimientos de falla para alta disponibilidad de infraestructura de balanceo de carga de red (que incluye fallas de manejo de componentes de clasificación, envío, y/o solicitud de enrutamiento), configuraciones adicionales de infraestructura de balanceo de carga de red, y así sucesivamente.

RCS

La FIG 25 ilustra la infraestructura de balanceo de carga de red de ejemplo que tiene solicitud de funcionalidad de enrutamiento como se realizó mediante la solicitud de enrutador 306(H/S). Como se anotó anteriormente con referencia al tráfico de la funcionalidad de enrutamiento 2012, el tráfico de enrutamiento puede relacionarse para clasificar (por ejemplo, con envío) y/o solicitud de enrutamiento. Clasificar el paquete-nivel, en conjunto con el envío, se describió anteriormente con referencia particular a la FIG. 4. La solicitud de enrutamiento se describe aquí con particular referencia a la FIG. 25.

La solicitud-nivel de enrutamiento ocurre en un nivel superior diferente del paquete-nivel de enrutamiento. Generalmente, una solicitud de enrutador 306 actúa como un Proxy para una aplicación 316 que corre en un anfitrión 108. La solicitud de enrutador 306 termina las conexiones TCP, analiza (quizás parcialmente) cada solicitud de un cliente 102, y retransmite cada solicitud al anfitrión 108. La solicitud del enrutador 306 puede desempeñar un pre-procesamiento en la conexión, tal como la descrición SSL. También, la solicitud del enrutador 306 puede escoger absorber ciertas solicitudes (por

26

ejemplo, la solicitud del enrutador puede mantener un caché de respuestas), y este puede "arbitrariamente" modificar las solicitudes antes de enviarlas a los anfitriones 108.

La solicitud de los enrutadores 306 usualmente son aplicación específica, y ellas pueden ser más bien abiertas en la forma en que ellas sean capaces de hacerlo. Sólo por vía de ejemplo, una clase sencilla de solicitud de enrutadores 306 - HTTP/SSL solicita de los enrutadores 306 (H/S)- son dirigidos en la siguiente descripción. Como se ilustra, un cliente 102 que tiene una dirección de red C1 se comunica a través de la red 104 con el anfitrión 108(1) y 108(2) que tienen direcciones de red H1 y H2, respectivamente. Las comunicaciones se efectúan vía infraestructura de balanceo de carga que incluye una solicitud de enrutador HTTP/SSL 306 (H/S).

La solicitud de enrutador HTTP/SSL 306(H/S) termina el tráfico HTTP y SSL, descripta el tráfico SSL, examina cada solicitud HTTP del cliente 102, aplica las reglas de aplicación específica para clasificar cada solicitud y determinar el "mejor" punto de extremo para tal solicitud

mientras se tiene en cuenta el punto de extremo de la solicitud de información de salud y carga, y presenta la solicitud al punto de extremo. La solicitud de presentación al punto de extremo utiliza una conexión TCP separada de una originada por el cliente 102 (la última conexión se terminó en la solicitud del enrutador HTTP/SSL 306(H/S)). Estas acciones pueden ser consideradas como el equivalente lógico a las acciones desarrolladas por un clasificador 304, pero aparece una diferencia en que estas acciones en la solicitud del enrutador HTTP/SSL 306(H/S) están ocurriendo en el nivel de solicitud lógica para cada solicitud dentro de la conexión TCP. La solicitud del enrutador HTTP/SSL 306(H/S) y la solicitud de los enrutadores 306 generalmente, pueden usar la misma (i) aplicación de salud y carga (ii) infraestructura de rastreo de sesión que se utiliza por los clasificadores 304.

La solicitud de enrutador HTTP/SSL 306(H/S) está actuando como un intermediario entre el cliente 102 y los dos anfitriones 108(1) y, 108(2). Este está manejando dos solicitudes para cliente 102 sobre una conexión sencilla TCP. En una implementación descrita, la solicitud de enrutamiento resultante involucra un número de acciones.



Primero, el cliente 102 establece una conexión http o https [1] a la solicitud de enrutador HTTP/SSL 306(H/S) y envía una solicitud #1 2502(1).

Segundo, la solicitud de enrutador HTTP/SSL 306(H/S) termina la sesión SSL (si el tráfico es SSL encriptado), analiza la solicitud #1 2502(1), y examina el contenido de la solicitud #1 2502(1). Tomando en cuenta la solicitud de salud y carga así como también las sesiones de información, la solicitud del enrutador HTTP/SSL 306(H/S) determina que el anfitrión 108(1) es el "mejor" anfitrión para esta solicitud particular #1 2502(1) en este ejemplo.

Tercero, la solicitud de enrutador HTTP/SSL 306(H/S) establece una conexión secundaria TCP [2] al anfitrión 108(1). Este puede alternativamente utilizar una conexión existente [2] al anfitrión 108(1). La solicitud de enrutador HTTP/SSL 306(H/S) luego envía una por ejemplo versión no encriptada de la solicitud #1 2502(1) al anfitrión 108(1). Cuarto, el anfitrión 108(1) responde con una respuesta #1 2504(1). Quinto, la solicitud de enrutador HTTP/SSL 306(H/S) encripta esta respuesta #1

2504(1) y la envía de regreso al cliente 102 sobre la conexión TCP [1].

Sexto, el cliente 102 envía otra solicitud, la solicitud #2 2502(2). La solicitud #2 2502(2) es manejada similarmente al manejo de la solicitud #1 2502(1), excepto que la solicitud del enrutador HTTP/SSL 306(H/S) selecciona el anfitrión 108(2). La selección diferente puede ser porque el anfitrión 108(1) está ahora fallando o es cargado más pesadamente, ya que la solicitud #2 2502(2) está dirigida a una URL diferente que la solicitud #1 2502(1), y así sucesivamente. De otra forma, la solicitud de enrutador HTTP/SSL 306(H/S) establece otra conexión secundaria TCP, pero esta conexión secundaria TCP [3] es al anfitrión 108(2). La solicitud no encriptada #2 2502(2) se enruta al anfitrión 108(2), y se recibe una respuesta #2 2504(2) como un resultado. Una versión encriptada de la respuesta #2 2504(2) es luego enviada al cliente 102 desde la solicitud de enrutador HTTP/SSL 306(H/S).

Séptimo, el cliente 102 cierra la conexión TCP [1] con la solicitud del enrutador HTTP/SSL 306(H/S). La solicitud del enrutador HTTP/SSL 306(H/S) (en algún tiempo futuro)

cierra la conexión [2] y [3] que hizo a los anfitriones 108(1) y 108(2), respectivamente, en nombre del cliente 102. La conexión TCP [2] puede ser cerrada alternativamente después de la solicitud del enrutador HTTP/SSL 306(H/S) decide abrir/usar la conexión TCP [3] para la solicitud #2 2502(2).

Debido a que una solicitud de enrutador HTTP/SSL 306(H/S) termina la conexión http/https, la solicitud de enrutador HTTP/SSL 306(H/S) puede hacer más que enrutar solicitudes. Por ejemplo, la solicitud de enrutador HTTP/SSL 306(H/S) puede mantener potencialmente su propio caché de respuestas (por ejemplo, con un mecanismo fuera de banda para invalidar el caché). Como se anotó en el ejemplo anterior, la solicitud de enrutador HTTP/SSL 306(H/S) también puede potencialmente enrutar diferentes clases de solicitudes a diferentes conjuntos de anfitriones 108 basados en por ejemplo la URL solicitada. Recíprocamente, la solicitud del enrutador HTTP/SSL 306(H/S) puede potencialmente agregar solicitudes de muchas conexiones de cliente de vida corta y enviarlas sobre unas pocas, conexiones TCP larga duración a los anfitriones 108. Tal agregación de conexión puede reducir

el de procesamiento de la conexión TCP hacia delante en los anfitriones 108.

La solicitud de enrutadores de otras clases puede corresponder a cualquier otro protocolo de ejemplo además del HTTP. Por ejemplo, una solicitud de enrutador puede ser una solicitud de enrutador SOAP. Las solicitudes de enrutadores SOAP funcionan análogamente a una solicitud de enrutador HTTP/SSL 306(H/S). Sin embargo, las solicitudes de enrutadores SOAP se especializan en enrutar el tráfico SOAP. Las solicitudes de enrutadores SOAP entienden los encabezados SOAP y hacen enrutar las decisiones basadas en los encabezados SOAP así como también la solicitud de salud y carga.

Tanto la clasificación del paquete-nivel como el envío (o enrutamiento de paquete-nivel) y el enrutamiento de la solicitud-nivel pueden suministrar alguna forma de balanceo de carga de capa-7. El balanceo de carga capa-7 se describe además adelante en la sección titulada "Migración de Conexión Ilustrativa con Canalización Opcional y/o Balance de Carga de Nivel de Aplicación". El paquete-nivel de enrutamiento suministra un acceso de sólo lectura a la porción inicial de los datos de

conexión TCP del cliente y suministran un enrutamiento de solicitud-nivel que lee y modifica el acceso a una corriente de datos completa.

Típicamente el enrutamiento del paquete-nivel tiene varias ventajas sobre el enrutamiento de solicitud-nivel. Estas ventajas incluyen transparencia (paquetes de cliente se distribuyen a anfitriones tal cual, preservar las direcciones de fuente y destino IP y los números de puerto), bajo procesamiento hacia delante (generalmente, dirigir o enviar tráfico involucra una búsqueda de ruta), baja latencia (paquetes individuales son enviados, y los paquetes no son puestos en fila una vez que el destino de la conexión TCP ha sido determinado) y alta disponibilidad (generalmente, una falla en el enviador no termina la conexión TCP). El enrutamiento por nivel de solicitud, por otro lado, típicamente tiene las siguientes ventajas sobre el enrutamiento por nivel de paquete: una habilidad para examinar una corriente de datos completa que fluye hacia y desde el cliente; y una habilidad para transformar una corriente de datos, y aún dividir la corriente de datos entre múltiples anfitriones o agregar corrientes de datos de múltiples clientes.

La FIG. 26 es un diagrama de flujo 2600 que ilustra un método ejemplar para enrutar los paquetes que llegan con respecto a (i) la información de sesión y (ii) la información de salud y carga. El diagrama de flujo 2600 incluye ocho bloques 2602 a 2616. Aunque las acciones del diagrama de flujo 2600 pueden llevarse a cabo en otros ambientes y con una variedad de esquemas de software, las FIGS. 1-3, 12, 18-20, 22, y 23B son usadas en particular para ilustrar ciertos aspectos y ejemplos del método.

En el bloque 2602, un paquete que llega es recibido. Por ejemplo, un paquete desde un cliente 102 puede ser recibido en el envidor 302 de una unidad de balanceo de carga 106. En el bloque 2604, se determina si el paquete recibido corresponde a una sesión preexistente. Por ejemplo, el envidor 302 puede consultar una tabla DAM local 2206() para determinar que el paquete recibido ya es parte de una sesión TCP/IP.

Adicionalmente, el envidor 302 puede consultar la tabla DAM local 2206() y determinar que el paquete recibido no era ya parte de una sesión TCP/IP. En este caso, el envidor 302 suministra el paquete recibido a un clasificador 304, el cual verifica por una afinidad de

sesión de nivel mayor para el paquete recibido si éste tiene una referencia de sesión. Ejemplos de estas acciones se describieron anteriormente con referencia particular a la FIG. 24 y más adelante con referencia particular a las FIGS. 27 y 28.

Si el paquete recibido es para una sesión preexistente (según se determinó en el bloque 2604), entonces el flujo continua al bloque 2606. En el bloque 2606, un anfitrión que tiene afinidad con la sesión preexistente es determinado. Por ejemplo, un anfitrión que tiene afinidad 108 puede ser determinado desde el DAM local 2206() y/o el DAM general distribuido 2206 por el enviador 302 o el clasificador 304.

En el bloque 2608, se ha determinado si el anfitrión que tiene afinidad es saludable. Por ejemplo, el clasificador 304 puede consultar una memoria caché de carga y salud consolidado 1208 para determinar si el anfitrión que tiene afinidad 108 es saludable, especialmente para aquellos paquetes recibidos que son parte de sesiones que son de un nivel lógico mayor que las sesiones TCP/IP. La acción o acciones de este bloque pueden llevarse a cabo en conjunto con un manejador de salud y carga 314.

Si el anfitrión que tiene afinidad es saludable (según se determino en el bloque 2608), entonces el flujo continúa hacia el bloque 2610. En el 2610, el paquete recibido es enrutado hacia el anfitrión que tiene afinidad. Por ejemplo, un enviador 302 (para las sesiones TCP/IP) o un clasificador 304 (para sesiones de mayor nivel) puede enrutar el paquete hacia el anfitrión que tiene afinidad 108. En una implementación alternativa, el clasificador 304 puede regresar el paquete recibido al enviador 302 para el enrutamiento hacia el anfitrión que tiene afinidad 108 aún para paquetes recibidos que son parte de sesiones de mayor nivel.

Si, de otro lado, el anfitrión que tiene afinidad no es saludable (como se determinó en el bloque 2608), entonces el flujo continúa al bloque 2612. También, si por otro lado, el paquete recibido no es para una sesión preexistente (según se determinó en el bloque 2604), entonces el flujo continúa hacia el bloque 2612. En el bloque 2612, un anfitrión es seleccionado como responsable para la información de salud y carga. Por ejemplo, el clasificador 304 puede seleccionar un anfitrión 108 desde y/o usando una asignación de

aplicación relacionada con una salud y carga (por ejemplo, desde una respuesta de asignación de punto de extremo de la aplicación objetivo 1804) lo que es logrado desde el manejador de salud y carga 314. Ejemplos para estas acciones se han descrito anteriormente con referencia particular a las FIGS. 19 y 18 y más adelante con referencia particular a la FIG. 30.

En el bloque 2614, el paquete recibido es enrutado hacia el anfitrión seleccionado. Por ejemplo, el clasificador 304 puede enrutar (opcionalmente por medio del enviador 302) el paquete hacia el anfitrión seleccionado 108. En el bloque 2616, una ruta para un trayecto de conexión hacia el anfitrión seleccionado es dirigida rectamente. Por ejemplo, el clasificador 304 puede agregar una entrada de información de sesión a la tabla DAM 2206, especialmente en la tabla DAM 2206() que es local para el enviador 302 que suministró el paquete recibido al clasificador 304. Esta entrada de información de sesión puede ser replicada de acuerdo con la política de redundancia instituida para un DAM 2202 (por ejemplo, de un rastreador de sesión 308).

Las acciones del bloque 2614 y aquellas del bloque 2616 se pueden llevar a cabo en el orden ilustrado específicamente, con aquellas de los bloques 2616 que se llevan a cabo antes de aquellas del bloque 2614, con las acciones parcial o completamente sobreponiéndose en cualquier orden y así en adelante. Debe notarse que las acciones llevadas a cabo por el clasificador 304 como se describió anteriormente pueden alternativamente ser llevadas a cabo por un enrutador de solicitud 306 (o más funcionalidad de enrutamiento de tráfico generalmente 2012).

La FIG. 27 ilustra un flujo de enrutamiento de tráfico ejemplar en la ausencia de fallas. Como se ilustró, uno o más conmutadores enterados del balance de carga 202(LBA) desde la infraestructura de balance de carga restante 106 (no indicada separadamente). El enviar y la funcionalidad de enviar y clasificar están distribuidas entre tres dispositivos o nodos. Un primer dispositivo incluye el enviador 302(1) y el clasificador 304(1). Un segundo dispositivo incluye el clasificador 304(2). Un tercer dispositivo incluye el enviador 302(2).

Con el clasificador 304(2) ejecutando sobre el segundo dispositivo y el envidor 302(2) ejecutando sobre el tercer dispositivo, cada dispositivo puede ser sintonizado especialmente para sus funciones respectivas. Por ejemplo, el hardware, software, firmware, algunas combinaciones de ellos, etc. del segundo dispositivo y del tercer dispositivo pueden ser adaptadas para soportar la funcionalidad deseada sin aprovisionamiento excesivo. Así, el tercer dispositivo que incluye el envidor 302(2) puede estar a tono con un conmutador y/o enrutador desde una perspectiva de capacidad de hardware, y el segundo dispositivo que incluye el clasificador 304(2) puede estar más a tono con un servidor y/o computador personal desde una perspectiva de capacidad de hardware.

Aunque se han mostrado como tres dispositivos que se les ha suministrado funcionalidad a través de cuatro componentes, las configuraciones alternativas lógicas y/o de nivel de dispositivo para las funcionalidades de envío y clasificación son aplicables al flujo de enrutamiento de tráfico ilustrativo que se ha descrito aquí para la FIG. 27. También, aunque los destinos de enrutamiento se muestran como anfitriones 108, las descripciones aquí de las implementaciones de enrutamiento pueden ser aplicadas

alternativamente de manera más general a un destino de siguiente nodo para el paquete y no necesariamente a un nodo final que consuma el paquete.

Una DAM 2202 de realización del rastreador de sesión 308 es usada para implementar la tabla DAM 2206. Sin embargo, los preservadores de afinidad de sesión 1904 en general son también aplicables al flujo de enrutamiento de tráfico ilustrativo de la FIG. 27. El enviador 302(1) incluye la porción de tabla DAM 2206(1), y el enviador 302(2) incluye la porción de tabla DAM 2206(2). Los paquetes que llegan son enrutados al anfitrión 108(1) o al anfitrión 108(2).

En una implementación descrita, la DAM 2202 es una tabla de "átomos" 2304 en memoria distribuida (por ejemplo, pares de valores de teclas, con megadatos opcionales) que tienen información de sesión. La DAM 2202 y la tabla DAM 2206 se describen adicionalmente anteriormente con referencia particular a las FIGS. 22-24. Cualquier nodo en el grupo de clasificadores 304 puede agregar, solicitar, y eliminar átomos 2304. El DAM 2202 mantiene una alta disponibilidad de la tabla DAM 2206 que incluye

rutas activas (por ejemplo, nivel TCP/IP) lo mismo que información de sesión de nivel mayor.

En (1), los conmutadores enterados del balance de carga 202(LBA) dirigen un paquete que llega al enviador 302(1).

En (2), el enviador 302(1) consulta su tabla de enrutamiento interna, la tabla DAM 2206(1). Cuando el enviador 302(1) no encuentra una entrada atómica 2304 para este paquete, envia el paquete a su clasificador asignado y/o asociado, el clasificador 304(1).

En (3), el clasificador 304(1) reconoce que el paquete en este ejemplo es un paquete TCP-SYN. El clasificador 304(1) por lo tanto trata el paquete como un inicio de una nueva conexión TCP desde un cliente 108. Usando la información de salud y carga desde un manejador de salud y carga 314 (no ilustrado explícitamente), el clasificador 304(1) determina que el anfitrión 108(1) debe recibir esta conexión TCP. El clasificador 304(1) actualiza la tabla DAM 2206(1) que sirve como la tabla de enrutamiento local para el enviador 302(1), y también inserta una entrada atómica 2304 que representa la ruta dentro de la DAM general 2206. Estas pueden ser operaciones separadas, una operación única en la cual las

326

sesiones de nivel TCP/IP de la tabla DAM 2206 están localizadas en los envidadores 302, y así en adelante. La DAM 2202 replica internamente esta ruta a una o más de los otros miembros del grupo de clasificadores 304 de acuerdo con su política de redundancia estipulada.

En (4), el envidador 302(1) dirige directamente paquetes subsecuentes para esta conexión al anfitrión 108(1) sin interactuar con el clasificador 304(1). El DAM 2202 puede ser usado para enmascarar, o por lo menos en parte, la falla de un envidador 302, un clasificador 304, o un par envidador/clasificador 302/304. La DAM 2202 también puede ser usada, por lo menos en parte, para preservar la conectividad del cliente si los conmutadores enterados del balance de carga 202(LBA) inician de manera inadvertida el envío de paquetes para una conexión establecida a un envidador diferente 302.

La Figura 28 ilustra un flujo de enrutamiento de tráfico ilustrativo en la presencia de fallas. En contraste con el flujo de enrutamiento de tráfico ilustrativo "libre de falla" de la Figura 27, una falla ha ocurrido en una porción de la infraestructura de balance de carga de red 106 (no identificado específicamente) de la Figura 28.

específicamente, el primer dispositivo, sobre el cual el emisor 302 (1) y el clasificador 304 (1) son residentes y ejecutan, fallan después de que se ha establecido la conexión ilustrada en la Figura 27. Esta falla es enmascarada, al menos en parte, por la DAM 2202.

En (1), los conmutadores conocedores del balance de carga 202 (LBA) detectan la falla del emisor 302 (1) e inician el envío de paquetes para la conexión en algún otro emisor 302 en grupo. En este ejemplo, el otro emisor 302 es el emisor 302 (2). Aunque la Figura 28 ilustra una situación de falla, los conmutadores del balance de carga 202 (LBA) pueden también enviar este tráfico al emisor 302(2) aún si el emisor 302(1) todavía está disponible. Este cambio de no falla inducida de los emisor 302 puede ocurrir, por ejemplo, por que los conmutadores conocedores del balance de carga 202 (LBA) "olvidan" la afinidad de este tráfico hacia el emisor 302(1). Las acciones de las notaciones (2)-(5) aplican tanto a la falla y a la situaciones de "afinidad olvidada".

En (2), el emisor 302(2) consulta su tabla de enrutamiento, la tabla DAM 2206(2). Cuando éste no

encuentra una ruta para este paquete, éste envía el paquete a su clasificador 304(2). En (3) el clasificador 304 (2) reconoce que este paquete es un paquete TCP "de conexión media" y el clasificar 304(2) solicita a la DAM 2202 para una ruta para este paquete. La DAM 2202 responde con la ruta para la conexión de una entrada atómica 2304 que está asociada con esta.

En (4), el clasificador 304(2) toca directamente la ruta en el envidor 302(2). Un protocolo ilustrativo para conectar directamente las rutas se describe más adelante. En (5), paquetes subsecuentes para esta conexión son dirigidos al envidor 302(2) y son enrutados directamente hacia el anfitrión correcto, que es el anfitrión 108(1) en este ejemplo, sin consultar el clasificador 304(2).

Generalmente, una ruta dirigida directamente a un protocolo para comunicaciones entre los clasificadores 304 y los envidores 302 incluye instrucciones para agregar y remover rutas. Más específicamente, una instrucción de agregar ruta es enviada desde un clasificador 304 a un envidor 302 para poder conectar una ruta desde el envidor 302 a un anfitrión de destino 108 para una conexión dada. Como vía de ejemplo, una

instrucción de agregar ruta puede ser suministrada al
enviador 302(2) desde el clasificador 304(2) como se
indicó en (4) en la Figura 28. La ruta (por ejemplo, una
clave y el valor correspondiente) es agregado a la tabla
DAM local 2206(2) para un acceso rápido por el enviador
302 (2) en el futuro. En este ejemplo, el clasificador
304(2) es un dispositivo separado desde el enviador
302(2), de modo que la ruta que dirige el protocolo puede
ser un protocolo interdispositivo. Sin embargo, la ruta
que dirige el protocolo puede también ser utilizada para
comunicaciones intradispositivo.

En una implementación descrita, el clasificador 304(2)
incluye un inventario de conexión 2802. Con el inventario
de conexión 2802, el clasificador 304(2) mantiene el
rastreo o la pista de las conexiones en cualquiera de los
enviadores 302 (tal como el enviador 302(2)) para el cual
el clasificador 304(2) dirige las rutas. Para permitir
que el clasificador 304(2) mantenga el rastro o la pista
de las conexiones, incluyendo la sensación de las misma,
el enviador 302(2) dirige paquetes finales para las
conexiones (tal como paquete TCP FIN) al clasificador
304(2). El clasificador 304(2) entonces elimina una
entrada en el inventario de conexión 2802 que corresponde

a la conexión y envía una instrucción de eliminar la ruta al envidor 302(2). Luego de recibir la instrucción de eliminar la ruta, el envidor 302(2) remueve o retira la ruta correspondiente en la tabla DAM 2206(2). De esta manera, la funcionalidad de clasificar en conjunto con la funcionalidad de rastrear la sesión puede controlar las tablas de ruta, y las rutas mismas, que son usadas para la funcionalidad de envío. Consecuentemente, la funcionalidad de envío que está separada en un dispositivo diferente puede ser efectuada usando hardware de alta velocidad, pero relativamente simple.

La Figura 29 ilustra un procedimiento de solución de falla ilustrativo adicional para una alta disponibilidad de la infraestructura del balance de carga de red 106. Los procedimientos de solución de falla para dos diferentes fallas, la falla 2902 y la falla 2906 se describen. Como se ilustró, la infraestructura de balance de carga de red 106 (no indicada separadamente) incluye 5 componentes: el envidor 302(1), el envidor 302(2), el envidor 302(3), el clasificado 304(1) y el clasificador 304(2).

En una implementación descrita cada uno de los cinco componentes 302(1), 302(2), 302(3), 304(1), y 304(2) corresponde a un dispositivo individual. Sin embargo, procedimientos de solución de falla similares aplican a ambientes en los cuales diferentes componentes de balance de carga comparten los dispositivos.

Inicialmente en [1], el enrutador/conmutador 202 dirige un paquete que sucede que está para una nueva conexión para el emisor 302(1). Debido a que el emisor 302(1) no tiene una ruta para esta conexión en esta tabla de enrutamiento local, éste envía el paquete al clasificador 304(1) como se indicó por la flecha doble en línea punteada en (1). El clasificador 304(1) verifica primero la información de sesión con referencia a un rastreador de sesión 308 para una afinidad de sesión de mayor nivel posible. En este ejemplo, el paquete no está afinizado con una sesión existente, de modo que el clasificador 304(1) selecciona un anfitrión 108 con referencia a la información de salud y carga con referencia al manejador de salud y carga 314.

Específicamente, el clasificador 304(1) selecciona el anfitrión 108(1) en este ejemplo. Asumiendo que el

paquete es para una conexión TCP/IP, esta sesión TCP/IP está enlazada al anfitrión 108(1) es agregada a la DAM 2202 usando una función de adición atómica 2204(A) llamada por el clasificador 304(1). El paquete inicial es enviado al anfitrión 108(1) por el clasificador 304(1) o al enviador 302(1). El clasificador 304(1) también dirige una ruta en la tabla de enrutamiento local del enviador 302(1). Los paquetes subsecuentes son enviados al anfitrión 108(1) por el enviador 302(1) y una interacción adición con el clasificador 304(1).

En algún tiempo durante la conexión [1], existe una falla 2902 en el enviador 302(1). Con el enrutador/conmutadores conocedores del balanceo de carga 202 (LBA), esta falla 2902 es detectada. Como resultado, en el punto 2904, el enrutador/conmutadores 202 dirige paquetes posteriores que deberían haber sido enviados al enviador 302(1) junto con la conexión [1] a otro enviador 302 que es el enviador 302(2) en este ejemplo.

El enviador 302(2) por lo tanto recibe los paquetes futuros junto con una conexión [2]. Debido a que el enviador 302(2) no tiene una entrada en esta tabla de enrutamiento local para los paquetes que fueron dirigidos

previamente al enviador 302(1), el enviador 302(2) envía el primer paquete recibido de la conexión [2] al clasificador al cual éste está asignado/asociado. En este ejemplo, el enviador 302(2) es asignado al clasificador 304(2) como se indicó por la fecha doble de línea punteada en (2).

El clasificador 304(2) usa una función de solicitud atómica 2204(Q) que llama para lograr la entrada atómica 2304 (no mostrada explícitamente) de la DAM 2202 que está asociada con la conexión existente TCP/IP. Esta entrada atómica 2304 es suministrada por medio de la DAM 2202 del rastreo de sesión 308 vía una llamada de función de retorno atómico 2204(R). El clasificador 304(2) extrae al anfitrión 108(1) que está afinizado con esta conexión TCP/IP desde la entrada atómica de retorno 2304. El clasificador 304(2) envía el primer paquete recibido para la conexión [2] al anfitrión 108(1) y conecta una ruta en la tabla de enrutamiento local del enviador 302(2). Los paquete subsecuentes son enviados al anfitrión 108(1) por el enviador 302(2) sin interacción adicional con el clasificador 304(2).

Las anteriores descripciones se enfocan predominantemente en fallas de los componentes individuales del enviador 302. Sin embargo, los componentes del clasificador 304 también pueden fallar. Por ejemplo, en algún punto, existe una falla 2906 en el clasificador 304(2). El enviador 302(2) detecta la falla 2906 cuando este intenta consumir los servicios de clasificación o por medio de dar noticia de una ausencia en alguna indicación de vida tal como un indicador tipo latido de corazón. Para manejar la falla 2906, el enviador 302(2) es reasignado o reasociado con un clasificador diferente 304, este es el clasificador 304(1) en este ejemplo. La funcionalidad de clasificación futura es suministrada al enviador 302(2) por el clasificador 304(1) como se indicó por la flecha doble en línea punteada (3).

La Figura 30 ilustra una implementación operacional ilustrativa de la interacción de enrutamiento de tráfico con la información de salud y carga. El enviador 302 y el clasificador 304 interactúan con el manejador de salud y carga 314 para poder enrutar los paquetes hacia los anfitriones 108(1), 108(2)....108(n). Aunque un enviador 302 y un clasificador 304 son ilustrados, la implementación operacional ilustrativa también es

330

aplicable a un enrutador de solicitudes 306 (o funcionalidad de enrutamiento de tráfico 2012 en general).

Como se ilustró, el anfitrión 108(1) incluye los puntos de extremo de aplicación IP1, IP3, e IP4 para la aplicación #1, la aplicación #1, y la aplicación #2, respectivamente. El anfitrión 108(2) incluye puntos de extremo de la aplicación IP2 e IP6 para la aplicación #1 y la aplicación #2, respectivamente. El anfitrión 108(n) incluye puntos de extremo de aplicación IP5 para la aplicación #2. Estos anfitriones 108(1), 108(2)... 108(n) y los puntos de extremo de aplicación IP1, IP2, IP3, IP4, IP5 e IP6 son monitoreados por el manejador de salud y carga 314 (por ejemplo, usando la infraestructura de salud y carga 1202, en la memoria caché consolidada de salud y carga 1208, etc.).

En una implementación descrita, en (1) el clasificador 304 solicita una o más asignaciones de punto de extremo de aplicación (por ejemplo, por medio de por lo menos una solicitud de asignación de punto de extremo de aplicación objetivo 1802) en un ambiente usando un esquema de asignación de señal 1806. El manejador de salud y carga

~~343~~

314 en este ejemplo, responde por medio de suministrar las asignaciones de señal 3002 (por ejemplo por medio de por lo menos una respuesta de asignación de punto de extremo de aplicación objetivo 1804).

Específicamente, una asignación de señal para la aplicación #1 3002(1) y una asignación de señal para la aplicación #2 3002(2) están disponibles en el clasificador 304. La asignación de señal para la aplicación #1 3002(1) inicialmente suministra 40 señales para IP1, 35 señales IP2 y 25 señales para IP3. La asignación de señal para la aplicación #2 3002(2) suministra 10 señales para IP4, 72 señales para IP5 y 18 señales para IP6. Para cada nueva conexión a la que se le asignado un enrutamiento a un punto de extremo de aplicación por medio de clasificador 304, una señal es consumida por el clasificador 304.

En (2), el emisor 302 recibe un paquete de entrada inicial para una nueva conexión debido a que no hay enrutamiento para esta nueva conexión presente en la porción de tabla DAM local 2206 del emisor 302, el emisor 302 envía el paquete inicial al clasificador 304 en (3).

En (4), el clasificador 304 (por ejemplo, después de determinar que el paquete inicial no incluye una referencia de sesión para una sesión de mayor nivel) selecciona un punto de extremo de aplicación (y por lo tanto un anfitrión 108) en respuesta a la información de salud y carga. Específicamente, para una nueva conexión que va a ser servida por la aplicación #1, el clasificador 304 puede seleccionar cualquiera entre IP1, IP2 e IP3 si una señal para el punto extremo respectivo todavía existe.

El clasificador 304 puede consumir señales en cualquiera de las muchas maneras posibles. Por ejemplo, el clasificador 304 puede usar un circuito cíclico sin importar el número de señales por punto de extremo. Alternativamente, el clasificador 304 puede simplemente iniciar desde IP1 y progresar hasta IP3 mientras se consumen todas las señales para cada punto de extremo antes de mover al siguiente punto de extremo en una enseñanza lineal. También el clasificador 304 puede consumir una señal desde un conjunto definido de puntos de extremo de señales que usualmente tiene el mayor número de señales en cualquier un momento. Usando esta

última enseñanza el clasificador 304 selecciona la IP1. Otras enseñanzas también pueden ser empleadas.

Como se ilustró, el clasificar 304 consume una señal para el punto de extremo de aplicación IP2. Consecuentemente, el conjunto de señal para IP2 es reducido de 35 señales a 34 señales a medida que una señal es consumida. También el paquete inicial para la nueva conexión debe ser enrutada al punto de extremo de aplicación IP2.

En (5A), el paquete inicial es enviado desde el clasificador 304 al punto de extremo de aplicación IP2 del anfitrión 108(2). Antes, durante, o después de este enviado, el clasificador 304 en (5B) establece una ruta para esta conexión en la porción de tabla DAM local 2206. El clasificador 304 puede también agregar una entrada atómica 304 para esta sesión dentro de la tabla DAM 2206 para propósitos de distribución y replicación. En (6), paquetes futuros para esta conexión /sesión son enviados desde el emisor 302 al punto de extremo de aplicación IP2 del anfitrión 108(2) usando la tabla de enrutamiento local del emisor 302 según se realizó por la porción de tabla DAM local 2206 en la Figura 30.

La Figura 31 ilustra unos mecanismos de alta disponibilidad ilustrativos para la infraestructura de balance de carga de red 106. Específicamente, la detección de falla ilustrativa 3104, el manejo de falla ilustrativo 3106, y la recuperación de fallas ilustrativa 3108 se muestran. Estos mecanismos de alta disponibilidad ilustrativos están descritos con respecto a los diferentes componentes de infraestructura de balance de carga de red 106. Los componentes de la infraestructura de balance de carga de red 106 incluyen un aviador 302, un clasificador 304, un enrutador de solicitudes 306, un rastreador de sesión 308, y un manejador de salud y carga 314.

En 3102(A), el enviador 302 sufre una falla local. En 3104(A), por lo menos un conmutador conocedor de balance de carga detecta la falla. Para manejar la falla local 3102(A) los paquetes son redireccionados u otros enviadores en 3106(A) por el conmutador conocedor del balance de carga. Para recuperar desde la falla del enviador 302, se enruta lo que fue almacenado localmente en el enviador 302 y es reconstruido en 3108(A) en los enviadores a los cuales los paquetes son redireccionados usando un administrador de rastreo de sesión distribuido

y una tabla del mismo tal como una DAM y una tabla DAM del mismo. El administrador de rastreo de sesión distribuido puede por lo tanto incluir redundancias de datos de uno o más niveles.

En 3102(B), el clasificador 304 sufre una falla local. En 3104(B), por lo menos un enviador detecta la falla. Para manejar la falla local 3102(B), los paquetes son redireccionados a otros clasificadores en 3106(B) por el enviador que detecta la falla. Para recuperar desde la falla del clasificador 304, la información de sesión que fue almacenada localmente en el clasificador 304 es reconstruida en 3108(B) en los clasificadores a los cuales los paquetes son redireccionados usando la DAM. Esta información de sesión puede ser, por ejemplo, información de sesión de un nivel más alto que las conexiones de línea base TCP/IP. También, tal información de sesión puede ser considerada como parte de la infraestructura de rastreo de sesión que está residente sobre el mismo dispositivo que el clasificador 304.

En 3102(C), el enrutador de solicitudes 306 sufre una falla local. En 3104(C), en por lo menos un enviador y/o un conmutador conocedor de balance de carga detectan la

falla. Para manejar la falla local 3102(C), los paquetes son redireccionados a otros enrutadores de solicitud en 3106(C) por el enviador y/o el conmutador conocedor del balance de carga. Las solicitudes lógicas usuales comunes individuales sobre las cuales el enrutador de solicitud 306 está trabajando la ocurrencia de la falla local 3102(C) puede perderse a menos que cada solicitud lógica individual sea replicada mientras la solicitud está siendo atendida. Para recuperar desde la falla del enrutador de solicitud 306, la información de sesión y/o las rutas que fueron almacenadas localmente en el enrutador de solicitudes 306 son reconstruidas en 3108(C) en los enrutadores de solicitud a los cuales los paquetes (y estas nuevas solicitudes lógicas) son redireccionadas. La información de sesión que se reconstruye puede ser efectuada usando la DAM. Nuevamente, tal información de sesión puede ser considerada como parte de la infraestructura de rastreo de sesión que está residente en el mismo dispositivo que el enrutador de solicitud 306.

En 3102(D), un rastreador de sesión 308 sufre una falla local. En 3104(D), en por lo menos un enviador y/o un clasificador detección detecta la falla. Por ejemplo, si

el rastreador de sesión 308 está residente en el mismo dispositivo de un clasificador, entonces un envidor u otro clasificador puede detectar la falla. Si el rastreador de sesión 308 está residente sobre un dispositivo separado, entonces un clasificador puede detectar la falla. Para manejar la falla local 3102(D), la redundancia de datos de uno o más niveles y la distribución a través de múltiples dispositivos se instituyen en 3106(D) para la información de sesión rastreada. Debe notarse que la redundancia y distribución son instituidas antes de que ocurra la falla 3102(D). Para recuperar desde la falla del rastreador de sesión 308, la información de sesión de las tablas de la DAM pueden ser redistribuidas y rereplicadas en 3108(D) a través de por lo menos dos dispositivos (si no está ya distribuida y suficientemente replicada) para poder manejar un segundo nivel de falla.

En 3102(E), el manejador de salud y carga 314 sufre una falla local. En 3104(E) por lo menos un clasificador y/o un enrutador de solicitud detecta la falla. Por ejemplo, un componente que está recibiendo información de salud y carga desde el manejador de salud y carga 314 puede detectar una falla si el manejador de salud y carga 314

no responde, especialmente si el manejador de salud y carga 314 está residente en un dispositivo diferente de aquel del componente que lo solicita. Para manejar la falla local 3102(E), la redundancia de daos de salud y carga guardados en caché y el manejo de la falla intrínseca se emplean en 3106(E) para la información de salud y carga.

Por ejemplo, cada manejador de salud y carga 314 puede incluir una memoria caché de información de salud y carga consolidada 1208 que duplica la información en las tablas de salud y carga 1204 sobre múltiples anfitriones 108. También, los consumidores de la información de salud y carga 1206 de un manejador de salud y carga dado 314 pueden estar localizados sobre el mismo dispositivo que el manejador de salud y carga 314 de modo que la falla del manejador de salud y carga 314 sea intrínsecamente aceptable. Similarmente, la versión autoritativa de una porción respectiva de la información de salud y carga 1206 está localizada sobre un anfitrión respectivo 108 de modo que la falla del anfitrión 108 convierte la falla de la porción respectiva de la información de salud y carga aceptable.

Para recuperar desde la falla del manejador de salud y carga 314, un componente dados de balance de carga de red que consume la información de salud y carga puede solicita un manejador de salud y carga diferente debido a que cada uno de tales manejadores de salud y carga incluye una memoria caché consolidada de información de salud y carga del manejador. También, cuando el manejador de salud y carga 314 es nuevamente accesible, el protocolo de mensaje 1500 puede ser usado en 3108(E) para reconstruir su memoria caché consolidada de información de salud y carga. Usando estos mecanismos de alta disponibilidad ilustrativos las fallas de los componentes de la infraestructura de balance de carga de red 106 pueden ser detectados, manejados, y recuperados desde la posibilidad de enmascarar tales fallas desde los clientes 102.

Migración de Conexión Ilustrativa

Con Canalización Opcional y/o Balance de Carga de Nivel de Aplicación

Esta sección describe como la manipulación de la conexión, tal como la migración de conexión, puede ser

utilizada en el balance de carga de red. Esta sección principalmente se refiere a las Figuras 32 a 39 e ilumina la funcionalidad de migración de conexión tal como está suministrada por el migrador de conexión 310 (de la Figura 3). Como se describió anteriormente con referencia a las Figuras 3 a 4, cada conexión de entrada en la infraestructura de balance de carga 106 puede ser terminada allí. Después de esto la conexión puede ser migrada a un anfitrión 108 de modo que la conexión es entonces terminada en el anfitrión 108. El migrador de conexión 310 es capaz de llevar a cabo esta migración de conexión y puede estar localizado parcialmente en los anfitriones 108 para efectuar la migración. Tal migración de conexión puede llevarse a cabo en conjunto en conjunto con el balance de carga a nivel de aplicación por un clasificador 304 y/o usando canalización por medio del canalizador 312.

La Figura 32 ilustra una enseñanza ilustrativa de balance de carga de red a nivel de aplicación con migración de conexión. El balance de carga a nivel de aplicación, o capa 7, pertenece al hacer decisiones de balance en carga con respecto a una aplicación que debe manejar una conexión. Para llevar a cabo el balance de carga a nivel

de aplicación, la infraestructura de balance de carga 106 usualmente toma en consideración una porción de datos de una conexión. A menos que la solicitud de enrutamiento se empleada, un clasificador 304 típicamente toma una parte en la porción inicial de una conexión y luego migra la conexión, en conjunto con el migrador de conexión 310, a un anfitrión seleccionado 108.

Para que el balance de de carga a nivel de aplicación en un ambiente con base en TCP generalmente los clasificadores 304 toman una porción inicial de un dato TCP del cliente cuando se decide a donde dirigir la conexión TCP del cliente. Así, la lógica a nivel de aplicación examina el dato del cliente y toma decisiones de balance de carga con base en este dato. Por ejemplo, si una conexión es una conexión HTTP (no encriptada), un clasificador 304 puede tomar una porción del encabezamiento HTTP de la primera solicitud HTTP en la conexión, y este puede tomar decisiones de enrutamiento con base en alguna porción del contenido del encabezamiento (por ejemplo, el URL, un cookie, etc.). Aunque el balance de carga a nivel de aplicación, la migración de conexión, y la canalización son aplicables a

otros protocolos, el TCP/IP es usado predominantemente en los ejemplos aquí.

Como se ilustró, la infraestructura de balance de carga 106 (no indicada específicamente) incluye un enviador 302, un clasificador 304, un canalizador 312, y un migrador de conexión 310 (y posiblemente por ejemplo un enrutador/conmutadores conocedores del balance de carga 202 (LBA)). El enviador 302 corresponde a la dirección IP virtual y envía paquetes a los anfitriones 108 de acuerdo con las selecciones de anfitrión hechas por el clasificador 304. Aunque no se muestra específicamente en la Figura 32 para claridad, los anfitriones 108 también incluyen funcionalidad de migrador de conexión 310 y funcionalidad de canalizador 312.

En una implementación descrita, el enviador 302, el clasificador 304 y el migrador de conexión 310 (en el clasificador 304 y en los anfitriones 108), junto con el software del protocolo TCP en el clasificador 304 y los anfitriones 108, cooperan para suministrar la migración de conexión. La migración de conexión ilustrada en la Figura 32 es para una conexión desde un cliente 102(1) que está inicialmente terminada en el clasificador 304.

Después de la migración de conexión, la conexión desde el cliente 102(1) es terminada en el anfitrión 108(1). Una vez que la conexión es terminada en el anfitrión 108(1), los paquetes para la conexión pueden ser canalizados usando el canalizador 312 (en el enviador 302 y el anfitrión 108(1)).

En (1), el cliente 102(1) envía un paquete SYN al enviador 302 para señalizar el inicio de una nueva conexión TCP. En (2), el enviador 302 dirige este paquete al clasificador 304. En (3), el clasificador 304 acepta la conexión TCP en nombre de un anfitrión 108 (cuya identidad todavía no se conoce debido a que el anfitrión objetivo real 108() todavía no ha sido seleccionado). En términos del protocolo TCP, el clasificador 304 envía un paquete SYN-ACK para el cliente 102(1).

En (4), el cliente 102(1) inicia el envío de datos. (El paquete inicial SYN puede también contener datos) los datos son procesados por el clasificador 304, que puede consultar la lógica específica de aplicación. La lógica específica de aplicación puede relacionarse con cualquiera de los anfitriones 108 que sea capaz de manejar mejor dichos tipos de solicitudes o conexiones.

Ya que, el clasificador 304 usa el dato, lo mismo que la información de salud y carga de la aplicación desde el manejador de salud y carga 314 y opcionalmente la información de sesión de aplicación desde el rastreador de sesión 308, para determinar un anfitrión 108 que este mejor adecuado para manejar esta conexión desde el cliente 102(1). En este ejemplo, el anfitrión 108(1) se ha seleccionado.

En (5), el clasificador 304 envía un "blob binario" que representa el estado de la conexión TCP en le anfitrión 108(1). Este estado de conexión es agregado con la cooperación desde una pila TCP en un clasificador 304 por el migrador de conexión 310. El blob binario contiene datos del cliente 102(1) que han sido reconocidos por el clasificador 304 y los parámetros TCP tal como las 4 tuplas TCP/IP, los números de secuencia inicial, y d e ahí en adelante.

En (6), un componente del migrador de conexión 310 en el anfitrión 108(1) (no mostrado explícitamente en la Figura 32) "inyecta" esta conexión dentro de una pila TCP en el anfitrión 108(1). Esta inyección del estado de conexión es llevada a cabo en cooperación con la pila TCP en el

:

.

:

.

357

anfitrión 108(1), haciéndola aparecer a las aplicaciones 316 en el anfitrión 108(1) que esta conexión fue aceptada originalmente por el anfitrión 108(1) mismo. El cliente 102(1) no está conciente de la migración de conexión.

En (7), el clasificador 304, en cooperación con la pila TCP en el clasificador 304, limpia silenciosamente el estado interno mantenido para esta conexión. El clasificador 304 también agrega una ruta en una tabla de enrutamiento local del emisor 302 que indique al anfitrión 108(1) como el destino de los paquetes para esta conexión.

En (8), paquetes subsecuentes para la conexión son enrutados por el emisor 302 hacia el anfitrión 108(1). Estos paquetes pueden ser tratados de la misma manera por el emisor 302 que aquellos paquetes para las conexiones que son clasificadas y enrutadas sin usar migración de conexión. Estos paquetes subsecuentes pueden opcionalmente ser canalizados desde el emisor 302 hacia el anfitrión 108(1) usando el canalizador 312. El canalizador 312 también se ha ilustrado (usando líneas punteadas) en el migrador de conexión 310 en el clasificador 304 debido a que ciertos parámetros usados

por el canalizador 312 pueden ser determinados durante una migración de conexión y/o pueden estar asociados con una conexión que está siendo migrada. Implementaciones ilustrativas para el canalizador 312 se describen más adelante con referencia particular a las Figuras 38 y 39.

La Figura 33 es un diagrama de flujo 3300 que ilustra un método ilustrativo para migrar una conexión desde un primer dispositivo a un segundo dispositivo. El diagrama de flujo 3300 incluye siete bloques 3302 a 3314. Aunque las Figuras 32 y 34 a 37 se enfocan principalmente en la migración de la conexión en un ambiente de balance de carga de red, la migración de la conexión según se describe aquí puede ser efectuada entre dos dispositivos en general en el cual cada uno incluye funcionalidad de migración de conexión, tal como el migrador de conexión 310.

En el bloque 3302, una conexión es aceptada por un primer dispositivo. Por ejemplo, un primer dispositivo puede terminar una conexión que entra de acuerdo con uno o más protocolos de una porción de pila de protocolos de una pila de red. En el bloque 3304, el dato es recibido para la conexión el primer dispositivo. Por ejemplo, este dato

puede ser recibido en un paquete inicial que solicita la conexión o uno o más paquetes que son recibidos subsecuentemente a una aceptación de la conexión.

En el bloque 3306, un estado de conexión para la conexión aceptada es agregado desde la pila de protocolo (o más generalmente desde la pila de red) en el primer dispositivo. Por ejemplo, un estado de protocolo de uno o más protocolos de la pila de protocolos puede estar compilado y agrado con cualquier dato recibido que haya sido reconocido. En el bloque 3308, el estado de conexión es enviado desde el primer dispositivo a un segundo dispositivo. Por ejemplo, la información agregada del estado de conexión puede ser enviada usando un protocolo de confiabilidad a un segundo dispositivo.

En el bloque 3310, el estado de conexión para la conexión que está siendo migrada es recibido desde el primer dispositivo en el segundo dispositivo. En el bloque 3312, el estado de conexión es inyectado dentro de una pila de protocolo (o más generalmente en la pila de red) del segundo dispositivo. Por ejemplo, la conexión puede ser rehidratada usando los protocolos de la pila de protocolos del segundo dispositivo de modo que los

~~348~~

programas anteriores del nivel de pila de protocolos no se dan cuenta de que la conexión es una conexión migrada. Más específicamente, el estado de protocolo puede ser infundido dentro de la pila de protocolo. El dato agregado del estado de conexión es también incorporado en el segundo dispositivo. En el bloque 3314 la conexión se continua en el segundo dispositivo. Por ejemplo, la conexión puede ser continuada en el segundo dispositivo como si la conexión no hubiera terminado previamente en otra parte.

La Figura 34 ilustra una enseñanza ejemplar para la migración de conexión desde una perspectiva de un dispositivo de origen 3400. La migración de la conexión en el dispositivo de origen 3400 es efectuada, por lo menos parcialmente, por el migrador de conexión 310. En una implementación descrita, el dispositivo de origen 3400 es un dispositivo que es parte de la infraestructura de balance de carga de red 106. Por ejemplo, el dispositivo de origen 3400 puede comprender un clasificador 304 posiblemente junto con un enviador 302, un enrutador de solicitudes 306, y de ahí en adelante.

Como se ilustró, el dispositivo de origen 3400 incluye como parte de su pila de red una interfaz de red física (PNI) 3410, un minipuerto PNI 3408, una interfaz de protocolo-hardware 3406 una pila de protocolos 3404, y una capa de conexión 3402. El dispositivo de origen 3400 también incluye una funcionalidad de balance de carga 106, tal como un clasificador 304 en un nivel de aplicación y un migrador de conexión 310. Específicamente, el migrador de conexión 310 incluye un controlador intermedio de migrador 3414 y un ajustador de migrador 3412. El migrador de conexión 310 es capaz de descargar una conexión desde un dispositivo de origen 3400.

En la implementación descrita, la interfaz de red física 3410 puede ser una tarjeta de interfaz de red (NIC) (por ejemplo, una NIC Ethernet), una interfaz inalámbrica, y etc. Aunque solamente una interfaz de red física 3410 es mostrada, un dispositivo dado puede realmente tener múltiples de tales interfaces de red físicas 3410 (por ejemplo, el dispositivo de origen 3400 puede ser un multihogar). Cada interfaz de red física 3410 corresponde típicamente a una o más direcciones de red físicas.

El minipuerto PNI 3408 es un módulo de software que entiende y hace interfaz con la realización de hardware específica de la interfaz de red física 3410. La interfaz protocolo-hardware 3406 es una capa que incluye una o más interfaces respectivas entre uno o más protocolos respectivos y el minipuerto PNI 3408.

La pila de protocolo 3404 incluye uno o más módulos respectivos que son cada uno dirigidos a uno o más protocolos respectivos. Ejemplos de tales protocolos se describen más adelante con referencia a las Figuras 36 y 37. En un contexto transciende, la pila de protocolo 3404 incluye un estado de protocolo 3420 para cada conexión que existe en el dispositivo de origen 3400. Una capa de conexión 3402 descansa entre un programa tal como una funcionalidad de balance de carga 106 y la pila de protocolo 3404. La capa de conexión 3402 suministra los API entre la funcionalidad de balance de carga 106 y la pila de protocolo 3404, y permite que los programas se registren para las conexiones, entre otras cosas.

El controlador intermedio de migrador 3414, o más generalmente el controlador de migrador 3414, está localizado en la capa de interfaz protocolo-hardware

3406. El ajustador del migrador 3412 está localizado de manera transparente entre la pila de protocolo 3404 y la capa de conexiones 3402.

Cuando un paquete inicial (no mostrado) que solicita una nueva conexión es presentado en el dispositivo de origen 3400, el paquete es dirigido hacia arriba desde la interfaz de red física 3410 hacia el minipuerto PNI 3408, por medio de la capa de interfaz protocolo-hardware 3406 y hacia la pila de protocolo 3404. A medida que el paquete atraviesa uno o más protocolos de la pila de protocolo 3404 el estado de protocolo 3420 es creado allí. También, como resultado de este paquete inicial o como una consecuencia de la funcionalidad de balance de carga 106 que acepta la conexión al tomar una porción de la solicitud, el dato 3416 llega al dispositivo de origen 3400.

En la operación, el controlador intermedio del migrador 3414 desvía una copia del dato 3416 a la lógica del migrador de conexión 310. Cuando la funcionalidad de balance de cargas 106 emite una llamada de función de conexión migrada, la llamada de función migrada es pasada a la capa más alta de la pila de protocolo 3404 de modo

que la agregación del estado de conexión 3418 puede iniciarse. El estado de protocolo 3420 es compilado desde uno o más protocolos de la pila de protocolo 3404. En una implementación TCP/IP el estado de protocolo 3420 puede incluir (i) puertos TCP de destino y fuente y direcciones IP (por ejemplo, una 4 tuplas TCP/IP), (ii) estado de ventana TCP, (iii) números iniciales de secuencia, (iv) información de tiempo de terminación, (v) ID del fragmento IP, (vi) información de enrutamiento, y (vii) etcétera.

La agregación del estado de conexión 3418 también agrega el dato 3416 que ha sido desviado al migrador de conexión 310 y que ya ha sido reconocido desde el dispositivo de origen 3400 (por ejemplo, por la funcionalidad de balance de carga 106). Este estado de conexión agregado 3418 incluye el estado de protocolo 3420 y el dato 3416 (y opcionalmente otra información relacionada con la conexión). El estado de conexión agregado 3418 es entonces enviado a blob binario 3422 retirado del dispositivo de origen 3400 hacia un dispositivo objetivo usando un protocolo confiable. Este blob binario 3422 puede también ser dado con un identificador de flujo si la conexión debe ser canalizada subsecuentemente con el

canalizador 312. Los identificadores de flujo con la canalización se describen más adelante con referencia particular a las Figuras 38 y 39.

La Figura 35 ilustra una enseñanza ejemplar para la migración de conexión desde la perspectiva de un dispositivo objetivo 3500. El dispositivo objetivo 3500 es similar al dispositivo de origen 3400 con respecto a las capas/módulos ilustrados, incluyendo el migrador de conexión 310. Como se ilustró sin embargo, por lo menos una aplicación 316 en un nivel de aplicación se le hace interfaz con la capa de conexiones 3402. El dispositivo objetivo 3500 puede por lo tanto comprender un anfitrión 108. También, el migrador de conexión 310 es capaz de cargar una conexión desde el dispositivo de origen 3400.

En una implementación descrita, la aplicación 316 es el destino del paquete conexión de inicio recibido en el dispositivo de origen 3400. Desde el dispositivo de origen 3400, el dispositivo objetivo 3500 recibe el blob binario 3422. El blob binario 3422 incluye el estado de conexión asociado con la conexión que está siendo migrada hacia el dispositivo objetivo 3500 y opcionalmente un identificador de flujo. Este estado de conexión incluye

el estado de protocolo 3420 y el dato de reconocimiento 3416 (posiblemente otra información de conexión o relacionada con la conexión).

En operación, cuando el blob binario 3422 alcanza la capa de interfaz protocolo-hardware 3406, el controlador intermedio del migrador 3414 lo reconoce como un blob para la migración de la conexión y lo desvía. El estado conexión es inyectado en 3502 para crear la apariencia a la solicitud 316 de que la conexión fue terminada originalmente en el dispositivo objetivo 3500.

Específicamente, el estado de protocolo 3420 del estado de conexión inyectado 3502 es infundido dentro de la pila de protocolo 3404. En una implementación descrita el estado de protocolo 3420 es infundido primero en los protocolos de más alto nivel y luego en los protocolos de más bajo nivel de la pila de protocolos 3404. Después de que el estado del protocolo 3420 es infundido dentro de la pila de protocolos 3404, el dato 3416 puede estar indicado en la aplicación 316. Este dato 3416 puede ser provisto a la aplicación 316 como si fuera parte de una conexión nuevamente y localmente termina.

Después de que la inyección del estado de conexión 3502 se ha completado, la conexión iniciada por el paquete recibido en el dispositivo de origen 3400 es migrada exitosamente desde allí hacia el dispositivo objetivo 3500. Los paquetes subsecuentes para la conexión pueden ser dirigidos directamente hacia el dispositivo objetivo 3500 sin pasar por el dispositivo de origen 3400, o por lo menos con solamente un enrutamiento simple y sin ningún análisis a nivel de aplicación que se le aplique. Opcionalmente, estos paquetes pueden ser canalizados de modo que el controlador intermedio del migrador 3414 opera efectivamente como un NIC virtual con base en software que no tiene enlace con la dirección virtual IP.

La Figura 36 ilustra una enseñanza ejemplar de un procedimiento de descarga 3600 para una migración de conexión. El procedimiento de descarga de la migración 3600 ilustra detalles ejemplares adicionales para una migración de conexión por un dispositivo de origen 3400. Como se ilustró, la pila de protocolo general 3404 incluye una pila TCP 3404 (T), una pila IP 3404 (I), y una pila de protocolo de resolución de dirección (ARP) 3404(A). Sin embargo, otras pilas de protocolo específicas 3404() pueden ser empleadas alternativamente.

Como vía de ejemplo, la capa de interfaz protocolo-hardware 3406 puede ser realizada como una capa con base en la especificación de interfaz de controlador de red (NDIS) en un ambiente de sistema operativo (OS) Microsoft® Windows®. También, la capa de conexión 3402 puede realizarse en una capa Winsock™ en un ambiente OS Microsoft® Windows®.

En una implementación descrita, el controlador intermedio del migrador 3414 incluye las interfaces protocolo-hardware 3406 en las uniones a la pila ARP 3404(A) y al minipuerto PNI 3408. El controlador intermedio del migrador 3414 sirve como un objetivo de descarga en el procedimiento de descarga de la migración 3600. El objetivo de descarga es una interfaz protocolo-hardware 3406 en el minipuerto como se ilustró en este ejemplo. En un procedimiento de carga de migración 3700 (como en la Figura 37) el controlador intermedio de migrador 3414 sirve como un desviador de carga.

Más específicamente, el controlador intermedio de migrador 3414 está ligado a cada una de las interfaces de red física 3410 por medio de las cuales una conexión TCP

puede ser migrada. El controlador intermedio de migrador 3414 opera usualmente como un controlador de paso para pasar los paquetes hacia arriba o hacia abajo en la pila red sin interactuar de otra manera con los paquetes. Sin embargo, el controlador intermedio de migrador 3414 si interactúa con los paquetes relacionados con la migración de la conexión (opcionalmente incluyendo paquetes que subsecuentemente son canalizados).

Las responsabilidades del controlador intermedio de migrador 3414 incluye: (i) la aceptación de la solicitudes de descarga de la migración; (ii) la agregación de la información del estado de protocolo que está relacionada con la conexión TCP que está siendo migrada según se compiló desde las pilas de protocolo específicas 3404(), junto con los datos de reconocimiento para producir la información de estado de la conexión; y (iii) la transmisión del estado de conexión agregada a un dispositivo objetivo 3500 para un procedimiento de carga de la migración 3700. Un protocolo alambrado confiable para tal transmisión puede ser compartido con aquel usado por los componentes de rastreo de sesión 2002 y 2010 para enviar y recibir mensajes de información de la sesión

1208 (por ejemplo, como se describió anteriormente con referencia a la Figura 20).

Otra responsabilidad del controlador intermedio de migrador 3414 (por ejemplo, en un procedimiento de carga de la migración 3700) es el de iniciar la carga de las conexiones migradas que son recibidas desde otros dispositivos y amortiguar cualquier paquetes que ingresan relacionados con la conexión de que migra mientras está en el proceso de ser cargada. Para cargar la conexión, el controlador intermedio del migrador 3414 envía una solicitud de carga a 1 ajustador de migrador 3412. El ajustador de migrador 3412 emite una llamada de inyección hacia dentro de la pila de protocolo 3404 en la pila TCP 3404(A) para darle el momento a la conexión en la porción de la pila de protocolo 3404 en la pila de red.

El ajustador de migrador 3412 expone una interfaz de cliente a la pila TCP 3404(T) y expone una interfaz de suministrador o de proveedor a la capa de conexiones 3402. El ajustador de migrador 3412 tiene dos papeles: (i) iniciar el procedimiento de descarga de migración de la conexión 3600 en un dispositivo de origen 3400 y subsecuentemente el procedimiento de carga de la

migración 3700 en un dispositivo objetivo 3500 y (ii) mediar en el proceso de clasificación entre un programa de aplicación de anfitrión 316, un programa de clasificador de balance de carga 304, y una capa de conexión 3402. El ajustador de migrador 3412 y el controlador intermedio de migrador 3414 están ambos descritos más abajo con referencia a las Figuras 36 y 37.

Par un procedimiento de descarga de migración ilustrativo 3600, la migración de una conexión TCP se lleva a cabo después de que el clasificador 304 clasifica la conexión TCP que ingresa usando uno, dos, o más paquetes de la misma. El procedimiento de descarga de la migración 3600 está descrito en los puntos <1> a <7>.

En <1>, una inicialización es llevada a cabo antes de las operaciones de clasificación. La pila de protocolo 3400 hace solicitudes a la capa de interfaz protocolo-hardware 3406 para determinar que capacidades de descarga, si las hay, están disponibles. El controlador intermedio de migrador 3414 indica que la descarga de la migración de la conexión está disponible y propaga la solicitud hacia el minipuerto PNI 3408. Si una habilidad de descarga de tipo chimenea TCP está provista por una interfaz de red

física 3410, el minipuerto PNI 3408 también se lo indica. La descarga tipo chimenea TCP permite algún procesamiento TCP/IP para ser descargado al hardware de la interfaz de red física 3410 e involucra alguna compilación del estado de protocolo 3420. Consecuentemente, alguna compilación y lógica de agregación puede ser compartida entre los dos mecanismos de descarga.

En <2>, una vez que una conexión TCP ha sido clasificada, el clasificador 304 inicia una migración de la conexión TCP a un anfitrión seleccionado 108. Específicamente, un comando de migración que indica un dispositivo objetivado 3500 es emitida por medio de la capa de conexiones 3402 al ajustador del migrador 3412.

En <3>, el ajustador de migrador 3412 inicia la migración de la conexión TCP para compilar el estado de protocolo TCP. Específicamente, el ajustador del migrador 3412 invoca un API de descarga de migrado iniciado en el TCP o más generalmente una llamada de función de conexión de migrado o comando de conexión de migrado). Esta rutina compila el estado relevante para la conexión TCP especificada que es usada para restablecer la conexión sobre el dispositivo objetivado 3500. El estado de

protocolo compilado 3420 incluye el estado de las capas de pila intermedia, incluyendo la pila TCP 3404(T), la pila IP 3404(I), y la pila ARP 3404(A).

En <4>, una vez que la pila de protocolos 3404 ha sido compilada en el estado de protocolo 3420 para la conexión TCP que está siendo migrada, esta invoca una API de descarga de migrado iniciado en el minipuerto al cual está ligado; en este ejemplo, este minipuerto es el controlador intermedio del migrador 3414. Sin embargo, en la práctica, puede existir otros controladores intermedios insertados entre la pila de protocolo 3404 y el controlador intermedio del migrador 3414 tal como QoS del IP. se es así, aquellos controladores IM pueden participar en la migración, si es relevante, compilando/agregando su estado a la información de estado para la conexión que está siendo migrada. Los controladores intermedios continúan propagando la llamada de descarga del migrado iniciado hacia abajo a la pila de red, que eventualmente resulta en la ejecución de un manejador de descarga de migrado en el controlador intermedio del migrador 3414. En este punto, el controlador intermedio de migrador 3414 también agrega cualesquier datos reconocidos con el estado de conexión

restante para transferir la conexión TCP al dispositivo objetivado 3500.

En <5>, después de almacenar/copiar la información de estado de la conexión para la conexión TCP que está siendo migrada, el controlador intermedio del migrador 3414 notifica a la pila de red que la migración está en sus etapas finales invocando un API de descarga completa del migrado inicial. Este API completo de descarga de migrado iniciado sigue el trayecto inverso hasta la pila de red, por medio de los mismos controladores intermedios (si había alguno), y eventualmente la pila de protocolo 3404. A medida que cada capa procesa esta llamada, la información de estado que está asociada con la conexión migrada puede liberarse. Hasta que el procesamiento de la llamada esté completo, cada capa puede enviar notificaciones de actualización hacia abajo a la pila red para actualizar cualquier parte del estado de la conexión que haya cambiado desde que la migración fue iniciada.

En <6>, cuando la rutina completa de descarga de migrado iniciada alcanza la pila TCP 3404(T), el TCP silenciosamente (es decir, que no hay un reposicionamiento enviado al cliente 108) cierra la

conexión, lavando todos los estados asociados con la conexión migrada, y propaga la llamada de descarga completa del migrado iniciado al ajustador del migrador 3412. En este punto, la pila de red está libre de cualquier conocimiento residual de la conexión TCP migrada.

En <7>, cuando la llamada completa de descarga de migrado iniciado regresa al controlador intermedio del migrador 3414 (por medio de la porción del ajustador del migrador 3412 del migrador de conexión 310) la migración de la conexión TCP desde el dispositivo de origen 3400 al dispositivo objetivo 3500 puede comenzar con la transferencia del estado de conexión a este mismo. El estado de conexión puede ser transferido asincrónicamente y confiablemente.

Una vez que la migración es iniciada el dispositivo de origen 3400 es también responsable de asegurar que los datos subsecuentes desde el cliente 108 sea dirigidos o enviados al dispositivo objetivo 3500. Consecuentemente, aún después de que la conexión ha migrado exitosamente al objetivo, el originador retiene alguna cantidad de estado para la conexión (por ejemplo, una entrada de tabla de

enrutamiento) para poder enrutar de manera apropiada paquetes subsecuentes hacia el objetivo. Cuando la conexión se ha terminado, el objetivo notifica al originador que permita purgar cualquier estado residual remanente o que permanezca para la conexión migrada más aún, como una consecuencia de la naturaleza asíncrona de la migración de conexión, los paquetes de datos para la conexión en migración que son dirigidos o enviados por el dispositivo de origen 3404 (o por un enviador designado si es un dispositivo separado) puede iniciar la llegada al dispositivo objetivado 3500 antes de que el dispositivo objetivo 3500 reciba el estado de conexión migrada. El controlador intermedio del migrador 3414 en el dispositivo objetivado 3500 es responsable de amortiguar aquellos paquetes hasta que la conexión migrada asociada se haya establecido en el dispositivo objetivo 3500.

La Figura 37 ilustra una enseñanza ejemplar de un procedimiento de carga 3700 para una migración de conexión. El procedimiento de carga de la migración 3700 ilustra detalles ilustrativos adicionales para una migración de conexión por el dispositivo objetivo 3500.

Cuando una conexión migrada llega a un dispositivo objetivo 3500, esta descansa en el controlador intermedio del migrador 3414 para su procesamiento. Después de amalgamar y asimilar el estado de conexión migrado, el controlador intermedio de migrador 3414 en conjunto con el ajustador del migrador 3412, inyecta la conexión migrada en la pila de red local de una manera transparente a la aplicación 316. Para un procedimiento de carga de migración ilustrativo 3700 la migración de una conexión TCP en los puntos <1> a <8> se ha descrito.

En <1>, como se describió anteriormente con referencia al procedimiento de descarga de la migración 3600, una inicialización es llevada a cabo en las operaciones de anfitrión de la aplicación. Específicamente, la pila de protocolo 3404 hace solicitudes con respecto a que capacidades de descarga, si existen, están disponibles. El controlador intermedio del migrador 3414 llena en la solicitud de soporte de migración de la conexión TCP para indicar que la carga de la migración de la conexión está disponible y también propaga una solicitud hacia el minipuerto PNI 3408 para averiguar sobre posibles capacidades de descarga tipo chimenea TCP.

En <2>, cuando los datos de migración de la conexión llegan al dispositivo objetivo 3500, la información de la migración de la conexión (por ejemplo, un blob binario 3422) es suministrado al controlador intermedio del migrador 3414. El controlador intermedio del migrador 3414 reensambla el estado de la conexión, la ajusta con cualquier dato asociado que haya llegado durante la migración y prepara la carga en la pila de red. Cualquier dato del cliente 102 que llegue durante el proceso de carga de la conexión migrada es amortiguado por el controlador intermedio de migrador 3414. Luego de que se completa exitosamente la migración, el dato será suministrado a la aplicación 316.

En <3>, para iniciar la carga de la conexión migrada en la pila de red local, el controlador intermedio de migrador 3414 notifica al ajustador de migrador 3412 que una solicitud de conexión migrada ha llegado. El controlador intermedio de migrador 3414 también suministra el estado de conexión (o por lo menos el estado de protocolo 3420) al ajustador de migrador 3412.

En <4>, el ajustador de migrador 3412 inicia la carga de la conexión migrada invocando una rutina de inyección

iniciada TCP (o más generalmente una rutina de infusión de estado de protocolo) y por medio de suministrar el estado de protocolo migrado 3420 a la pila TCP 3404(T). En <5>, el TCP/IP recrea la conexión migrada por toda la pila de protocolo 3404 usando el estado de protocolo suministrado 3420. Este estado de protocolo 3420 puede incluir una o más d l estado de transporte (TCP), estado de trayecto (IP), estado de vecino y siguiente salto (ARP), etcétera.

En <6>, si la conexión migrada es reestablecida exitosamente en el dispositivo objetivo 3500, el TCP inicia un evento de conexión a una porción de cliente del ajustador de migrador 3412 para indicar que una nueva conexión ha sido establecida. Existe una multitud de razones posibles para falla, pero las razones comunes pueden incluir la carencia de un escuchador correspondiente, falla de rutina, etcétera. En estos casos en donde la pila de red no está disponible para restablecer la conexión migrada, ningún evento de conexión es indicado y un estado de falla es especificado en la llamada completa de inyección iniciada. El migrador de conexión 310 es responsable de limpiar la migración y de enviar una notificación de restablecimiento

reposicionamiento de nuevo al cliente 102 para abandonar la conexión.

En <7>, el ajustador del migrador 3412 actúa como un proveedor para propagar el evento de conexión a la capa de conexiones 3402 de modo que se indique a la aplicación que escucha 316 que una nueva conexión ha sido establecida. Si la aplicación 316 acepta la conexión, esta procesa la solicitud y responde por medio de operaciones de conexión de lectura y escritura normales; la aplicación 316 puede estar inadvertida o no tener conocimiento de que la conexión fue migrada. Si la conexión no es aceptada por la aplicación 316, el TCP termina la conexión pero no envía una notificación de restablecimiento de nuevo al cliente 102. Nuevamente, un estado de fallas especificado en la llamada completa de inyección iniciada, y el migrador de conexión 310 es responsable de limpiar la migración y de enviar una notificación de restablecimiento de nuevo al cliente 102 para abandonar la conexión.

Una situación especial surge cuando la aplicación 316 y el clasificador 304 están localizados en el mismo dispositivo: el ajustador del migrador 3412 puede hacer

referencia entre ellos. Cuando ambas clases de programa residen en el mismo anfitrión 108, ellas pueden ambas ser escuchadoras de las mismas direcciones IP y puertos IP. Sin embargo, el TCP típicamente tiene un escuchador por dirección IP y puerto IP único. Consecuentemente, el ajustador del migrador 3412 puede oscurecer una configuración en donde dos programas están escuchando sobre la misma dirección y puerto IP por medio de multiplexar las dos conexiones en un escuchador único en la capa TCP.

En tal caso, cuando los eventos de conexión llegan a la porción del cliente del ajustador del migrador 3412, el ajustador de migrador 3412 como un proveedor determina sobre que conexión de escucha se suministra la notificación de conexión a la capa de conexiones 3402. Si solamente hay una conexión escuchando en la dirección y puerto IP correspondiente, entonces esa conexión recibe el evento de conexión. Si existe más de una conexión escuchando, entonces el receptor depende del contexto en el cual el evento de conexión está indicado. Si el evento de conexión es una conexión totalmente nueva para una dirección IP virtual, entonces el evento de conexión es suministrado al clasificador 304; si el evento de

conexión es para una dirección IP dedicada (dirección IP sin balance de carga) o el resultado de cargar una conexión migrada, entonces el evento de conexión es suministrado a la aplicación objetivo 316.

En <8>, una vez que la inyección de la conexión migrada está completa, el TCP notifica al ajustador de migrador 3412 invocando al manejador completo de inyección iniciado suministrado. Un código de estado es suministrado al ajustador de migrador 3412 de si o no la conexión fue cargada exitosamente. Si la carga de la conexión migrada falla, el migrador de conexión 310 es responsable de limpiar la migración y de notificar al cliente 102 que la conexión ha sido abandonada enviándole un restablecimiento. Si la conexión migrada fue inyectada exitosamente dentro de la pila de red local, el controlador intermedio del migrador 3414 puede iniciar el suministro de cualquier dato amortiguado o detenido del cliente 102 haciendo pasar el paquete recibido a través del trayecto de recepción de paquete de la interfaz protocolo-hardware 3406.

Cuando una conexión migrada es terminada (debido a que falló la carga, debido a que la conexión migrada fue

cerrada subsecuentemente por medios normales, etc.), el dispositivo objetivo 3500 notifica al dispositivo de origen 3400. El dispositivo de origen 3400 usa estas notificaciones para una limpieza más eficiente y confiable del estado de las conexiones migradas, incluyendo las entradas en la tabla de enrutamiento. Por lo tanto, para tener en cuenta las conexiones migradas exitosamente que terminan arbitrariamente en el futuro, el ajustador de migrador 3412 puede monitorear su actividad y notificar al controlador intermedio de migrador 3414 cuando las conexiones para ella fueron cerradas.

La Figura 38 ilustra una enseñanza ejemplar de canalización de paquetes entre un enviador 302 y un anfitrión 108. Los paquetes encapsulados 3808 pueden ser canalizados desde el enviador 302 al anfitrión 108 sin incurrir en un sobrecabezamiento para cada paquete transmitido. Como se describe más adelante, la canalización es efectuada usando un identificador de flujo 3814 y tablas de mapeo de encapsulación 3806 y 3810 de los canalizadores 312(F) y 312(H), respectivamente, del enviador 302 y del anfitrión 108, respectivamente. El

~~312~~

identificador de flujo 3814 es insertado dentro de los paquetes encapsulados 3808.

Como se anotó anteriormente con referencia a la Figura 32, los paquetes para una conexión que llegan subsecuentemente de una migración de conexión pueden ser enrutados por el enviador 302 hacia el anfitrión 108(1) usando la canalización por medio de un canalizador 312. En (8) (de la Figura 32), el enviador 302 envía tales paquetes subsecuentes desde el enviador 302 teniendo una dirección de red de "F" al anfitrión 108(1) que tiene una dirección de red de "H1". Como se describió anteriormente con referencia a la Figura 4, el enviador 302 puede llevar a cabo canalización NAT, media NAT, etc. para poder enrutar los paquetes que ingresan al anfitrión 108(1).

Tales paquetes de entrada incluyen una dirección IP de destino de la dirección virtual IP ("VIP") y una dirección de fuente IP del "C1" para los paquetes que llegan del cliente 102(1). Los paquetes son enrutados hacia el anfitrión 108(1) tienen una dirección de destino IP de H1 y una dirección de fuente de C1 (para media NAT) o "F" (para NAT completa). Esta reescritura de las

direcciones puede interferir con algunos protocolos que esperan tanto que el cliente 102(1) y el anfitrión 108(1) tengan vistas idénticas de las direcciones fuente y destino.

Más aún, por lo menos con respecto a la NAT completa, los trayectos de retorno desde el anfitrión 108(1) al cliente 102(1) que no corran por el envidador 302 son prohibitivas por que el anfitrión 108(1) no conoce la dirección del cliente 102(1). Los trayectos directos del anfitrión 108(1) al cliente 102(1) son deseables en situaciones en las cuales el tráfico desde el anfitrión 108(1) al cliente 102(1) es especialmente alto y/o significativamente mayor que el tráfico en la dirección opuesta (por ejemplo cuando el anfitrión 108(1) suministra medios de corrientes a los clientes 102(1)).

La canalización por los canalizadores 312 como se describe aquí puede suministrar vistas idénticas con respecto a las direcciones fuente y destino (y los puerto) para los clientes 102 y las aplicaciones 316 en los anfitriones 108. Por vía de ejemplo con referencia a las Figuras 34 y 35, el canalizador 312 en cada uno de los envidadores 302 y anfitriones 108 puede operar como

parte de o en conjunto con un controlador intermedio de migrador 3414 de un migrador de conexión 310.

En una implementación descrita para la Figura 38, el migrador de conexión 310 suministra un mapeo de encapsulación 3812 entre un identificador de flujo 3814 y 4 tuplas TCP/IP 3804. El migrador de conexión 310 puede estar asociado con un clasificador 304, y el migrador de conexión 310 (opcionalmente junto con tal un clasificador 304) puede estar localizado sobre un mismo dispositivo que el emisor 302. Alternativamente, el migrador de conexión 310 (lo mismo que el clasificador 304) pueden estar localizados sobre dispositivos diferentes del emisor 302. El mapeo de encapsulación 3812 puede alternativamente ser suministrado en conjunto con la funcionalidad de canalizador 312 que está, por ejemplo localizada en y/o asociada con un clasificador 304.

Al ser mapeada a 4 tuplas TCP/IP 3804 en el mapeo de encapsulación 3812, el identificador de flujo 3814 sirve para identificar un flujo de paquetes encapsulados 3808 para una conexión particular. Las 4 tuplas TCP/IP 3804 incluye las direcciones de red (y puertos, etc.) para la fuente y destino de una conexión particular de acuerdo

con un protocolo TCP/IP, o cualquier protocolo similar o análogo. El identificador de flujo 3814 es 32 bits en una implementación descrita por que 32 bits está disponible para conexiones establecidas de acuerdo con un protocolo de Internet IPv4. Sin embargo, los identificadores de flujo 3814 de otras longitudes pueden ser usados alternativamente, especialmente para otros protocolos tales como los Internet IPv6, UDP, etc.

Los identificadores de flujo 3814 pueden ser generados usando cualquier mecanismo apropiado, tal como un contador de conexión incremental. Más aún, la 4 tuplas TCP/IP 3804 es más generalmente un par fuente/destino. Cada valor de fuente y valor de destino de un par fuente/destino individual puede incluir un identificador de nodo de red (por ejemplo, dirección de red, puerto, alguna combinación de ellos, etc.) para la fuente y el destino, respectivamente de un paquete dado que se propaga sobre una conexión particular.

El migrador de conexión 310 suministra el mapeo de encapsulación 3812 al anfitrión 108. El canalizador 312(H) en el anfitrión 108 almacena el mapeo de encapsulación 3812 en la tabla de mapeo de encapsulación

3810 como una entrada de mapeo de encapsulación 3810(1). El canalizador 312(H) puede de allí en adelante usar el identificador de flujo 3814 para mapear e identificar la conexión particular correspondiente a las 4 tuplas TCP/IP 3804. el mapeo de encapsulación 3812 puede opcionalmente ser suministrado al anfitrión 108 como parte de un blob binario amarrado 3422 en una operación de migración de conexión.

El enviador 302 también incluye un componente de canalizador 312(F) con una tabla de mapeo de encapsulación 3806. La tabla de mapeo de encapsulación 3806 almacena una entrada de mapeo de encapsulación 3806(1) que enlaza/mapea las cuatro tuplas TCP/IP 3804 para una conexión particular a un identificador de flujo 3814. El canalizador 312(F) también recibe la información de mapeo para la entrada de mapeo de encapsulación 3806(1) del migrador de conexión 310 (por ejemplo, como un mapeo de encapsulación 3812).

Aunque solamente una entrada de mapeo de encapsulación 3806(1) y 3810(1) se muestran, cada una de las tablas de mapeo de encapsulación 3806 y tabla de mapeo de encapsulación 3810 puede tener múltiples de tales

entradas. Estas tablas de mapeo de encapsulación 3806 y 3810 pueden estar combinadas con otra información, tales como tablas para información de sesión del rastreador de sesión 308.

Cuando un dispositivo de transmisión (tal como el emisor 302) y un dispositivo de recepción (tal como el anfitrión 108) de paquetes encapsulados 3808 solamente canalizados entre ellos, las tablas de mapeo de encapsulación de los mismos similarmente tienen las mismas entradas de mapeo de encapsulación. De otra manera, la tabla de mapeo de encapsulación 3806 y la tabla de mapeo de encapsulación 3810 similarmente tienen un conjunto total diferente de entradas de mapeo de encapsulación 3806() y entradas de mapeo de encapsulación 3810() respectivamente.

En operación, un paquete que entra 3802 para una conexión particular es recibido en el emisor 302. La conexión particular está asociada con las 4 tuplas TCP/IP 3804. El paquete que entra 3802 incluye las 4 tuplas TCP/IP 3804 con una dirección fuente IP (de un cliente 102), una dirección destino IP (IP virtual) un puerto fuente TCP (del cliente 102), y un puerto destino TCP.

El canalizador 312(F) acepta el paquete que ingresa 3802 para la canalización hacia el anfitrión 108. Usando las 4 tuplas TCP/IP 3804, el canalizador 312(F) accesa la tabla de mapeo de encapsulación 3806 para localizar la entrada de mapeo de encapsulación 3806(1). El identificador de flujo 3814 es extraído de la entrada de mapeo de encapsulación 3806(1) al ser enlazada, mapiado a las 4 tuplas TCP/IP 3804.

Para crear el paquete encapsulado 3808, el canalizador 312(F) inserta el identificador de flujo 3814 en las porciones de puerto de fuente y destino del encabezador de las 4 tuplas TCP/IP. Para una implementación Internet IPv4, estas dos porciones de puerto TCP ofrecen 32 bits de espacio total. También, para la porción de dirección fuente IP del encabezador de las 4 tuplas TCP/IP, el canalizador 312(F) inserta la dirección IP "F" del enviador 302. Para la porción de dirección de destino IP del encabezador de las 4 tuplas TCP/IP, el canalizador 312(F) inserta la dirección IP "H" del anfitrión 108.

El enviador 302 enruta/transmite el paquete encapsulado 3808 hacia el anfitrión 108, y el anfitrión 108 recibe el

paquete encapsulado 3808 del enviador 302. El componente del canalizador 312(H) en el anfitrión 108 detecta que el paquete encapsulado 3808 es un paquete canalizado que debe ser desencapsulado.

El identificador de flujo 3814 es extraído del paquete encapsulado 3808 y usado para buscar las 4 tuplas TCP/IP correspondientes 3804 que están ligadas a el en la entrada de mapeo de encapsulación 3810(1) de la tabla de mapeo de encapsulación 3810. Las 4 tuplas TCP/IP 3804 se usan por el canalizador 312(H) para recrear el encabezador de las 4 tuplas TCP/IP 3804 como fueron recibidas originalmente en el paquete de entrada 3802 en el enviador 302.

Específicamente, la dirección IP F del enviador 302 es remplazada con la dirección fuente IP y la dirección IP H del anfitrión 108 es remplazada con la dirección destino IP. Más aún, el identificador de flujo 3814 es remplazado por el puerto fuente TCP y el puerto destino TCP. El paquete desencapsulado es entonces indicado a la pila de red del anfitrión 108 hacia la aplicación objetivo 316.

Más generalmente, una porción del encabezador del paquete, incluyendo una porción de un par fuente/destino para un paquete dado que no sea necesariamente usado para comunicación del paquete dado para llevar un identificador de flujo 3814. Por presuministrar por lo menos parte del par fuente/destino al anfitrión 108, un identificador de flujo 3814 puede ser empleado para canalizar (por ejemplo encapsular y/o desencapsular) paquetes sin incurrir en una sobre escrituración de encapsulación en cada paquete. Más aún, los paquetes que son de tamaño completo con respecto a un protocolo dado pueden ser canalizados sin ser rotos o divididos.

La Figura 39 es un diagrama de flujo 3900 que ilustra un método ejemplar para el canalizado de paquetes entre un primer dispositivo y un segundo dispositivo. Por ejemplo, el primer dispositivo y el segundo dispositivo pueden corresponder a un dispositivo de origen 3400 y un dispositivo objetivo 3500, respectivamente, de una infraestructura de balance de carga 106 y un grupo de anfitriones 108 respectivamente. Sin embargo, la canalización puede ser empleada en implementaciones sin balance de carga.

El diagrama de flujo 3900 incluye 12 bloques 3902 a 3924. Aunque las acciones del diagrama de flujo 3900 pueden llevarse a cabo en otros ambientes y con una variedad de esquema de software, las Figuras 1 a 3, 32, 34, 35 y 38 son usadas en particular para ilustrar ciertos aspectos y ejemplos de método.

En el bloque 3902, un mapeo de un identificador de flujo a las 4 tuplas TCP/IP es enviado a un dispositivo objetivo desde un dispositivo de origen. Por ejemplo, el dispositivo de origen 3400 puede enviar un mapeo de encapsulación 3812 que enlaza un identificador de flujo 3814 a 4 tuplas TCP/IP 3804. En el bloque 3914 el mapeo del identificador de flujo a las 4 tuplas TCP/IP es recibido en el dispositivo objetivo desde el dispositivo de origen. Por ejemplo el dispositivo objetivo 3500 recibe el mapeo de encapsulación 3812 que enlaza el identificador de flujo 3814 a las 4 tuplas TCP/IP 3804 desde dispositivo de origen 3400.

Alternativamente, el dispositivo objetivo 3500 puede recibir el mapeo de encapsulación 3812 desde otro dispositivo. Como se indicó por las flechas en línea punteada 3926 y 3928, las acciones de los bloques 3904 a

3912 y los bloques 3916 a 3924 pueden ocurrir en algún tiempo después de las acciones de los bloques 3902 y 3914, respectivamente.

En el bloque 3904, un paquete que ingresa es recibido en el dispositivo de origen desde un cliente. Por ejemplo, un paquete que ingresa 3802 tiene un encabezador con 4 tuplas TCP/IP 3804 que puede ser recibido en el dispositivo de origen 3400 desde un cliente 102. En el bloque 3906, un identificador de flujo está siendo buscado para una conexión correspondiente al paquete del cliente usando las 4 tuplas TCP/IP del paquete que ingresa. Por ejemplo, el identificador de flujo 3814 puede ser buscado para la conexión con el cliente 102 usando las 4 tuplas TCP/IP 3804 que son mapeadas allí en una entrada de mapeo de encapsulación 3806(1) de una tabla de mapeo de encapsulación 3806.

En el bloque 3908 la fuente IP y el destino IP del paquete de entrada son reemplazados con una dirección de origen IP del dispositivo de origen y una dirección objetivo IP del dispositivo objetivo, respectivamente. Por ejemplo, el dispositivo de origen 3400 puede reemplazar las porciones de dirección IP de la porción de

las 4 tuplas TCP/IP 3804 de un encabezador del paquete que ingresa 3802 con las direcciones IP del dispositivo de origen 3400 y del dispositivo de objetivo 3500.

En el bloque 3910, el puerto de fuente y el puerto de destino del paquete que ingresa son remplazados con el identificador de flujo. Por ejemplo, el dispositivo de origen 3400 puede remplazar los puertos fuente y destino TCP de la porción de las 4 tuplas TCP/IP 3804 del encabezador del paquete que ingresa 3802 con el identificador del flujo 3814. En el bloque 3912 el paquete encapsulado es enviado desde el dispositivo de origen hacia el dispositivo objetivo. Por ejemplo, el dispositivo de origen 3400 puede enviar un paquete encapsulado 3808 al dispositivo objetivo 3500.

En el bloque 3916, el paquete encapsulado es recibido en el dispositivo objetivo desde el dispositivo de origen. Por ejemplo, el dispositivo objetivo 3500 puede recibir el paquete encapsulado 3808 desde el dispositivo de origen 3400. En el bloque 3918, las 4 tuplas TCP/IP son buscadas para la conexión correspondiente al paquete recibido desde el cliente usando el identificador de flujo. Por ejemplo, el dispositivo objetivo 3500 puede

accesar una tabla de mapeo de encapsulación 3810 en una entrada de mapeo de encapsulación 3810(1) que mapea al identificador de flujo 3814 para las 4 tuplas TCP/IP 3804.

El bloque 3920, la dirección IP de origen y la dirección objetivo IP son remplazadas con la dirección fuente IP y la dirección destino IP respectivamente, usando las 4 tuplas TCP/IP buscadas. Por ejemplo, el dispositivo objetivo 3500 puede remplazar las direcciones IP del dispositivo de origen 3400 y el dispositivo objetivo 3500 en el paquete encapsulado 3808 con la dirección fuente IP y la dirección destino IP desde las 4 tuplas TCP/IP 3804 de acuerdo con lo logrado de la tabla de mapeo de encapsulación 3810.

En el bloque 3922 el identificador de flujo es remplazado con el puerto fuente y el puerto destino del paquete que ingresa usando las 4 tuplas TCP/IP buscadas. Por ejemplo, el dispositivo objetivo 3500 puede remplazar el identificador de flujo 3814 en el paquete encapsulado 3808 con el puerto fuente TCP y el puerto destino TCP de la 4 tuplas TCP/IP 3804. En el bloque 3924, el paquete del cliente se le ha indicado hacia una aplicación en el

dispositivo objetivo. Por ejemplo, una versión desencapsulada del paquete encapsulado 3808, o el paquete que ingresa 3802, se ha indicado a la aplicación 316 del dispositivo objetivo 3500.

Las acciones aspecto, rasgos, componentes, etc. de las Figuras 1 a 39 son ilustradas en diagramas que están divididos en bloques múltiples. Sin embargo, el orden, las interconexiones, el diseño, etc. en las cuales las Figuras 1 a 39 se han descrito y/o mostrado no tiene el propósito de construir una limitación, y cualquier número de bloques que puedan ser combinados, redispuestos, aumentados, omitidos, etc. en cualquier manera para implementar uno o más sistemas, métodos, dispositivos, procedimientos, medios, API, aparatos, disposiciones, etc. para el balance de red de carga. Más aún, aunque la descripción aquí incluye referencias a implementaciones específicas (y el ambiente operativo ilustrativo de la Figura 40), las implementaciones ilustradas y/o descritas pueden ser implementadas en cualquier hardware, software, firmware, o combinaciones adecuadas de ellos y usando cualesquiera organizaciones de red adecuadas protocolos de transporte/comunicación, interfaces de programación de aplicaciones (API), arquitecturas cliente-servidor, etc.

Ambiente Operativo Ilustrativo para Computador u otro
Dispositivo

La Figura 40 ilustra un ambiente operativo de cómputo ilustrativo (o un dispositivo general) 4000 que es capaz de (completa o parcialmente) implementar por lo menos un sistema, dispositivo, aparato, componente, disposición, protocolo, enseñanza, método, procedimiento, media, API, alguna combinación de ellos, etc. para el balance de carga de red como se describe aquí. El ambiente operativo 4000 puede ser utilizado en el computador en arquitecturas de computación y de red descritas más adelante o en situaciones de un aparato único.

El ambiente operativo ilustrativo 4000 es solamente un ejemplo de un ambiente y no tiene el propósito de sugerir ninguna limitación en cuanto al alcance de uso o funcionalidad del dispositivo aplicable (incluyendo computador, nodo de red, dispositivo de entretenimiento, aplicación móvil, dispositivo electrónico general, etc.). Tampoco debería el ambiente operativo 4000 (o los dispositivos de el) ser interpretados como que tienen alguna dependencia o requerimiento relacionado con uno

cualquiera o cualquier combinación de los componentes como se ilustra en la Figura 40.

Adicionalmente, el balanceo de carga de red puede ser implementado con numerosos otros propósitos o dispositivos de propósito especial (sistema de cómputo incluido) ambientes o configuraciones. Ejemplos de dispositivos, sistemas, ambientes, y/o configuraciones bien conocidas que pueden ser adecuadas para el uso incluyen, pero no se limitan a, computadores personales, computadores servidores, clientes delgados, clientes grandes, asistentes digitales personales (PDA) o teléfonos móviles, relojes, dispositivos manuales o portátiles, sistemas multiprocesador, sistemas con base en microprocesador, cajas de TV clave, electrónicos de consumidor programables, máquinas de video juegos, consolas de juego, unidades de juego portátiles o manuales, PC de red, minicomputadores computadores de tamaño medio, nodos de red, ambientes de cómputo distribuidos sobre multiprocesamiento que incluyen cualquiera de los anteriores sistemas o dispositivos, y algunas combinaciones de ellos etc.

~~308~~

Las implementaciones para el balanceo de carga de red pueden ser descritas en el contexto general de instrucciones ejecutables por procesador. Generalmente, las instrucciones ejecutables por procesador incluyen rutinas, programas, protocolos, objetos, interfaces, componentes, estructuras de datos, etc. que llevan a cabo y/o permiten tareas particulares y/o implementan tipos de datos abstractos particulares. El balanceo de carga de red, como se describe en ciertas implementaciones aquí también puede ser practicado en ambientes de procesamiento distribuido en donde las tareas se llevan a cabo por dispositivos de procesamiento enlazados remotamente que están conectados por medio de un enlace de comunicaciones y/o una red. Especialmente en un ambiente de cómputo distribuido, las instrucciones ejecutables por procesador pueden estar localizadas en medios de almacenamiento separado ejecutas por diferentes procesadores, y/o propagadas sobre medios de transmisión.

El ambiente operativo ilustrativo 4000 incluye un dispositivo de cómputo de propósito general en la forma de una computador 4002, que puede comprender cualquiera (por ejemplo electrónico) cualquier dispositivo con capacidades de cómputo/procesamiento. Los componentes del

computador 4002 pueden incluir, pero no limitarse a, uno o más procesadores o unidades de procesamiento 4004, una memoria de sistema 4006, y un bus de sistema 4008 que acopla los varios componentes del sistema incluyendo el procesador 4004 a la memoria del sistema 4006.

Los procesadores 4004 no están limitados por los materiales de los cuales ellos están formados o de los mecanismos de procesamiento empleados allí. Por ejemplo, los procesadores 4004 pueden comprender semiconductores y/o transistores (por ejemplo circuitos integrados electrónicos (IC)). En tal contexto, las instrucciones ejecutables por procesador pueden ser instrucciones ejecutables electrónicamente. Alternativamente, los mecanismos para los procesadores 4004, y por lo tanto de o para los computadores 4002, pueden incluir, pero no están limitados a, cómputo quantum, cómputo óptico, cómputo mecánico (por ejemplo usando la nanotecnología), etc.

El bus del sistema 4004 representa uno o más de cualquiera de los muchos tipos de estructuras de bus alambradas o inalámbricas, incluyendo un bus de memoria o controlador de memoria, y una conexión punto a punto, un

grupo de cables de conmutación, un bus periférico, un puerto de gráficos acelerado, un procesador o bus local que usa cualquiera de una variedad de arquitectura de bus. Por vía de ejemplo, tales arquitecturas pueden incluir una arquitectura de industria estándar (ISA), un bus de arquitectura de microcanal (MCA), un bus de ISA mejorado (EISA), un bus local de asociación estándar de video electrónica (VESA), un bus de interconexión de componentes periféricos (PCI) conocido como un bus mezanine, algunas combinaciones de ellos, etc.

El computador 4002 típicamente incluye una variedad de medios accesibles por procesador. Tales medios pueden ser cualesquiera medios disponibles que sean accesibles por el computador 4002 o por otro dispositivo (por ejemplo electrónico), y éste incluya tanto medios volátiles como no volátiles, medios removibles y no removibles, y medios de almacenamiento y transmisión.

La memoria del sistema 4006 incluye medios de almacenamiento accesibles por procesador en la forma de memoria volátil, tal como la memoria de acceso aleatorio (RAM) 4040, y/o memoria no volátil, tal como la memoria de solo lectura (ROM) 4012. Un sistema básico de

301

entrada/salida (BIOS) 4014, que contienen las rutinas básicas que ayudan a transferir la información entre los elementos dentro del computador 4002, tal como durante el arranque, son típicamente almacenados en la ROM 4012. La RAM 4010 típicamente contiene los datos y/o módulos de programa/instrucciones de programa que son accesibles inmediatamente a y/o están operados actualmente por la unidad procesadora 4004.

El computador 4002 puede también incluir otros medios de almacenamiento removibles/no removibles y/o volátiles/no volátiles. Como vía de ejemplo, la Figura 40 ilustra un positivo de disco duro o arreglo de disco duro 4016 para leer desde y escribir hacia (típicamente) no removible, medio magnético no volátil (no mostrados separadamente); un dispositivo de disco magnético 4018 para leer desde y escribir hacia (típicamente) un disco magnético no volátil, removible 4020 (por ejemplo "un disco flexible"); y un dispositivo de disco óptico 4022 para leer desde y/o escribir hacia (típicamente) un disco óptico removible, no volátil 4024 tal como un CD, DVD u otro medio óptico. El dispositivo de disco duro 4016, el dispositivo de disco magnético 4018, y el dispositivo de disco óptico 4022 están cada uno conectados al bus del

sistema 4008 por uno o más interfaces de medio de almacenamiento 4026. Alternativamente, el dispositivo de disco duro 4016, el dispositivo de disco magnético 4018 y el dispositivo de disco óptico 4022 pueden estar conectados al bus del sistema 4008 por uno o más de otras interfaces separadas o combinadas (no mostrados).

Los dispositivos de disco y sus medios accesibles por procesador asociados suministran almacenamiento no volátil de las instrucciones ejecutables por procesador, tal como estructuras de datos, módulos de programa, y otros datos para el computador 4002. Aunque el computador ilustrativo 4002 ilustra un disco duro 4016, un disco magnético removible 4020, y un disco óptico removible 4024, debe apreciarse que otros tipos de medios accesibles por procesador pueden almacenar instrucciones que son accesibles por un dispositivo, tal como casete magnéticos u otros dispositivos de almacenamiento magnético, memorias flash, disco compactos (CD), discos digitales versátiles (DVD) u otros almacenamientos ópticos, RAM, ROM, memorias programables de solo lectura borrables eléctricamente (EEPROM), etc. Tales medios pueden también incluir los llamados chip IC alambrados para propósitos especiales. En otras palabras, cualquier

medio accesible por procesador puede ser utilizado para realizar los medios de almacenamiento del ambiente operativo ilustrativo 4000.

Cualquier número de módulos de programa (u otras unidades o conjuntos de instrucciones/códigos) pueden ser almacenados en el disco duro 4016, disco magnético 4020, disco óptico 4024, ROM 4012, y/o RAM 4040 incluyendo como vía de ejemplo general, un sistema operativo 4028, uno o más programas de aplicación 4030, otros módulos de programa 4032, y datos de programa 4034.

Un usuario puede ingresar comandos y/o información al computador 4002 por medio de dispositivos de entrada tales como teclado 4036 y un dispositivo de apuntamiento 4038 (por ejemplo un "ratón"). Otros dispositivos de entrada 4040 (no mostrados específicamente) pueden incluir un micrófono, un joystick, una almohadilla de juego un plato satelital, puerto serial, escáner, y/o similares. Estos y otros dispositivos de entrada están conectados a la unidad de procesamiento 4004 por medio de interfaces de entrada/salida 4042 que son acoplados al bus del sistema 4008. Sin embargo, los dispositivos de entrada y/o dispositivos de salida pueden en su lugar

estar conectados por otras interfaces y estructuras de bus, tales como puerto paralelo, un puerto para juegos, un bus serial universal (USB), un puerto infrarrojo, una interfaz IEEE 1394 ("Firewire"), una interfaz IEEE 802.11 inalámbrica, una interfaz inalámbrica Bluetooth®, etc.

Una pantalla o monitor para visualización 4044 u otro tipo de dispositivo de visualización puede ser también conectado al bus del sistema 4008 por medio de una interfaz, tal como un adaptador de video 4046. El adaptador de video 4046 (u otro componente) puede incluir una tarjeta de gráficos para el procesamiento de cálculos intensivos de gráficos y para manejar requerimientos de visualización grandes. Típicamente, una tarjeta de gráficos incluye una unidad de procesamiento de gráficos (GPU), una RAM de video (VRAM), etc. para facilitar la visualización rápida de gráficos y operaciones de comportamiento de los gráficos. En adición al monitor 4044, otros dispositivos periféricos de salida pueden incluir componentes tales como parlantes (no mostrados) y una impresora 4048, que puede estar conectada al computador 4002 por medio de las interfaces entrada/salida 4042.

El computador 4002 puede operar en un ambiente en red usando conexiones lógicas a uno o más computadores remotos, tal como un dispositivo de cómputo remoto 4050. Por vía de ejemplo, el dispositivo de cómputo remoto 4050 puede ser un computador personal, un computador portátil (por ejemplo un computador portátil, un computador de tableta un PDA, una estación móvil, etc.), un computador de palma o de tamaño bolsillo, un reloj, un dispositivo de juego, un servidor, un concentrador, un computador de red un dispositivo par, otro nodo de red, u otro tipo de dispositivo como se listó anteriormente etc. Sin embargo, el dispositivo de cómputo remoto 4050 está ilustrado como un computador portátil que puede incluir muchos de todos los elementos y características descritos aquí con respecto al computador 4002.

Las conexiones lógicas entre el computador 4002 y el computador remoto 4050 son ilustradas en una red de área local (LAN) 4052 y una red de área amplia general (WAN) 4054. Tales ambientes de red son lugares comunes en oficinas, redes de cómputo de empresas grandes, intranet, el Internet, redes de teléfono fijo y móvil, redes inalámbricas al hoc y de infraestructura, otras redes inalámbricas, redes de juegos, algunas combinaciones de

ellos, etc. Tales redes y conexiones de comunicación son ejemplos de medios de transmisión.

Cuando se han implementado en una ambiente de red LAN, el computador 4002 está usualmente conectado a la LAN 4052 por medio de una interfaz de red o adaptador 4056. Cuando se implementa en un ambiente de red WAN, el computador 4002 típicamente incluye un módem 4058 u otros medios para establecer comunicaciones sobre la WAN 4054. El módem 4058, puede ser interno o externo al computador 4002, puede estar conectado al bus del sistema 4008 por medio de interfaces de entrada/salida 4042 o cualquier otro mecanismo apropiado. Debe apreciarse que las conexiones de red ilustradas son ilustrativas y que otros medios de establecer enlace de comunicación entre computadores 4002 y 4050 pueden ser empleados.

Más aún, otro hardware que esté diseñado específicamente para servidores pueden ser empleados. Por ejemplo, las tarjetas de aceleración SSL pueden ser usadas para descargar computaciones SSL. Adicionalmente, especialmente en un ambiente operativo de balance de carga de red, el hardware de descarga TCP y/o los clasificadores de paquetes en interfaces de red o

:

~~787~~

adaptadores 4056 (por ejemplo en tarjetas de interfaz de red) pueden ser instalados y usados en dispositivos servidores.

En un ambiente en red, tal como el ilustrado con el ambiente operativo 4000, los módulos del programa y otras instrucciones que son ilustradas en relación al computador 4002 o porciones de ellas, pueden ser almacenadas completamente o parcialmente en un dispositivo de almacenamiento de medio remoto. Por vía de ejemplo, los programas de aplicación remotos 4060 residen en un componente de memoria de un computador remoto 4050 pero pueden ser utilizables y de otra manera accesible por medio del computador 4002. También, para propósitos de ilustración, los programas de aplicación 4030 y otras instrucciones ejecutables por procesador tales como el sistema operativo 4028 son ilustrados aquí como bloques discretos, pero se debe reconocer que tales programas, componentes, y otras instrucciones residen a diversos tiempos en diferentes componentes de almacenamiento del dispositivo de cómputo 4002 (y/o el dispositivo de cómputo remoto 4050) y son ejecutados por el procesador o procesadores 4004 del computador 4002 (y/o aquellos del dispositivo de cómputo remoto 4050).

Aunque se han descrito sistemas, medios, dispositivos, métodos, procedimientos, aparatos, técnicas, esquemas, enseñanzas, procedimientos, disposiciones, y otras implementaciones en lenguaje específico rasgos estructurales, lógicos, algorítmicos, y funcionales y/o diagramas debe entenderse que la invención definida en la reivindicaciones adjuntas no necesariamente están limitadas a los rasgos específicos o diagramas descritos. En su lugar, los rasgos y diagramas específicos se describen como formas ilustrativas de implementación de la invención reivindicada.

REIVINDICACIONES

1. Uno o más medios accesibles por procesador comprenden instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen un sistema para llevar a cabo acciones que comprende:

Recibir información de estado del anfitrión desde una pluralidad de anfitriones; y

Tomar decisiones de balance de carga en respuesta a la información de estado de anfitrión recibida.

2. el uno o más medios accesibles por procesador según la reivindicación 1, comprenden instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen el sistema para llevar a cabo una acción adicional que comprende:

Recibir una solicitud para una nueva conexión desde un cliente; y

En donde la acción de hacer comprende una acción de:

Seleccionar un objetivo destino para la nueva conexión en respuesta a la información de estado del anfitrión recibida.

3. El uno o más medios accesibles por procesador según la reivindicación 1, en donde la información del estado del anfitrión comprende información de salud y/o carga que está asociado con la pluralidad de anfitriones y la acción de recibir comprende una acción de recibir comprende una acción de:

Recibir la información de salud y/o carga de la pluralidad de anfitriones.

4. El uno o más medios accesibles por procesador según la reivindicación 1, en donde la información del estado del anfitrión comprende información de estado de anfitrión específica a la aplicación, y la acción de hacer comprende una acción de:

Tomar decisiones de balance de carga en respuesta a la información de estado de anfitrión específica a aplicación.

5. El uno o más medios accesibles por procesador según la reivindicación 1, en donde la acción de recibir comprende por lo menos una acción de:

Recibir la información de estado de anfitrión desde uno o más anfitriones de la pluralidad de anfitriones; y

Recibir la información de estado de anfitrión indirectamente desde uno o más anfitriones de la pluralidad de anfitriones.

6. Un sistema comprende:

Por lo menos un procesador; y

Uno o más medios que incluyen instrucciones ejecutables por procesador que son capaces de ser ejecutadas por al menos un procesador, las instrucciones ejecutables por procesador

adaptadas para dirigir el sistema para llevar a cabo acciones que comprende:

Recibir información de estado de anfitrión de una pluralidad de anfitriones; y

Tomar decisiones de balance de carga en respuesta a la información de estado de anfitrión recibida.

7. El sistema según la reivindicación 6, en donde la acción de recibir comprende una acción de:

Recibir la información de estado de anfitrión desde una pluralidad de anfitriones por medio de por lo menos un programa enrutador.

8. El sistema según la reivindicación 6, en donde la información de estado de anfitrión incluye por lo menos una directiva de balanceo de carga; y en donde la acción de recibir comprende una acción de:

Recibir en al menos una directiva de balanceo de carga desde la pluralidad de anfitriones por

medio de por lo menos un programa enrutador que está invocando una o más interfaces de programación de aplicación (API) para empujar en por lo menos una directiva de balanceo de carga.

9. El sistema según la reivindicación 6, en donde el sistema comprende en por lo menos un dispositivo único y una pluralidad de dispositivos.

10. Un método para balancear la carga de red, el método comprende:

Determinar la información de estado de anfitrión para una pluralidad de anfitriones;

Enviar la información de estado de anfitrión desde la pluralidad de anfitriones a por lo menos un dispositivo que comprende la infraestructura de balanceo de carga;

Recibir la información de estado de anfitrión desde la pluralidad de anfitriones en por lo menos un dispositivo; y

Tomar decisiones de balanceo de carga con la infraestructura de balanceo de carga que responde a la información de estado de anfitrión recibida.

11. El uno o más medios accesibles por procesador comprenden instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen un sistema para llevar a cabo acciones que comprenden:

Acumular información de estado de anfitrión en una pluralidad de anfitriones; y

Enviar la información de estado de anfitrión acumulada desde la pluralidad de anfitriones.

12. El uno o más medios accesibles por procesador según la reivindicación 11, en donde la información de estado de anfitrión comprende información de salud y/o carga que está asociada con la pluralidad de anfitriones y la acción de enviar comprende una acción de:

Enviar la información de salud y/o carga desde la pluralidad de anfitriones.

13. El uno o más medios accesibles por procesador según la reivindicación 11, en donde la acción de acumular comprende una acción de acumular comprende una acción de:

Acumular la información de estado de anfitrión con respecto a las aplicaciones que se ejecutan en la pluralidad de anfitriones.

14. El uno o más medios accesibles por procesador según la reivindicación 11, en donde la acción de enviar comprende una acción de:

Enviar la información de estado de anfitrión acumulada desde la pluralidad de anfitriones a por lo menos un programa enrutador.

15. El uno o más medios accesibles por procesador según la reivindicación 11, en donde la acción de enviar comprende una acción de:

Enviar la información de estado de anfitrión acumulad
desde la pluralidad de anfitriones a la
funcionalidad de balanceo de carga.

16. Un sistema comprende:

Un determinador de información de estado de anfitrión
que está adaptado para determinar la información
de estado de anfitrión para un anfitrión; y

Un diseminador de información de estado de anfitrión
que está adaptado para diseminar la información
de estado de anfitrión a la infraestructura de
balanceo de carga.

17. El sistema según la reivindicación 16, en donde
el sistema comprende adicionalmente:

Una o más aplicaciones;

En donde el determinador de información de estado de
anfitrión determina la información de estado de
anfitrión con respecto a una o más aplicaciones.

18. El sistema según la reivindicación 16, en donde el sistema comprende un dispositivo sobre el cual el anfitrión está residente.

19. El sistema según la reivindicación 16, en donde el sistema comprende adicionalmente:

Otro determinador de información de estado de anfitrión que está adaptado para determinar información de estado de anfitrión adicional para otro anfitrión, y

Otro diseminador de información de estado de anfitrión que está adaptado para diseminar la información de estado de anfitrión adicional a la infraestructura de balanceo de carga;

En donde el sistema comprende un primer dispositivo sobre el cual el anfitrión es residente y un segundo dispositivo sobre el cual el otro anfitrión es residente.

20. Uno o más medios accesibles por procesador que comprenden instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten a un sistema implementar un protocolo de mensajes entre por lo menos un anfitrión y una o más unidades de balanceo de carga, el protocolo de mensajes es utilizable para comunicar la información de salud y/o carga entre el por lo menos un anfitrión y la una o más unidades de balanceo de carga.

21. El uno o más medios accesibles por procesador según la reivindicación 20, en donde por lo menos una porción de las instrucciones ejecutables por procesador comprenden software que está adaptado para ser ejecutado sobre el por lo menos un anfitrión.

22. El uno o más medios accesibles por procesador según la reivindicación 20, en donde por lo menos una porción de las instrucciones ejecutables por procesador comprenden software que está adaptado para ser ejecutado sobre la una o más unidades de balanceo de carga.

23. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensajes que incluye por lo menos un mensaje, el por lo menos un mensaje comprende un mensaje tipo latido de corazón que indica a la una o más unidades de balanceo de carga que el por lo menos un anfitrión está funcionando.

24. El uno o más medios accesibles por procesador según la reivindicación 23, en donde un formato del mensaje tipo latido de corazón comprende un identificador para el por lo menos un anfitrión, datos de verificación de error para la información de salud y/o carga, y un nombre del sistema de nombre de dominio (DNS).

25. El uno o más medios accesibles por procesador según la reivindicación 23, en donde un formato del mensaje tipo latido de corazón permite la

inclusión de un par de número de agrupamiento/identificador de generación (ID).

26. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende las instrucciones ejecutables por procesador que, cuando son ejecutadas, permiten al sistema implementar el protocolo de mensaje que incluye por lo menos un mensaje de adiós que indica a la una o más unidades de balanceo de carga que el por lo menos un anfitrión está planeando apagarse.

27. El uno o más medios accesibles por procesador según la reivindicación 26, en donde un formato del mensaje adiós comprende un identificador para el por lo menos un anfitrión.

28. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensaje que incluye por lo menos un mensaje, el por lo menos un

mensaje comprende un mensaje de cambio de fila que le indica a la una o más unidades de balanceo de carga que la información de salud y/o carga para una aplicación de el por lo menos un anfitrión ha cambiado.

29. El uno o más medios accesibles por procesador según la reivindicación 28, en donde un formato del mensaje de cambio de fila comprende un identificador para el por lo menos un anfitrión, un identificador para la aplicación, una operación para reflejar el cambio, y datos para la operación.

30. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensajes que incluye por lo menos un mensaje, el por lo menos un mensaje comprende un mensaje de obtenga una tabla de manera instantánea que es enviado desde la una o más unidades de balanceo de carga a el por lo menos un anfitrión, el mensaje de obtenga una

~~424~~

tabla instantánea solicitando una instantánea de la información actual de salud y/o carga del por lo menos un anfitrión.

31. El uno o más medios accesibles por procesador de acuerdo con la reivindicación 30, en donde un formato del mensaje de obtenga una tabla instantánea comprende una identificación de una solicitud de unidad de balanceo de carga de la una o más unidades de balanceo de carga.

32. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensajes que incluye por lo menos un mensaje, el por lo menos un mensaje comprende un mensaje de envíe una tabla instantánea que es enviado desde el por lo menos un anfitrión a una unidad de balanceo de solicitada de uno de la una o más unidades de balanceo de carga, el mensaje de envíe tabla instantánea que suministra una instantánea de la

información actual de salud y/o carga del por lo menos un anfitrión.

33. El uno o más medios accesibles por procesador según con la reivindicación 32, en donde un formato del mensaje de envíe tabla instantánea comprende la instantánea de la información actual de salud y/o carga del por lo menos un anfitrión.

34. El uno o más medios accesibles por procesador según la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensajes que incluye por lo menos un mensaje, el por lo menos un mensaje comprende un mensaje de estado de tabla postulada que es enviado desde el por lo menos un anfitrión a una o más unidades de balanceo de carga, el mensaje de estado de tabla postulada incluye una directiva del estado de balanceo de carga que indica una directiva de estado de balanceo de carga actual que es esperada por el por lo menos un anfitrión para que exista en una o más unidades de balanceo de carga.

35. El uno o más medios accesibles por procesador de acuerdo con la reivindicación 34, en donde un formato del mensaje de estado de tabla postulada comprende un identificador para el por lo menos un anfitrión y la directiva de estado de balanceo de carga actual.

36. El uno o más medios accesibles por procesador de acuerdo con la reivindicación 20, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, le permiten al sistema implementar el protocolo de mensajes que incluye por lo menos un mensaje, el por lo menos un mensaje comprende un mensaje de equivocado postulado que es enviado desde una unidad de balanceo de carga de la una o más unidades de balanceo de carga a por lo menos un anfitrión, el cual previamente ha enviado un mensaje de estado de tabla postulada; el mensaje de postulado equivocado indica que la unidad de balanceo de carga tiene una directiva de estado de balanceo de carga actual que difiere de una directiva de estado de balanceo de carga postulada que está

incluida en el mensaje de estado de tabla postulada.

37. Un sistema comprende:

Por lo menos un dispositivo que es anfitrión de una o más aplicaciones, el por lo menos un dispositivo incluye una tabla de salud y carga que incluye una pluralidad de entradas, cada entrada de la pluralidad de entradas asociadas con una aplicación de la una o más aplicaciones; cada entrada de la pluralidad de entradas comprende:

Un identificador de aplicación para una aplicación en particular de la una o más aplicaciones;

Información que caracteriza por lo menos un estado de la aplicación particular, y

Por lo menos una directiva de balanceo de carga con respecto a la aplicación particular.

38. El sistema según la reivindicación 37, en donde el identificador de aplicación identifica de

manera única la aplicación particular entre la una o más aplicaciones.

39. El sistema según la reivindicación 37, en donde el identificador de aplicación comprende por lo menos una dirección y puerto de protocolo de Internet virtual (IP), una dirección y puerto de IP física, un protocolo relevante a la aplicación en particular, e información que especifica al protocolo.

40. El sistema según la reivindicación 37, en donde el identificador de aplicación comprende por lo menos un identificador único globalmente (GUID).

41. El sistema según la reivindicación 37, en donde la información que caracteriza por lo menos un estado de la aplicación en particular comprende por lo menos una salud, carga de aplicación, y capacidad de aplicación.

42. El sistema según la reivindicación 41, en donde la salud de la aplicación indica si un estado de la aplicación particular es saludable, falla, o

es desconocido; en donde la carga de la aplicación indica que tan ocupada está la aplicación en particular; y en donde la capacidad de aplicación indica una capacidad máxima de la aplicación particular.

43. El sistema según la reivindicación 42, en donde la capacidad máxima de la aplicación particular es expresada en relación a una capacidad total para las aplicaciones del mismo tipo que aquella de la aplicación particular que se está ejecutando en el sistema.

44. El sistema según la reivindicación 42, en donde la capacidad máxima de la aplicación particular está expresada en una unidad menor y número ligado.

45. El sistema según la reivindicación 37, en donde el por lo menos una directiva de balanceo de carga puede estar suministrada a una pluralidad de unidades de balanceo de carga para suministrar guía al balanceo de carga de la red con respecto

a la aplicación particular y con respecto a otras aplicaciones de un mismo tipo de aplicación.

46. El sistema según la reivindicación 37, en donde el por lo menos una directiva de balanceo de carga comprende por lo menos una activa, drenaje, e inactiva.

47. El sistema según la reivindicación 37, en donde el por lo menos una directiva de balanceo de carga comprende una directiva de estado de balanceo de carga objetivo y una directiva de estado de balanceo de carga actual.

48. El sistema según la reivindicación 47, en donde la directiva de estado de balanceo de carga objetivo indica un estado de balanceo de carga que la infraestructura de salud y carga, que está localizada en el por lo menos un dispositivo, tiene el propósito para la infraestructura de salud y carga, que está localizada en una pluralidad de unidades de balanceo de carga, para ser operada posteriormente.

49. El sistema según la reivindicación 47, en donde la directiva de estado de balance de carga actual indica un estado de balance de carga que la infraestructura de salud y carga, que está localizada en por lo menos un dispositivo, considera que la infraestructura de salud y carga, que está localizada en una pluralidad de unidades de balanceo de carga, está actualmente bajo operación.

50. El sistema según la reivindicación 37, en donde el sistema comprende adicionalmente:

Una pluralidad de dispositivos, con cada uno de los dispositivos respectivos de una pluralidad de dispositivos incluyendo una tabla respectiva de salud y carga.

51. Uno o más medios de accesibles por procesador que comprende instrucciones ejecutables por procesador que comprenden por lo menos parte de un sistema operativo, en donde las instrucciones ejecutables por procesador, cuando son



ejecutadas, dirigen a un sistema a llevar a cabo acciones que comprende:

Determinar cuando una aplicación se inicia;

Reportar, en respuesta a la determinación, que la aplicación ha iniciado a la infraestructura de salud y carga que es capaz de comunicar con la infraestructura de balance de carga de red;

Detectar cuando la aplicación se ha detenido; y

Reportar, en respuesta a la detección, la aplicación como detenida a la infraestructura de salud y carga.

52. El uno o más medios accesibles por procesador según la reivindicación 51, en donde por lo menos una porción de las instrucciones ejecutables por procesador comprenden por lo menos parte de un administrador de control del sistema operativo.

53. Uno o más medios accesibles por procesador que comprenden instrucciones ejecutables por

procesador que comprende por lo menos parte de un sistema operativo, en donde las instrucciones ejecutables por procesador, cuando son ejecutadas, dirigen a un sistema a llevar a cabo acciones que comprenden:

Identificar que una aplicación se ha iniciado como resultado del inicio de la aplicación;

Reportar, en respuesta a la identificación, la aplicación como iniciada a la infraestructura de salud y carga que es capaz de comunicarse con la infraestructura de balanceo de carga de red;

Establecer que la aplicación se ha detenido como resultado de detener la aplicación; y

Reportar, en respuesta al establecimiento, la aplicación como detenida a la infraestructura de salud y carga.

54. El uno o más medios accesibles por procesador según la reivindicación 53, en donde por lo menos una porción de las instrucciones ejecutables por

procesador comprenden por lo menos parte de un administrador de control del sistema operativo.

55. Un sistema comprende:

Una infraestructura de salud y carga que está adaptada para determinar la información de salud y carga específica de la aplicación; y

La infraestructura de balanceo de carga que está adaptada para utilizar la información de salud y carga específica de la aplicación cuando se asignan solicitudes a una pluralidad de aplicaciones.

56. El sistema según la reivindicación 55, en donde la infraestructura de salud y carga comprende una tabla de salud y carga que almacena por lo menos una porción de la información de salud y carga específica a la aplicación.

57. El sistema según la reivindicación 55, en donde la infraestructura de salud y carga comprende una tabla de salud y carga que almacena por lo menos

una porción de la información de salud y carga específica a la aplicación; La tabla de salud y carga incluye una pluralidad de entradas, cada entrada de la pluralidad de entradas está asociada con una aplicación particular de la pluralidad de aplicaciones.

58. El sistema según la reivindicación 55, en donde la infraestructura de salud y carga comprende una tabla de salud y carga que almacena por lo menos una porción de la información de salud y carga específica a la aplicación; la tabla de salud y carga incluye una pluralidad de entradas, cada entrada de la pluralidad de entradas incluye: un identificador de aplicación para una aplicación particular con la cual la entrada está asociada, información que caracteriza por lo menos un estado de la aplicación particular, y por lo menos un estado de la aplicación particular, y por lo menos una directiva de balanceo de carga con respecto a la aplicación particular.

59. El sistema según la reivindicación 55, en donde la infraestructura de balanceo de carga comprende

una memoria caché de salud y carga consolidada que almacena la información de salud y carga específica a la aplicación.

60. El sistema según la reivindicación 55, en donde la infraestructura de balanceo de carga comprende una memoria caché de salud y carga consolidada que almacena la información de salud y carga específica a la aplicación para la pluralidad de aplicaciones que están siendo ejecutadas en una pluralidad de anfitriones.

61. El sistema según la reivindicación 55, en donde la información de salud y carga específica la aplicación comprende información de salud y carga específica del punto de extremo de la aplicación.

62. El sistema según la reivindicación 55, que comprende adicionalmente:

Un dispositivo de programa enrutador que comprende por lo menos parte de la infraestructura de salud y carga, el por lo menos de la infraestructura de salud y carga adaptada para determinar la

información de salud y carga específica a la aplicación llevando a cabo acciones de monitoreo externo.

63. El sistema según la reivindicación 55, en donde:

La infraestructura de salud y carga comprende una pluralidad de tablas de salud y carga que almacenan la información de salud y carga específica a la aplicación, y

La infraestructura de balanceo de carga que comprende una pluralidad de caché de salud y carga consolidados que almacenan la información de salud y carga específica a la aplicación.

64. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones sobre los cuales la infraestructura de salud y carga se distribuye, cada anfitrión de la pluralidad y anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga que corresponde a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una caché de salud y carga consolidada de la pluralidad de caché de salud y carga consolidada.

65. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones sobre los cuales la infraestructura de salud y carga se distribuye, cada anfitrión de la pluralidad y anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga correspondiente a por lo menos una porción de la infraestructura de balanceo y carga, cada unidad de balanceo y carga de la pluralidad de unidades de balanceo de carga tiene una caché de salud y

carga consolidada de la pluralidad de caché de salud y carga consolidadas;

En donde cada memoria caché de salud y carga consolidada de la pluralidad de memorias caché de salud y carga incluye la información de salud y carga específica a aplicación que es almacenada en cada tabla de salud y carga de la pluralidad de tablas de salud y carga.

66. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones sobre los cuales la infraestructura de salud y carga se distribuye, cada anfitrión de la pluralidad de anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga correspondiente a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una memoria caché de

salud y carga consolidada de la pluralidad de memorias caché de salud y carga consolidadas;

En donde la pluralidad de aplicaciones se ejecutan sobre la pluralidad de anfitriones.

67. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones residentes en una pluralidad de dispositivos, la infraestructura de salud y carga distribuida sobre la pluralidad de anfitriones, cada anfitrión de la pluralidad y anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga comprendidas de por lo menos un dispositivo y correspondientes a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una memoria caché de salud y carga consolidada de la pluralidad de memorias caché de salud y carga consolidadas;

68. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones residentes en una pluralidad de dispositivos, la infraestructura de salud y carga distribuida sobre la pluralidad de anfitriones, cada anfitrión de la pluralidad de anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga comprendidas por lo menos un dispositivo y corresponden a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una memoria caché de salud y carga consolidada de la pluralidad de memorias caché de salud y carga consolidadas;

En donde la infraestructura de salud y carga comprende una parte remota de la infraestructura de balanceo de carga.

69. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones residentes en una pluralidad de dispositivos, la infraestructura de salud y carga distribuida sobre la pluralidad de anfitriones, cada anfitrión de la pluralidad de anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga comprendidas de por lo menos un dispositivo y correspondientes a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una memoria caché de salud y carga consolidada de la pluralidad de memorias caché de salud y carga consolidadas;

En donde el por lo menos un dispositivo es uno de la pluralidad de dispositivos.

70. El sistema según la reivindicación 63, en donde el sistema comprende adicionalmente:

Una pluralidad de anfitriones residentes en una pluralidad de dispositivos, la infraestructura de salud y carga distribuida sobre la pluralidad de anfitriones, cada anfitrión de la pluralidad de anfitriones tienen una tabla de salud y carga de la pluralidad de tablas de salud y carga; y

Una pluralidad de unidades de balanceo de carga comprendidas de por lo menos un dispositivo y correspondientes a por lo menos una porción de la infraestructura de balanceo de carga, cada unidad de balanceo de carga de la pluralidad de unidades de balanceo de carga tiene una memoria caché de salud y carga consolidada de una pluralidad de memorias caché de salud y carga consolidadas;

En donde el por lo menos un dispositivo no es uno de la pluralidad de dispositivos; y

En donde la infraestructura de salud y carga está adaptada adicionalmente para diseminar la información de salud y carga específica a la

aplicación desde la pluralidad de dispositivos al por lo menos un dispositivo.

71. El sistema según la reivindicación 55, en donde la infraestructura de salud y carga y la infraestructura de balanceo de carga son capaces de usar un protocolo de mensaje para las comunicaciones entre ellas que se relaciona con la información de salud y carga específica a la aplicación.

72. El sistema según la reivindicación 71, en donde el protocolo de mensajes comprende uno o más de los tipos de mensaje siguientes: un tipo de mensaje de latido de corazón, un tipo de mensaje adiós, un tipo de mensaje de cambio de fila, un tipo de mensaje de obtenga una tabla instantánea, un tipo de mensaje de envíe una tabla instantánea, un tipo de mensaje de estado de tabla postulado, y un tipo de mensaje de equivocación de postulado.

73. El sistema según la reivindicación 71, en donde el protocolo de mensajes incluye una capacidad para comunicarse usando miembros del grupo.
74. El sistema según la reivindicación 55, en donde la infraestructura de balanceo de carga, después de una falla, es capaz de recuperar la información de salud y carga específica a la aplicación por medio de la infraestructura de salud y carga usando un protocolo de mensaje para las comunicaciones entre ellas.
75. El sistema según la reivindicación 55, en donde la infraestructura de balanceo de carga está adaptada adicionalmente para asignar solicitudes a la pluralidad de aplicaciones usando uno o más esquemas de asignación.
76. El sistema según la reivindicación 75, en donde el uno o más esquemas de asignación comprenden por lo menos uno de un esquema de asignación de señal y un esquema de asignación de porcentaje.

77. El sistema según la reivindicación 75, en donde el uno o más esquemas de asignación involucra usar un mecanismo de expiración temporizado.

78. El sistema según la reivindicación 75, en donde el uno o más esquemas de asignación son implementados por un componente manejador de salud y carga de la infraestructura de balanceo de carga.

79. Uno o más medios accesibles por procesador comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen a un sistema a llevar a cabo acciones que comprende:

Analizar la información de salud y /o carga para una pluralidad de puntos de extremo de aplicación; y

Determinar una asignación de señal para la pluralidad de punto de extremo de aplicación en respuesta al análisis.

80. El uno o más medios accesibles por procesador según la reivindicación 79, que comprende instrucciones ejecutables por procesador que, cuando se ejecutan, dirigen al sistema a llevar a cabo acciones adicionales que comprenden:

Recibir una solicitud de asignación de punto de extremo de una aplicación objetivo que identifica la pluralidad de puntos de extremo de la aplicación; y

Enviar una respuesta de asignación de punto de extremo de la aplicación objetivo que incluye la asignación de señal.

81. El uno o más medios accesibles por procesador como se da en la reivindicación 79, comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo acciones adicionales que comprenden:

Recibir una solicitud de asignación de punto de extremo de la aplicación objetivo que incluye uno

o más de una dirección y puerto de protocolo de Internet virtual (IP), un protocolo, e información que especifica al protocolo; y

Enviar una respuesta de asignación de punto de extremo de la aplicación objetivo que incluye una dirección y puerto IP físicos.

82. El uno o más medios accesibles por procesador según la reivindicación 79, en donde la acción de analizar comprende una acción de:

Analizar la información de salud y/o carga específica al punto de extremo de la aplicación para la pluralidad de puntos de extremos de la aplicación.

83. El uno o más medios accesibles por procesador según la reivindicación 79, en donde la acción de determinar comprende una acción de:

Determinar la asignación de señal para la pluralidad de puntos de extremo de la aplicación con base en las capacidades disponibles relativas de la

pluralidad de puntos de extremos de la aplicación.

84. El uno o más medios accesibles por procesador según la reivindicación 79, en donde la asignación de señal comprende un primer número señales que corresponde a un primer punto de extremo de aplicación de la pluralidad de puntos de extremo de aplicación y un segundo número de señales correspondientes a un segundo punto de extremo de aplicación de la pluralidad de puntos de extremos de aplicación.

85. El uno o más medios accesibles por procesador según la reivindicación 79, en donde la asignación de señal comprende:

Un primer número de señales correspondiente a un primer punto de extremo de aplicación de la pluralidad de puntos de extremos de aplicación;

Un segundo número de señales correspondiente a un segundo punto de extremo de aplicación de la pluralidad de puntos de extremo de aplicación; y

Un tiempo límite, en donde una expiración del tiempo límite convierte en no utilizable cualesquier señales remanentes del primer número de señales y del segundo número de señales.

86. El uno o más medios accesibles por procesador según la reivindicación 79, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas dirigen al sistema a llevar a acabo una acción adicional que comprende:

Usar la asignación de señales para clasificar las solicitudes de conexión entrantes.

87. El uno o más medios accesibles por procesador que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen a un sistema a llevar a cabo acciones que comprenden:

Determinar la información de salud y carga sobre una base por aplicación; y

Seleccionar una aplicación entre una pluralidad de aplicaciones que responden a la información de salud y carga.

88. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de determinar comprende una acción de:

Determinar cuando las aplicaciones de la pluralidad de aplicaciones se inician y se detienen.

89. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de determinar comprende una acción de:

Determinar cuando una aplicación de la pluralidad de aplicaciones es saludable y cuando la aplicación está fallando o ha fallado.

90. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de determinar comprende una acción de:

Determinar una carga de una aplicación dada de un tipo de aplicación particular relativo a la carga de uno o más de las otras aplicaciones del tipo de aplicación particular.

91. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Recibir una entrada externa con respecto a la determinación de la información de salud y carga sobre una base por aplicación;

En donde la acción de determinar comprende una acción de:

Determinar la información de salud y carga sobre una base por aplicación de acuerdo con una entrada externa.

92. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende

instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Diseminar la información de salud y carga desde por lo menos un anfitrión a una o más unidades de balanceo de carga.

93. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Diseminar la información de salud y carga desde por lo menos un anfitrión a una o más unidades de balanceo de carga usando un protocolo de mensaje.

94. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Diseminar la información de salud y carga desde por lo menos un anfitrión a una o más unidades de balanceo de carga usando un agrupamiento de miembros.

95. El uno o más medios accesibles por procesador según la reivindicación 94, en donde la acción de diseminar comprende una acción de:

Enviar un mensaje de latido de corazón desde el por lo menos un anfitrión a un anfitrión líder, en donde el mensaje de latido de corazón incluye una indicación de envío tal que el anfitrión líder es dirigido a enviar el mensaje de latido de corazón a uno o más unidades de balanceo de carga aún si no hay un cambio en el agrupamiento de miembros involucrado.

96. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Diseminar la información de salud y carga desde por lo menos una tabla de salud y carga a uno o más memorias caché de salud y carga consolidadas.

97. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Recibir la información de salud y carga desde una pluralidad de anfitriones; y

Guardar en memoria caché la información de salud y carga.

98. El uno o más medios accesibles por procesador según la reivindicación 87, que comprende instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen al sistema a llevar a cabo una acción adicional que comprende:

Recibir la información de salud y carga desde una pluralidad de anfitriones;

~~456~~

Guardar en memoria caché la información de salud y carga recibida;

Recibir un paquete que solicita una iniciación de conexión; y

Consultar la información de salud y carga guardada en memoria caché para la iniciación de la conexión;

En donde la acción de seleccionar comprende una acción de:

Seleccionar la aplicación entre la pluralidad de aplicaciones en respuesta a la consulta.

99. El uno o más medios accesibles por procesador según la reivindicación 98, en donde la iniciación de la conexión pertenece a un tipo de aplicación particular.

100. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de seleccionar comprende una acción de:

Seleccionar un punto de extremo de aplicación entre una pluralidad de puntos de extremo de aplicación en respuesta a la información de salud y carga.

101. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de seleccionar comprende una acción de:

Seleccionar, en respuesta a la información de salud y carga, un punto de extremo de aplicación entre una pluralidad de puntos de extremos de aplicación que están distribuidas entre una pluralidad de anfitriones.

102. El uno o más medios accesibles por procesador según la reivindicación 87, en donde el uno o más medios accesibles por procesador comprende por lo menos uno de (i) uno o más medios de almacenamiento y (ii) uno o más medios de transmisión.

103. El uno o más medios accesibles por procesador según la reivindicación 87, en donde por lo menos

4/6

una porción de las instrucciones ejecutables por procesador comprenden por lo menos parte de un programa de software de balanceo de carga.

104. El uno o más medios accesibles por procesador según la reivindicación 87, en donde por lo menos una porción de las instrucciones ejecutables por procesador están adaptadas para correr sobre un sistema en donde el sistema comprende un dispositivo único.

105. El uno o más medios accesibles por procesador según la reivindicación 87, en donde por lo menos una porción de las instrucciones ejecutables por procesador están adaptadas para correr sobre un sistema en donde el sistema comprende una pluralidad de dispositivos.

106. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de seleccionar comprende una acción de:

Seleccionar, en respuesta a la información de salud y carga, una asignación de puntos de extremo de

aplicación entre una pluralidad de puntos de extremo de aplicación con respecto a capacidades disponibles relativas entre la pluralidad de punto de extremo de aplicación.

107. El uno o más medios accesibles por procesador según la reivindicación 106, en donde la acción de seleccionar comprende una acción adicional de:

Seleccionar la asignación de los puntos de extremo de aplicación usando un esquema de asignación de señal.

108. El uno o más medios accesibles por procesador según la reivindicación 106, en donde la acción de seleccionar comprende una acción adicional de:

Seleccionar la asignación de puntos de extremo de la aplicación usando un esquema de asignación de porcentaje.

109. El uno o más medios accesibles por procesador según la reivindicación 106, en donde la pluralidad de puntos de extremo de la aplicación

~~460~~

corresponden a aplicaciones de un tipo de aplicación única.

110. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de seleccionar comprende una acción de:

Seleccionar la aplicación entre una pluralidad de aplicaciones que responden a la información de salud y carga para poder balancear una carga de red causada por paquetes que ingresan.

111. El uno o más medios accesibles por procesador según la reivindicación 87, en donde la acción de seleccionar comprende una acción de:

Seleccionar la aplicación entre la pluralidad de aplicaciones en respuesta a la información de salud y carga para poder balancear una carga de red causada por solicitudes de conexión que ingresan.

112. Una disposición para balancear la carga de la red con información del estado de anfitrión, la disposición comprende:

Medios para determinar la información de estado de anfitrión para una pluralidad de anfitriones; y

Medios para tomar decisiones de balanceo de carga en respuesta a la información del estado del anfitrión.

113. La disposición de la reivindicación 112, en donde los medios para determinar están localizados en la pluralidad de anfitriones, y los medios para tomar decisiones están localizados en la infraestructura de balanceo de carga; y

En donde la disposición comprende adicionalmente:

Medios para diseminar la información de estado de anfitrión desde la pluralidad de anfitriones a la infraestructura de balanceo de carga.

114. La disposición de la reivindicación 112, en donde los medios para determinar comprenden medios para determinar información de estado de anfitrión específico a la aplicación para la pluralidad de anfitriones, y los medios para tomar decisiones comprenden medios para tomar decisiones de balanceo de carga en respuesta a la información de estado de anfitrión específico a la aplicación.

115. La disposición según la reivindicación 112, en donde la disposición comprende por lo menos un sistema.

116. La disposición según la reivindicación 112, en donde la disposición comprende uno o más medios accesibles por procesador.

117. Una disposición comprende:

Medios de acumulación para acumular información de estado de anfitrión en una pluralidad de anfitriones; y

Medios de envío para enviar la información de estado de anfitrión acumulada desde la pluralidad de anfitriones.

118. La disposición de la reivindicación 117, en donde la información de estado de anfitrión comprende información de salud y/o carga que está asociada con la pluralidad de anfitriones y los medios de envío comprende:

Medios para enviar la información de salud y/o carga desde la pluralidad de anfitriones.

119. La disposición según la reivindicación 117, en donde los medios de acumulación comprenden:

Medios para acumular la información de estado de anfitrión con respecto a las aplicaciones que se ejecutan sobre la pluralidad de anfitriones.

120. La disposición según la reivindicación 117, en donde los medios de enviar comprende:

Medios para enviar la información de estado de anfitrión acumulada desde la pluralidad de anfitriones hacia la funcionalidad de balanceo de carga.

121. Una disposición comprende:

Medios de determinación para determinar la información de salud y carga sobre una base por aplicación; y

Medios de selección para seleccionar una aplicación entre una pluralidad de aplicaciones en respuesta a la información de salud y carga.

122. La disposición según la reivindicación 121, en donde los medios de determinación comprende:

Medios para determinar cuando una aplicación de la pluralidad de aplicaciones está saludable y cuando la aplicación está fallando o ha fallado.

123. La disposición según la reivindicación 121, en donde los medios de determinación comprende:

Medios para determinar una carga de una aplicación dada de un tipo de aplicación particular en relación con cargar una o más de las otras aplicaciones del tipo de aplicación particular.

124. La disposición según la reivindicación 121, en donde los medios de selección comprenden:

Medios para seleccionar un punto de extremo de aplicación entre una pluralidad de puntos de extremo de aplicación en respuesta a la información de salud y carga.

212

467

RESUMEN

En una primera implementación de medios ilustrativos, uno o más medios accesibles por procesador incluyen instrucciones ejecutables por procesador que, cuando son ejecutadas dirigen a un sistema a llevar a cabo acciones que incluyen:

Acumular información de estado de anfitrión en múltiples anfitriones; y enviar la información de estado de anfitrión acumulada desde los múltiples anfitriones. En una segunda implementación de los medios ilustrativos, uno o más medios accesibles por procesador incluyen instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen a un sistema lleva a cabo acciones que incluyen: recibir información del estado de anfitrión desde múltiples anfitriones; y tomar decisiones de balanceo de carga en respuesta a la información recibida del estado de anfitrión. En una tercera implementación de medios ilustrativo, uno o más medios accesibles por procesador incluyen instrucciones ejecutables por procesador que, cuando son ejecutadas, dirigen a un sistema a llevar a cabo acciones que incluyen: determinar la información de salud y carga sobre una base por aplicación; y seleccionar una aplicación entre múltiples aplicaciones en respuesta a la información de salud y carga.

~~95~~

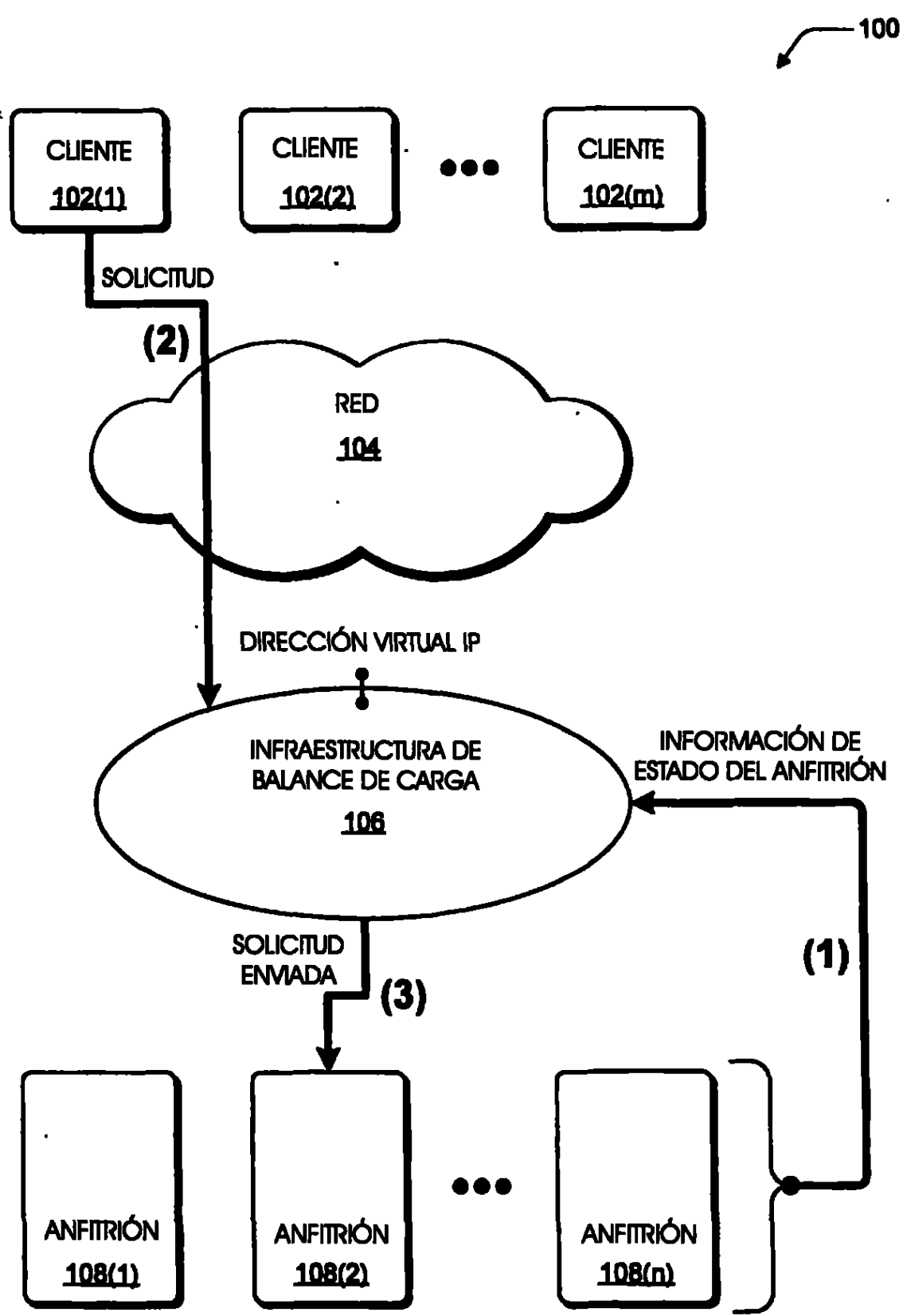


Fig. 1

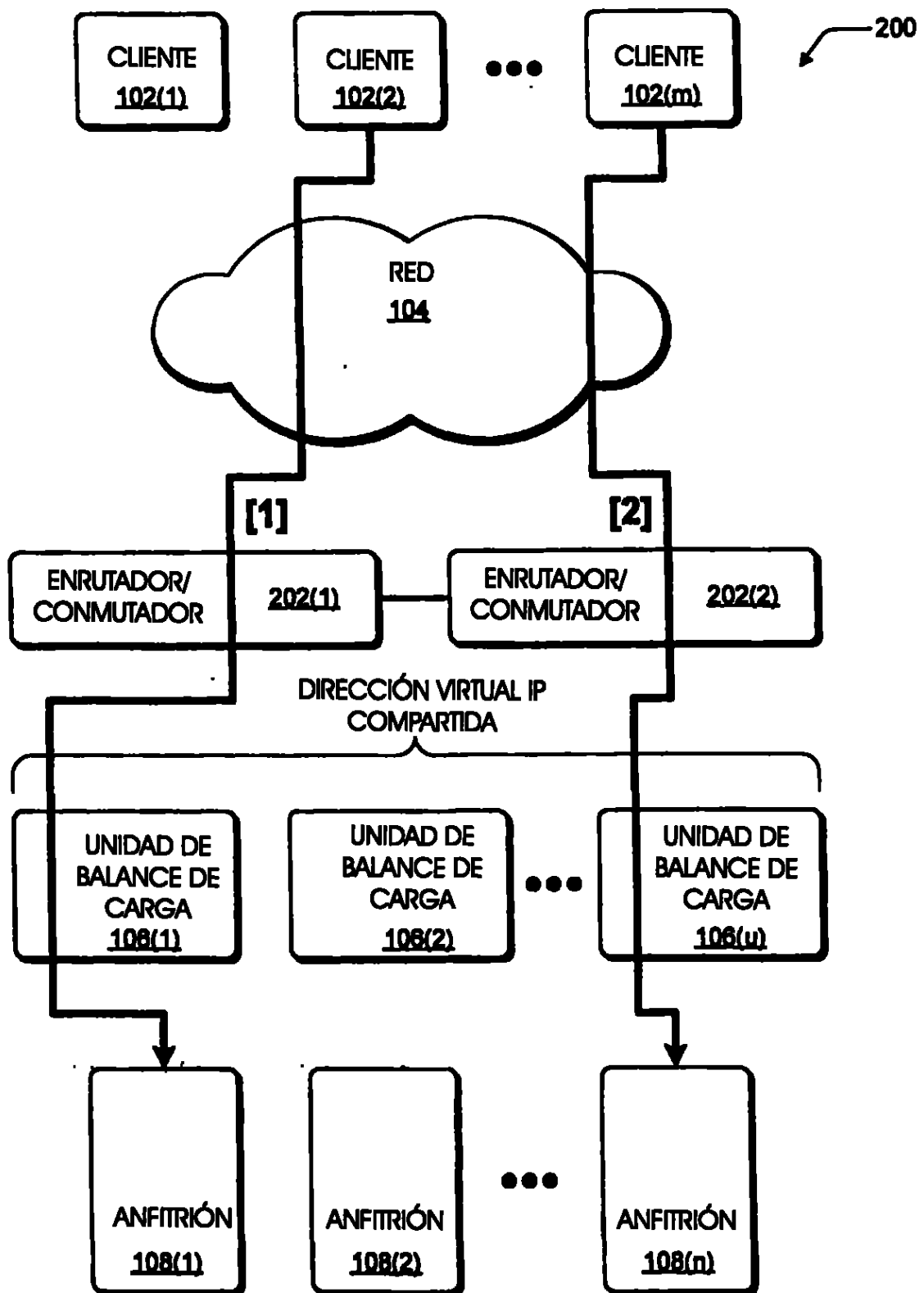


Fig. 2

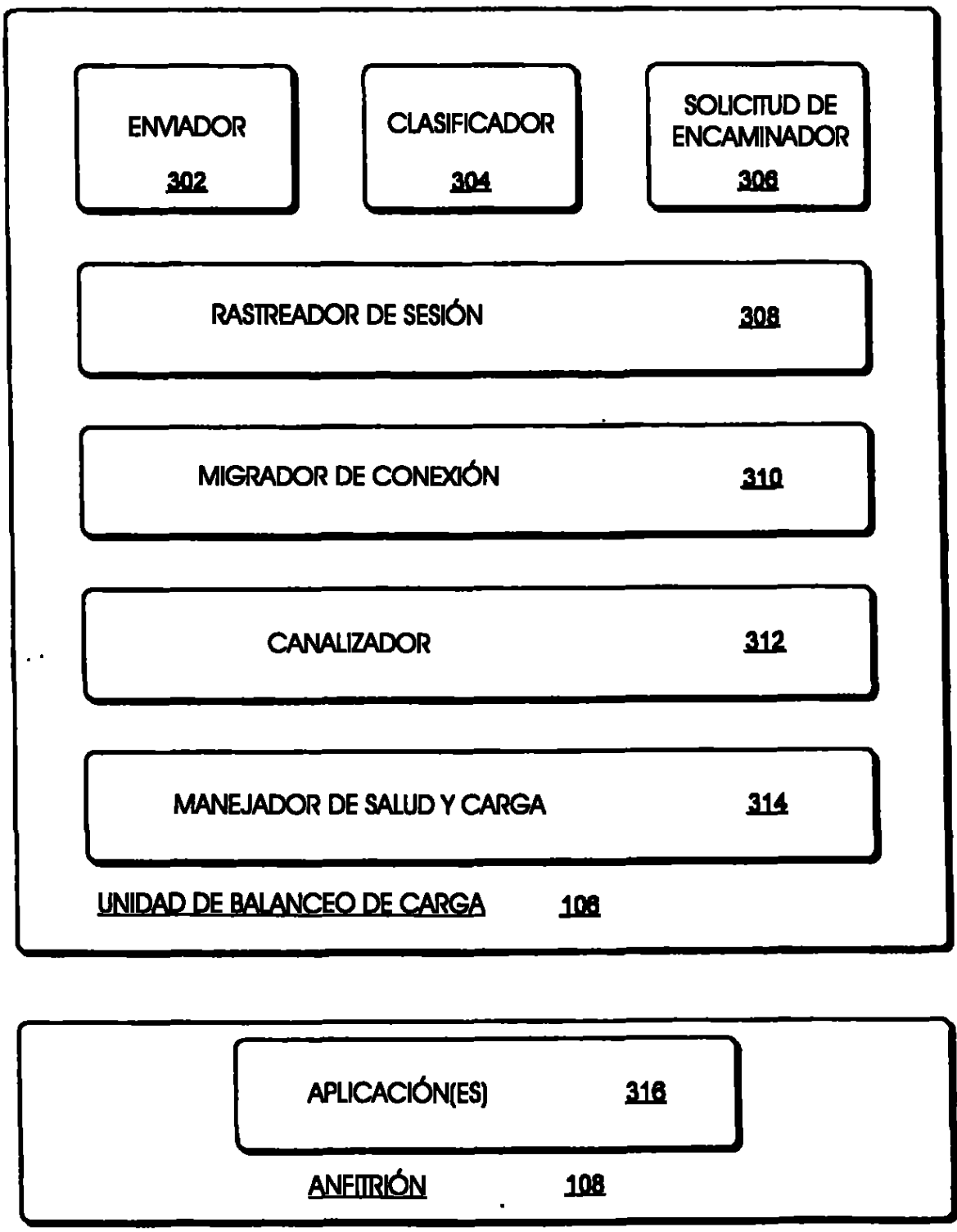


Fig. 3

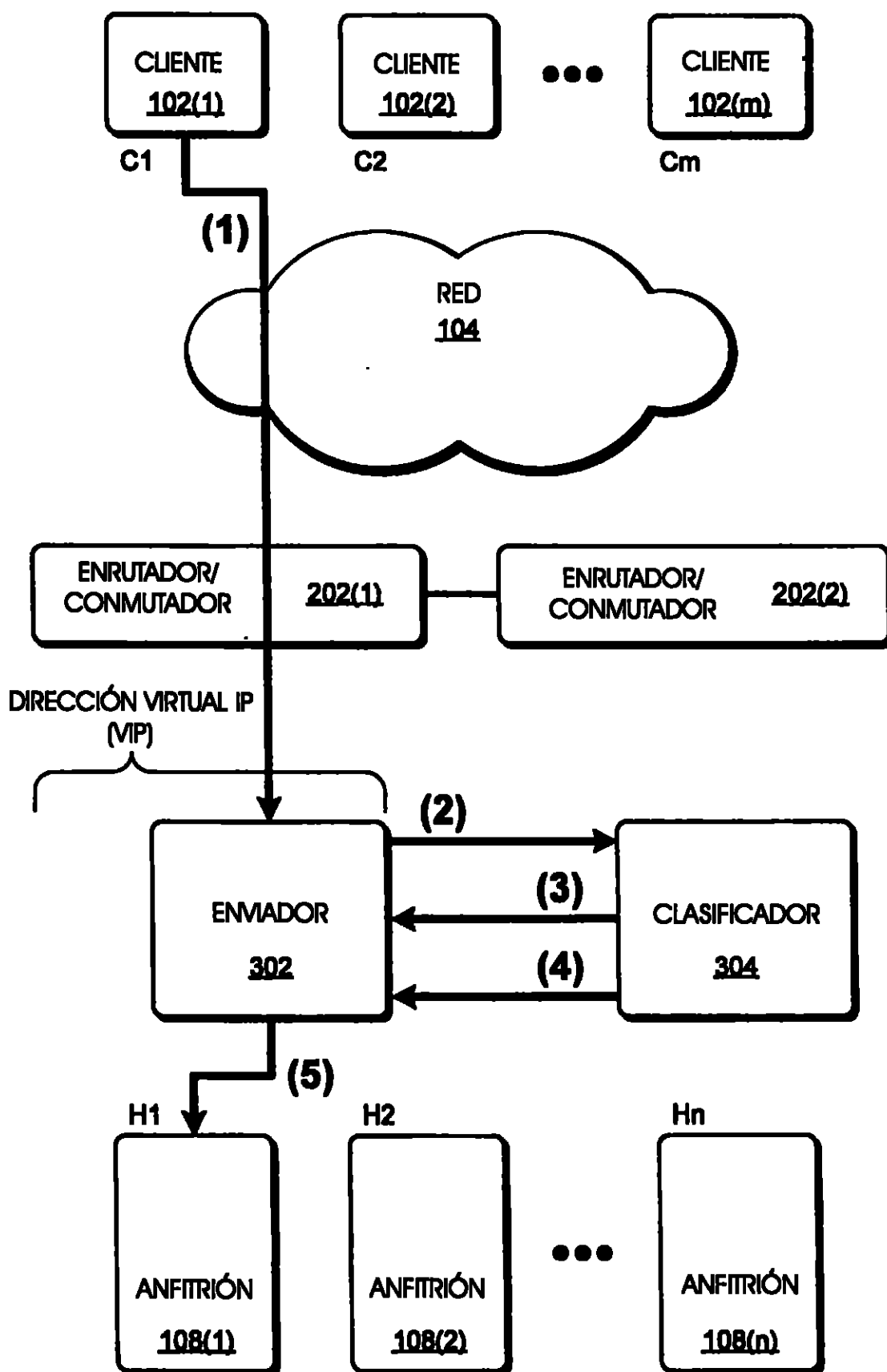


Fig. 4

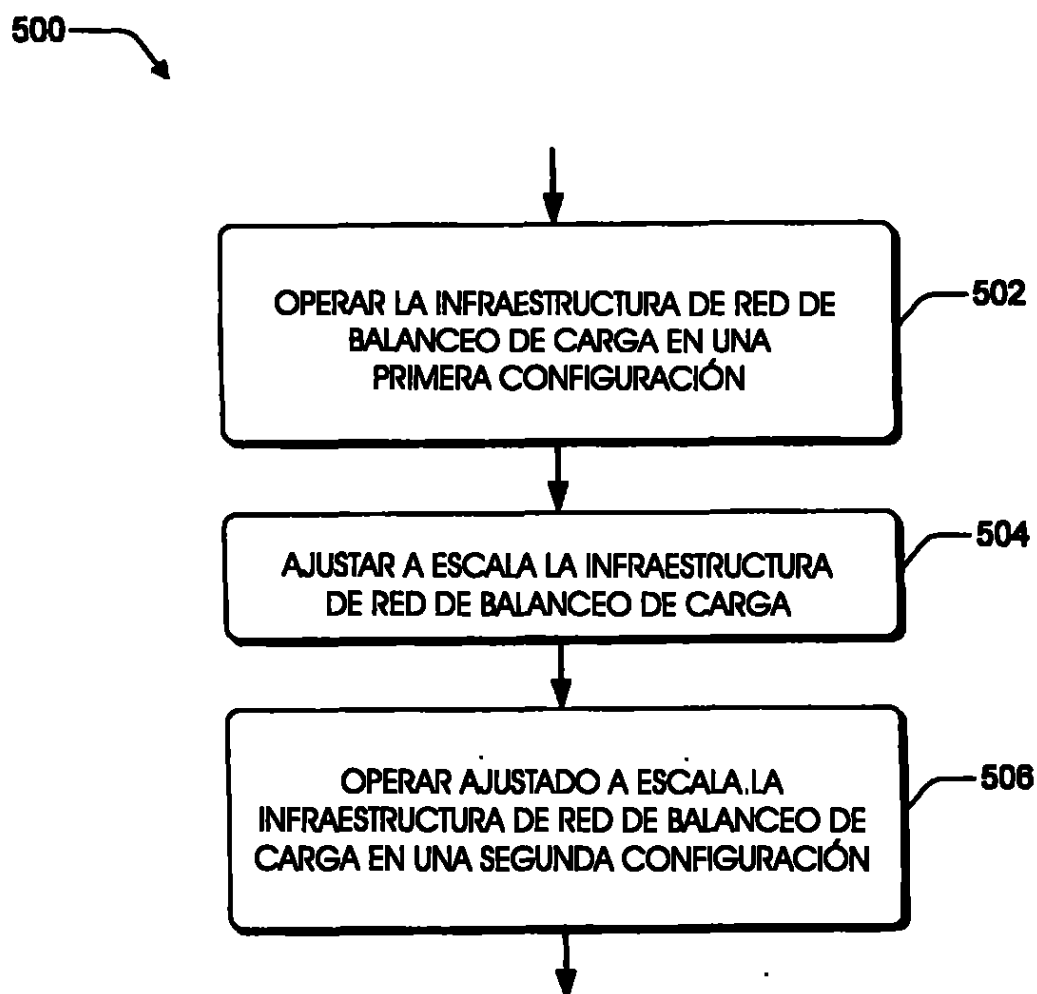


FIG. 5

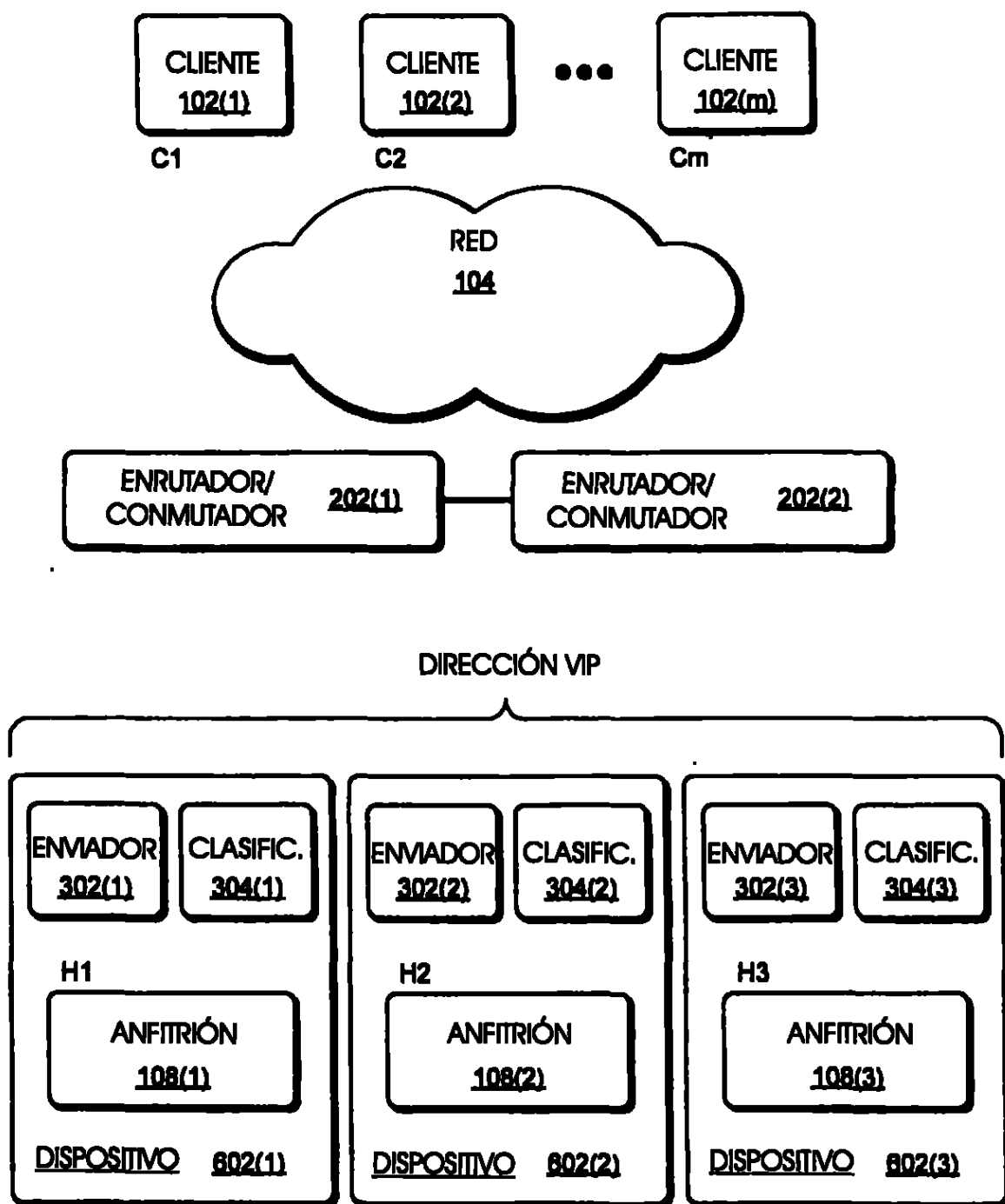


Fig. 6

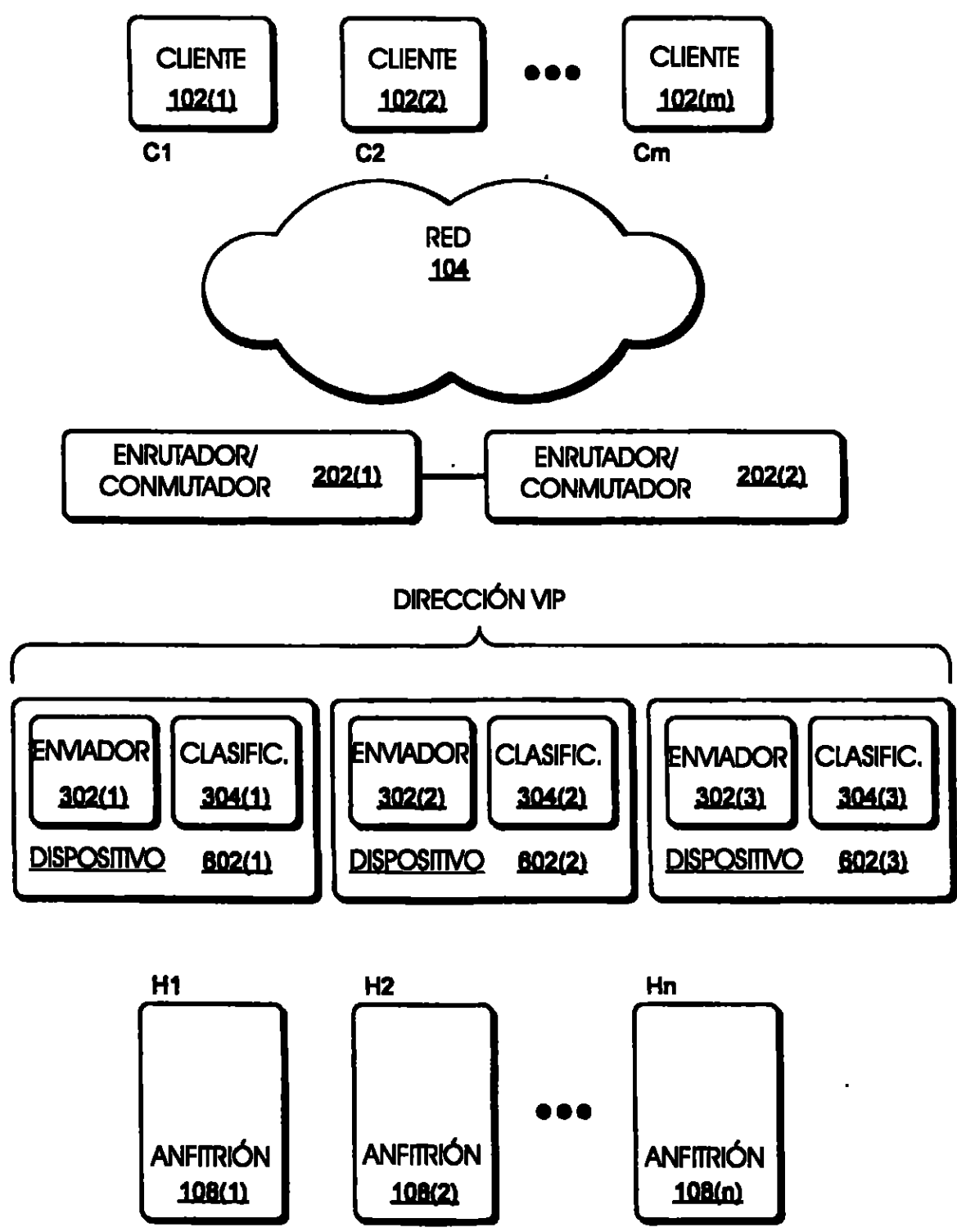


Fig. 7

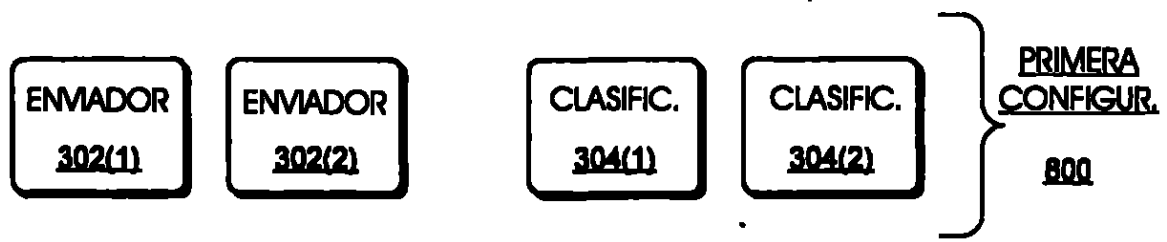


Fig. 8A

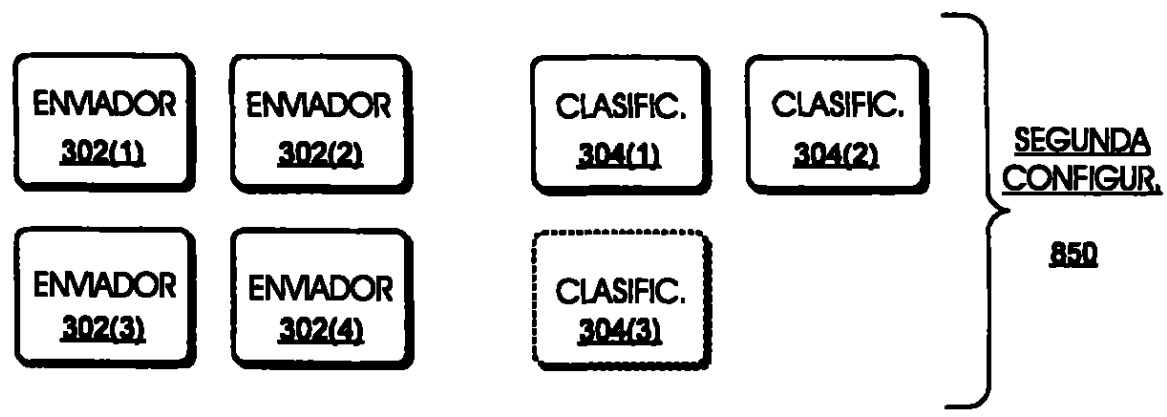


Fig. 8B

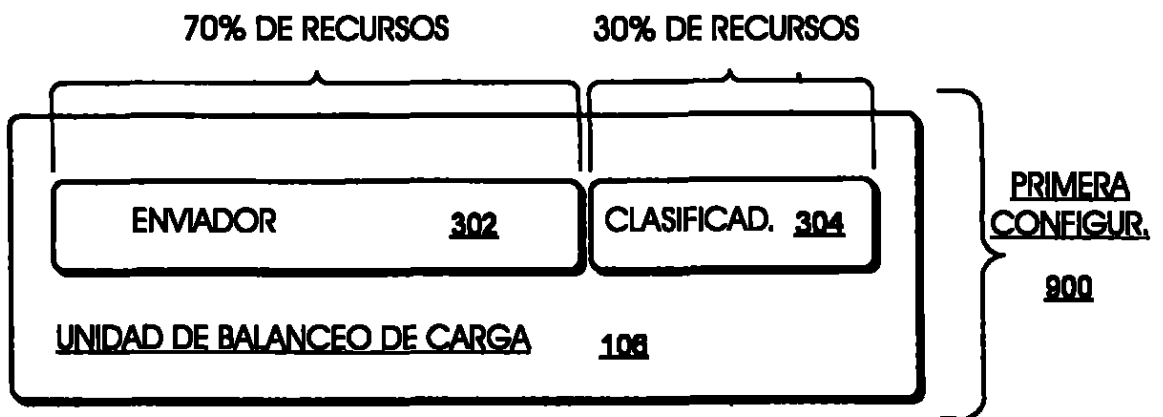


Fig. 9A

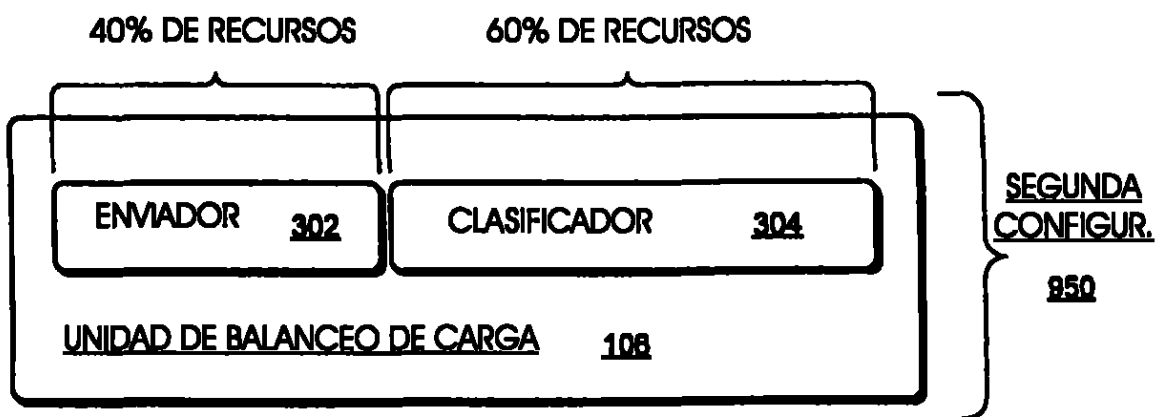


Fig. 9B

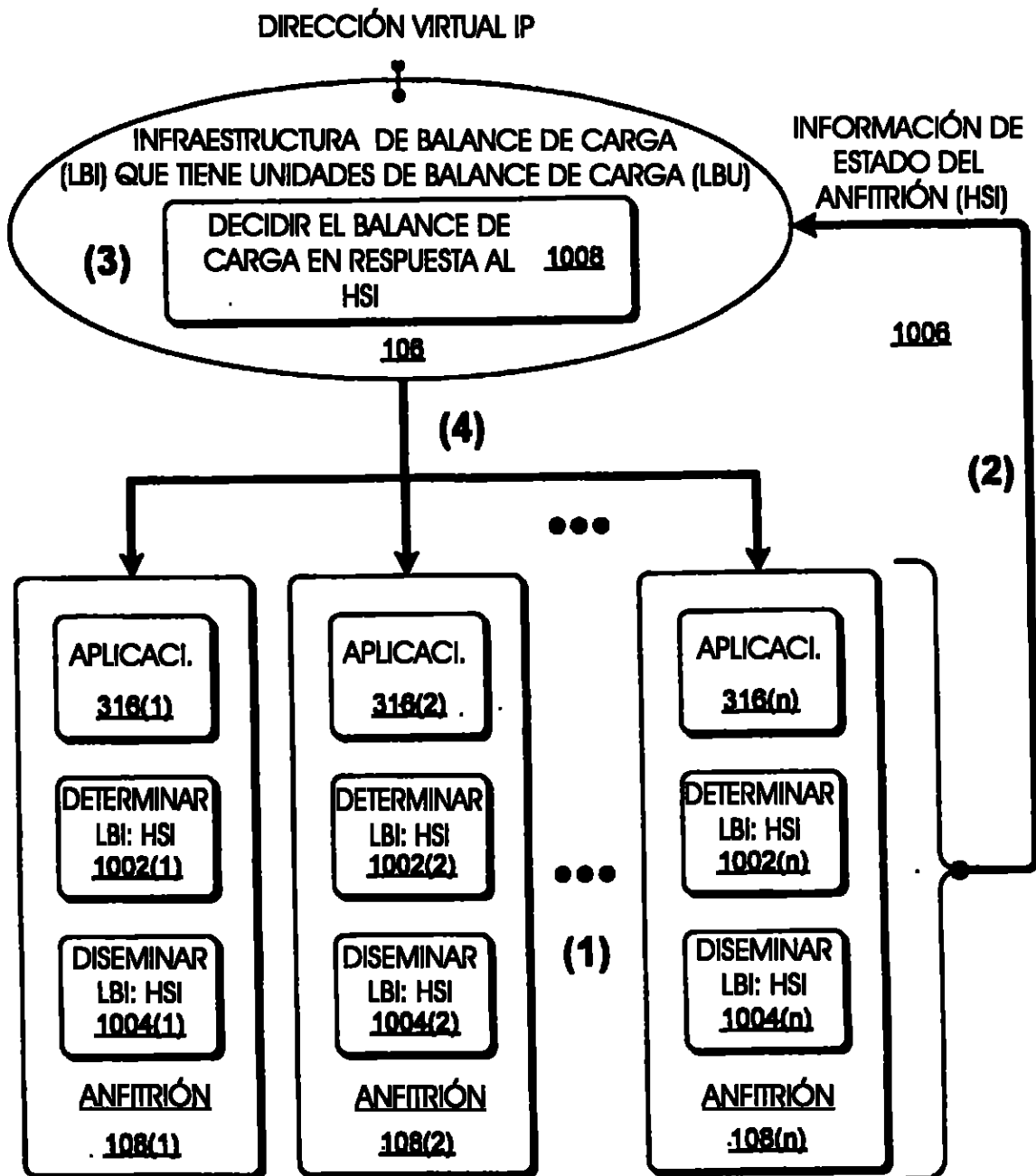


Fig. 10

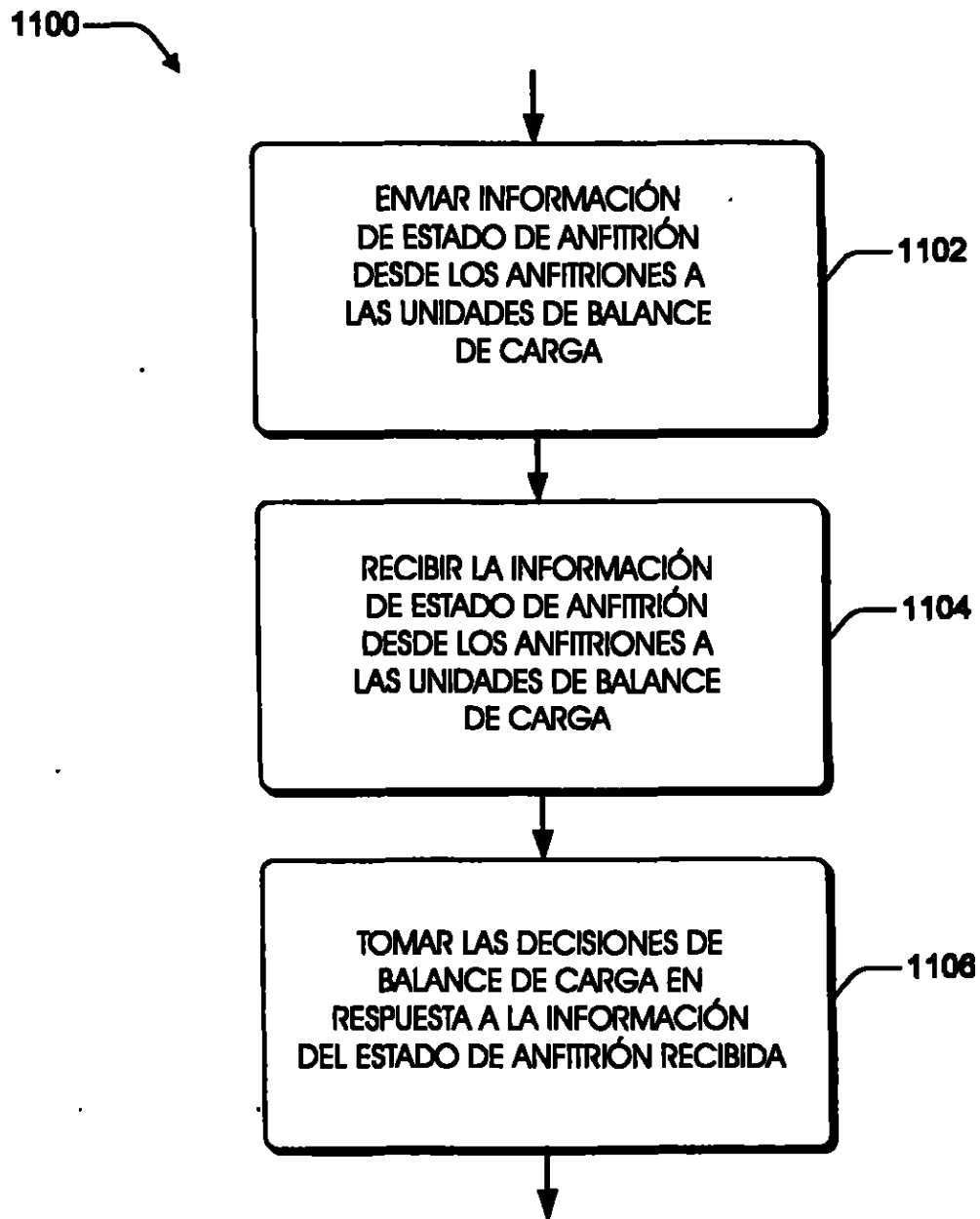


Fig. 11

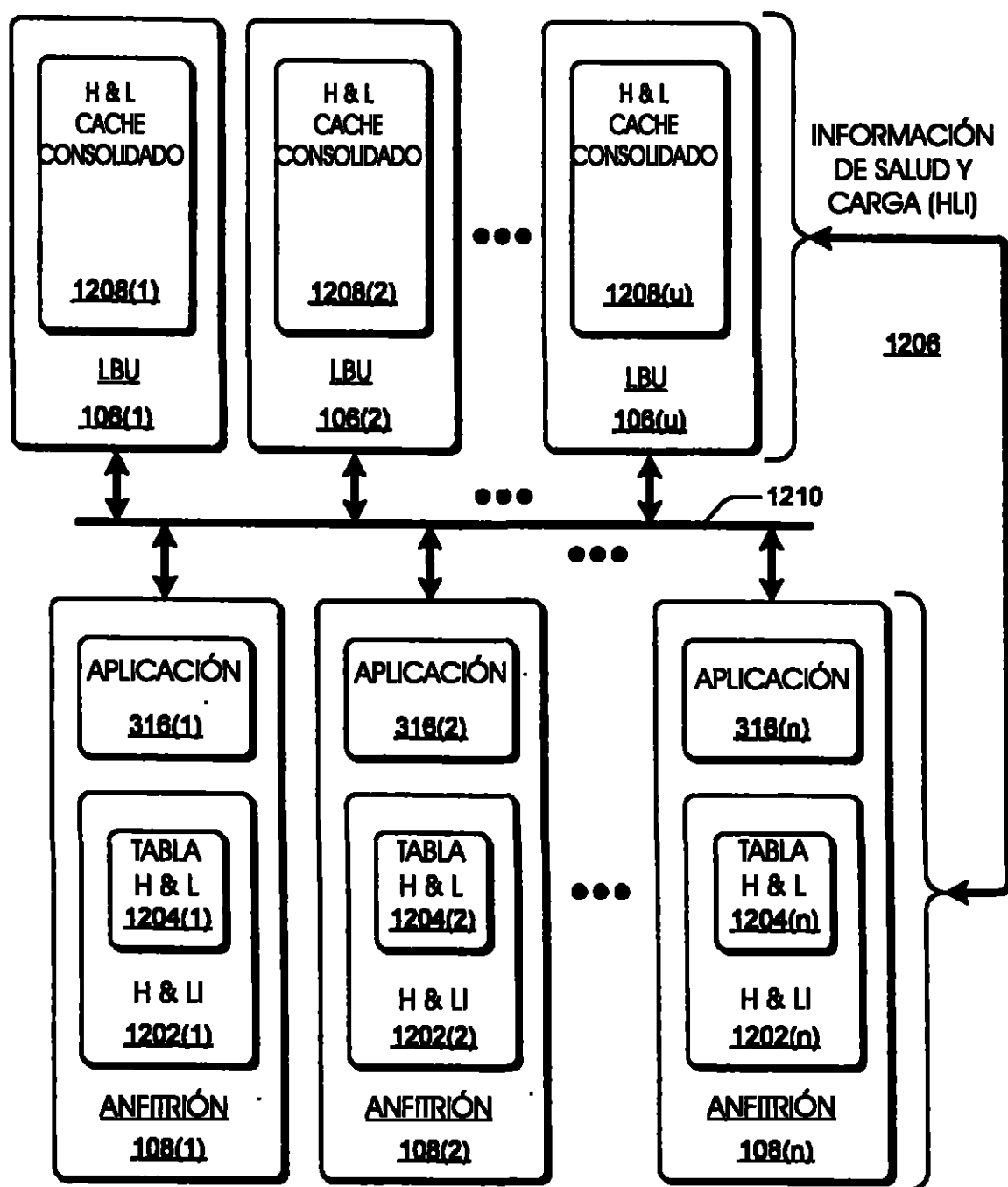


Fig. 12

TABLA H&L			1204
IDENTIFICADOR DE APLICACIÓN (ID) 1302(A)	CARACTERIZACIÓN DEL ESTADO DE LA APLICACIÓN 1302(B)	DIRECTIVA DE BALANCEADOR DE CARGA 1302(C)	
DIRECCIÓN VIRTUAL IP Y PUERTO	SALUD DE LA APLICACIÓN	ESTADO OBJETIVO DE BALANCEO DE CARGA	
DIRECCIÓN FÍSICA IP Y PUERTO			
PROTOCOLO	CARGA DE LA APLICACIÓN	ESTADO ACTUAL DE BALANCEO DE CARGA	
INFORMACIÓN ESPECÍFICA DE PROTOCOLO	CAPACIDAD DE LA APLICACIÓN		

Fig. 13A

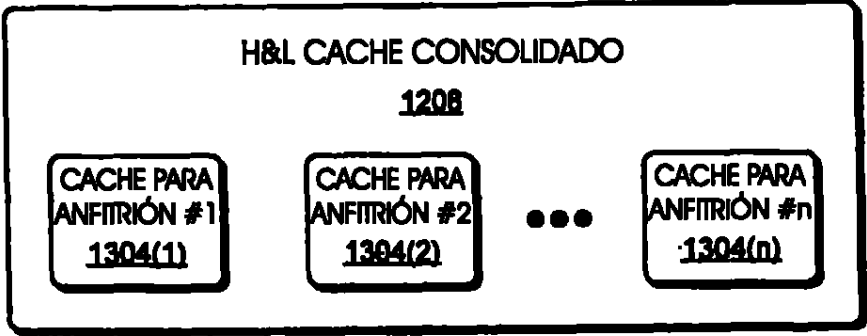


Fig. 13B

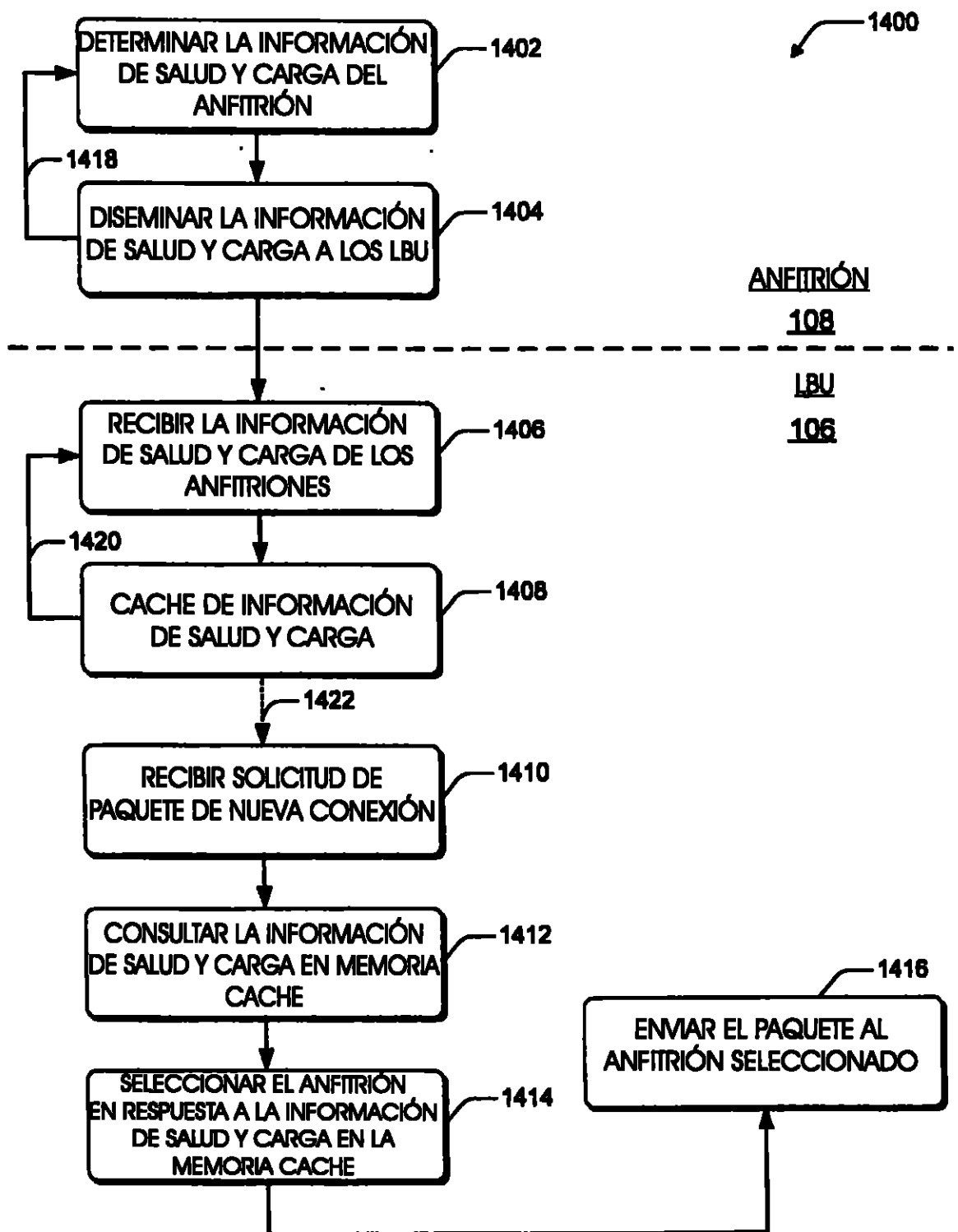


Fig. 14

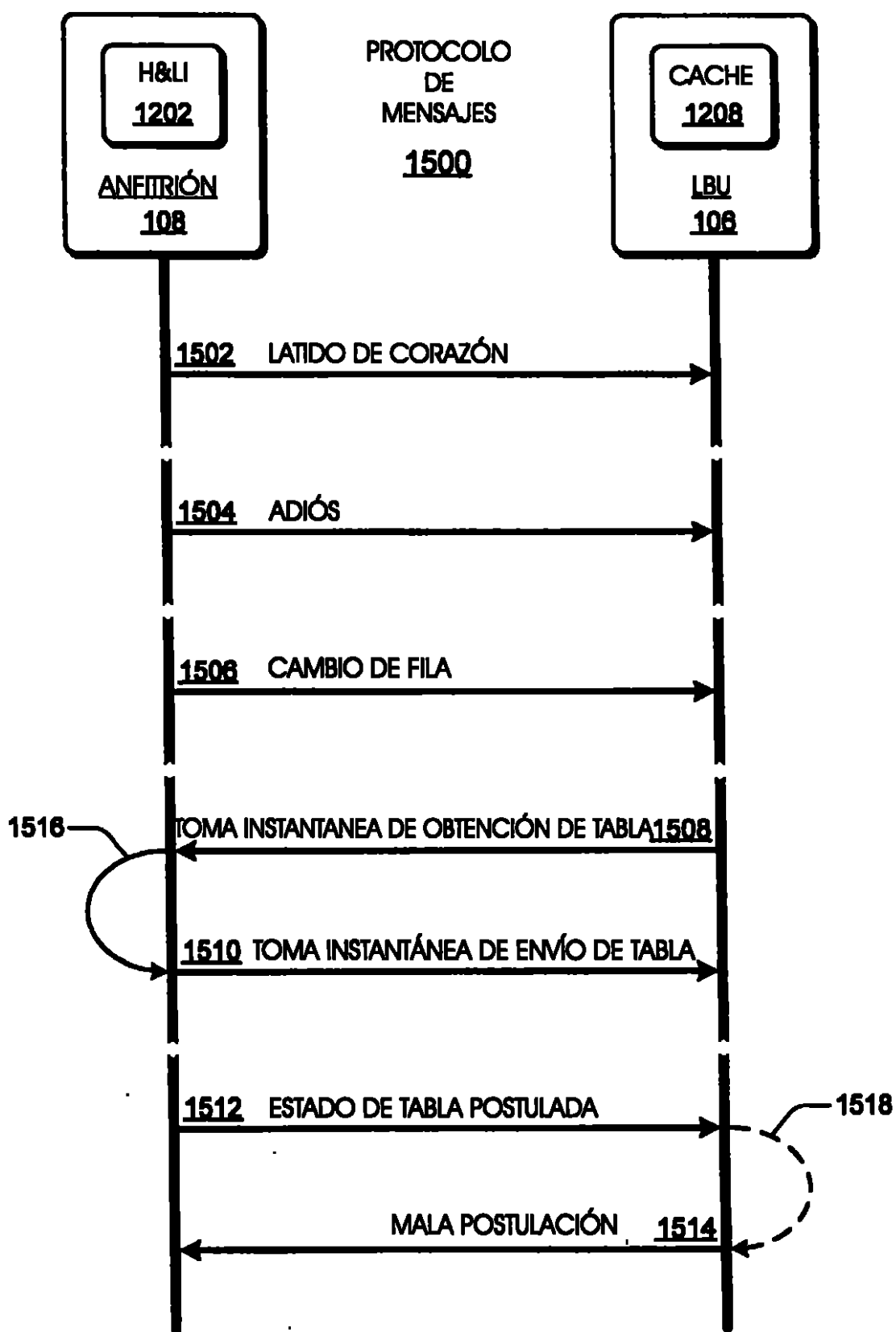


Fig. 15

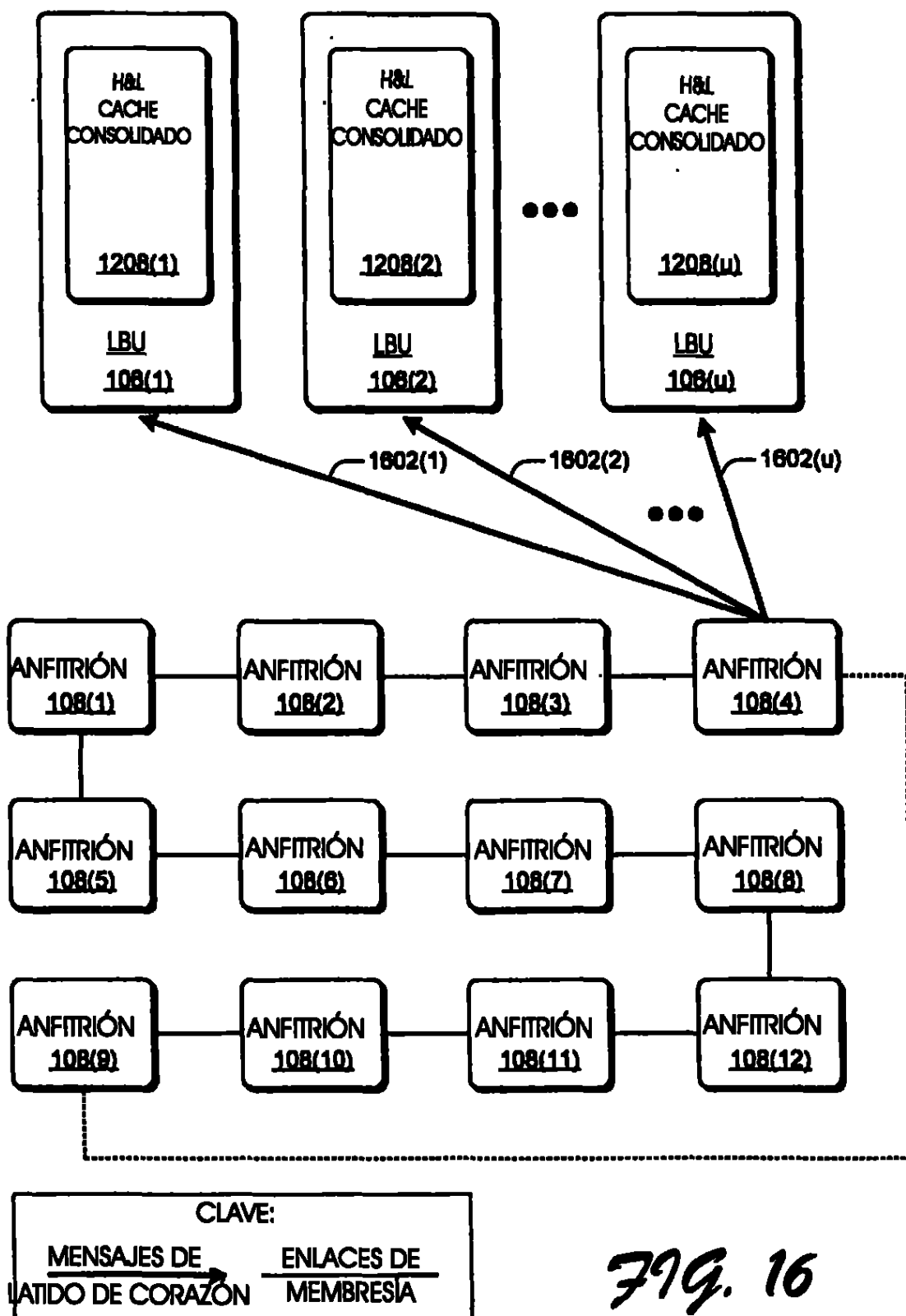


Fig. 16

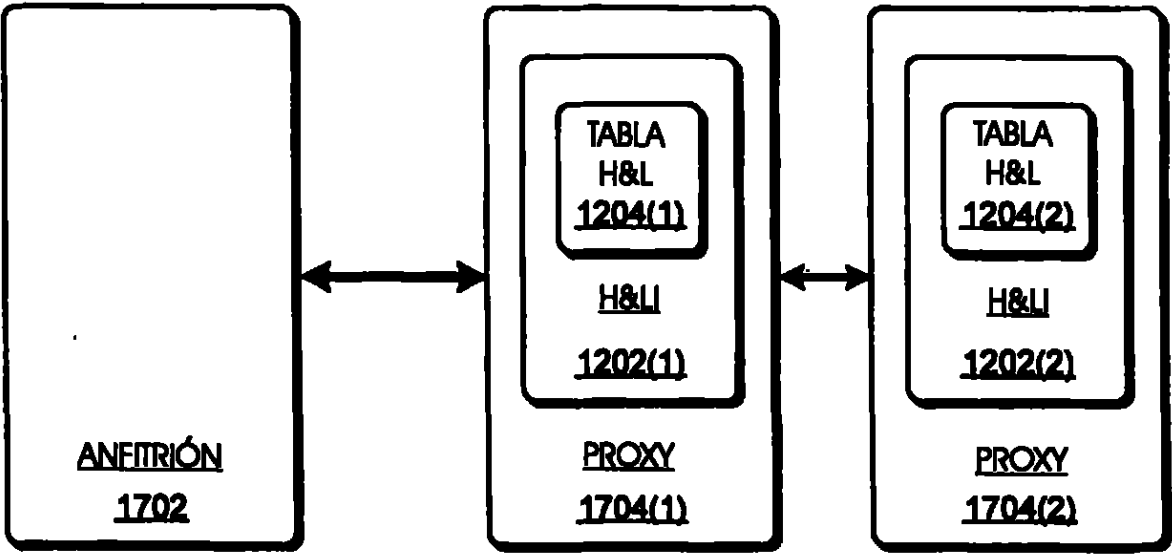


Fig. 17A

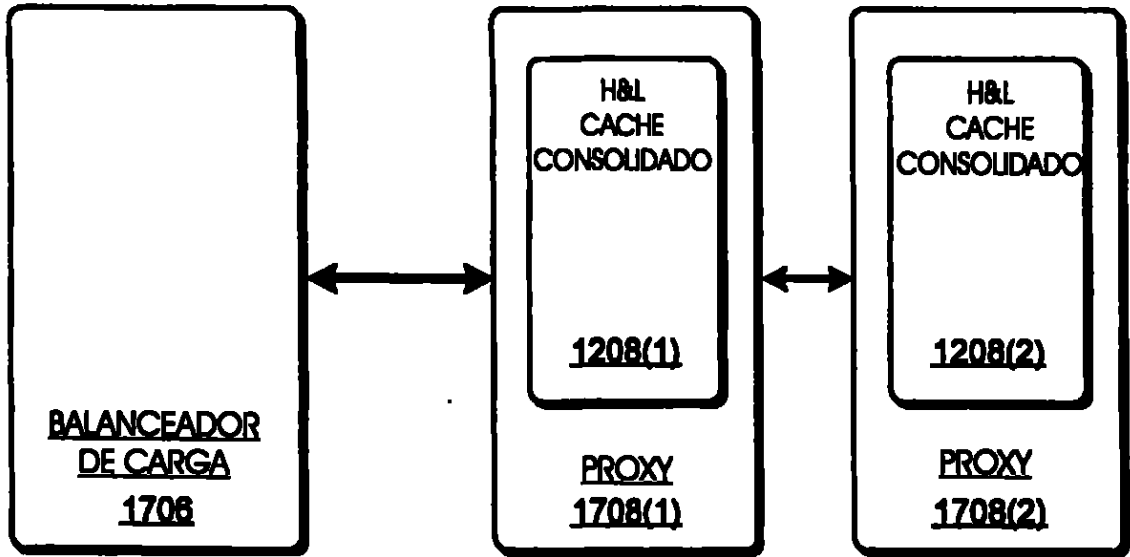


Fig. 17B

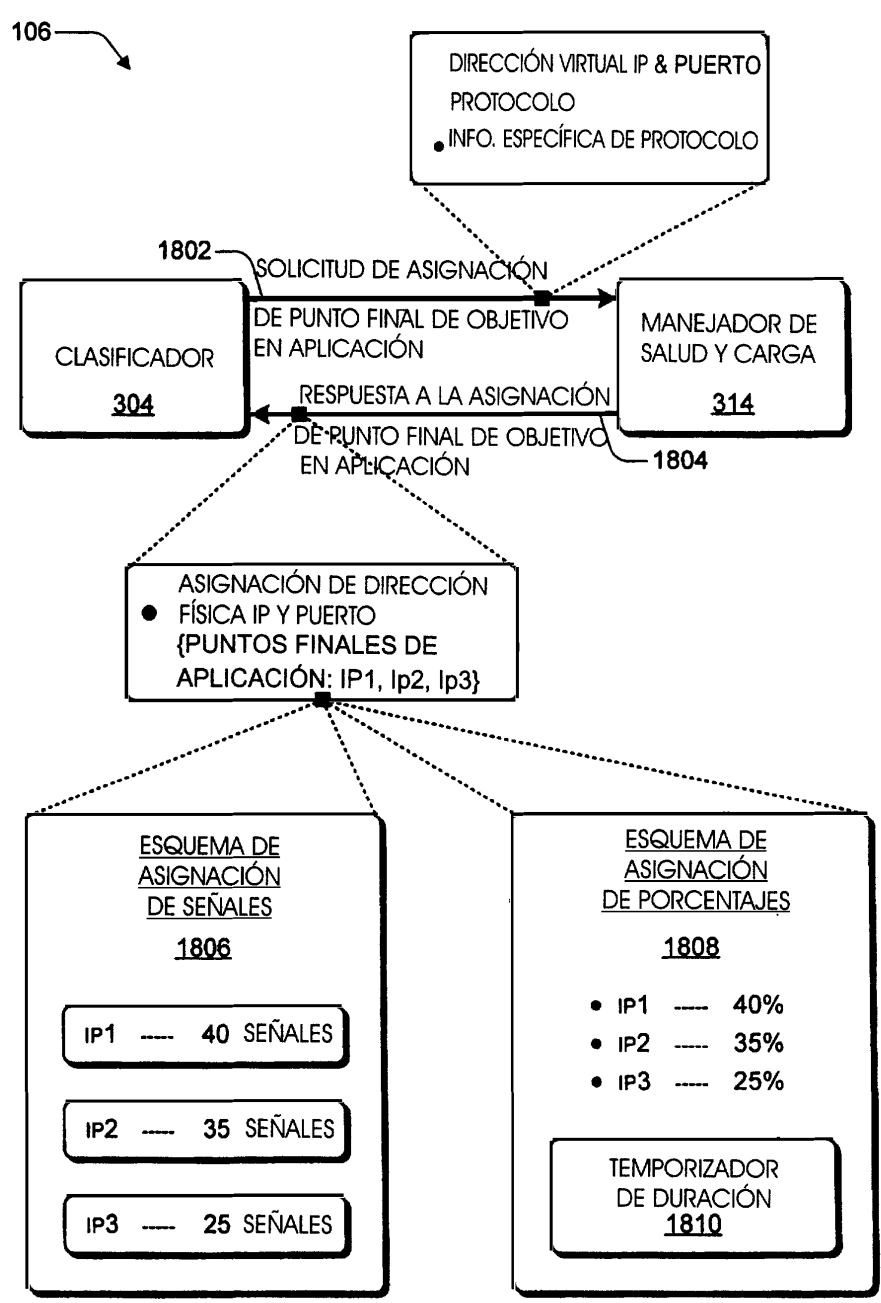


Fig. 18

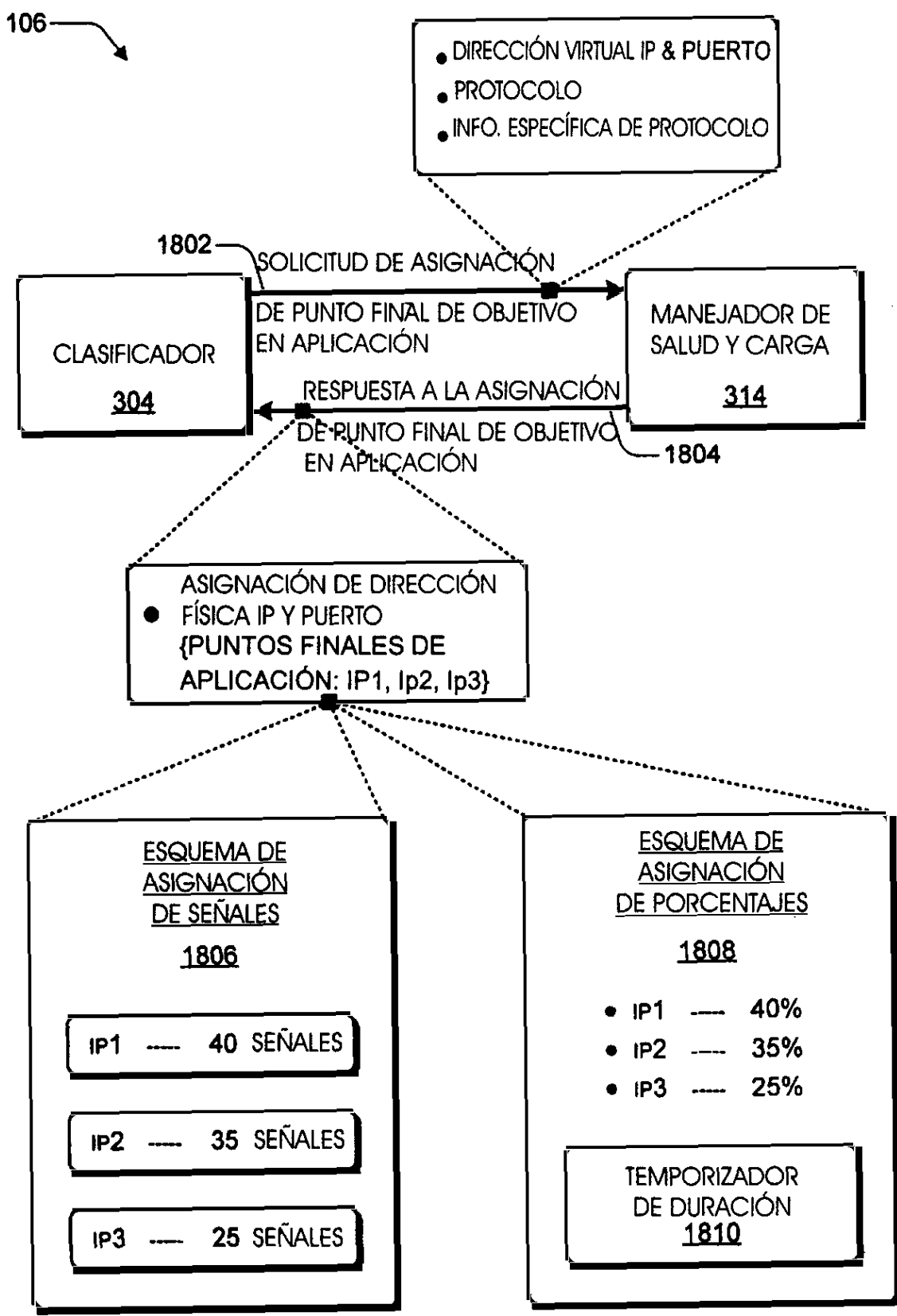


Fig. 18

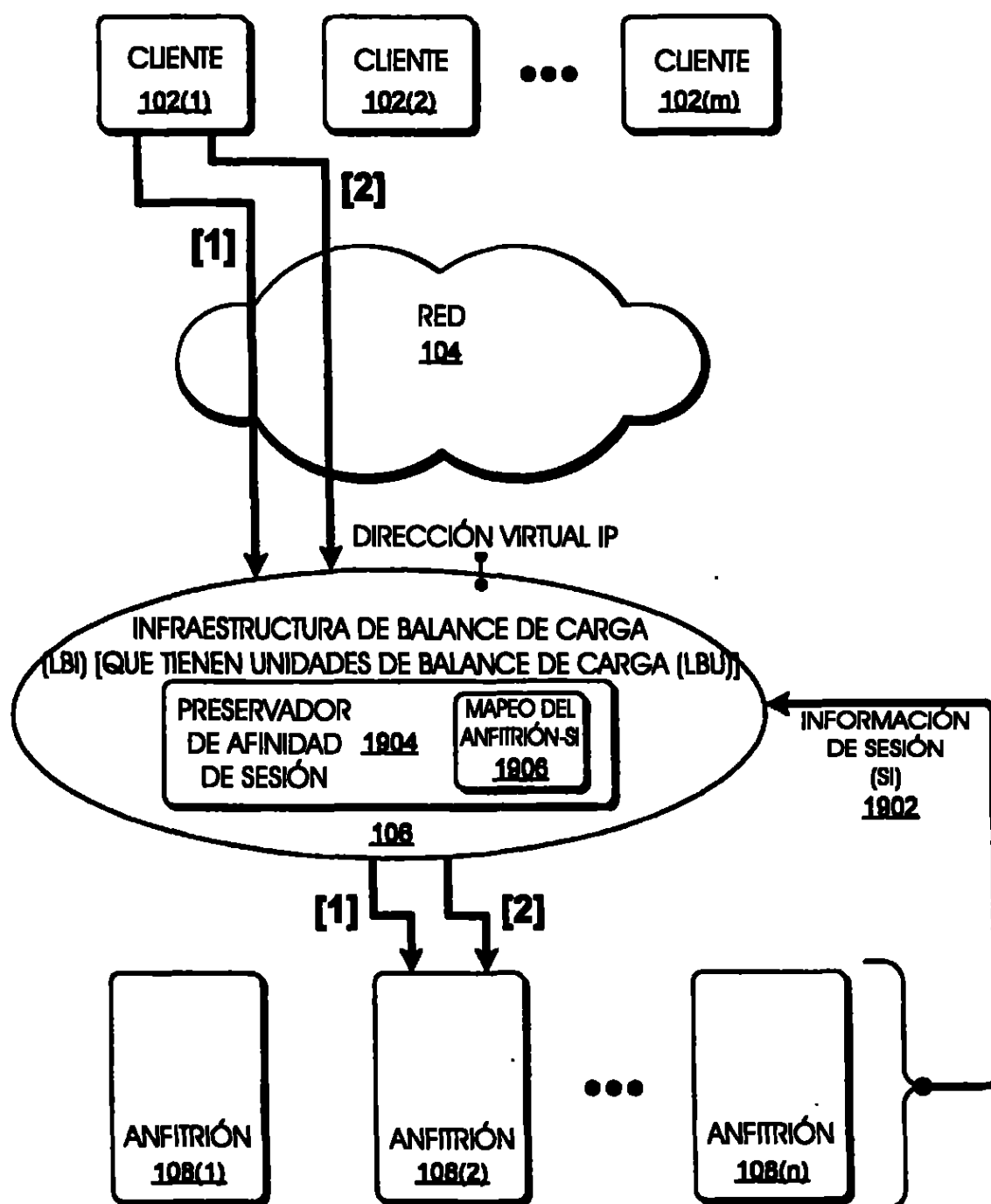


Fig. 19

486

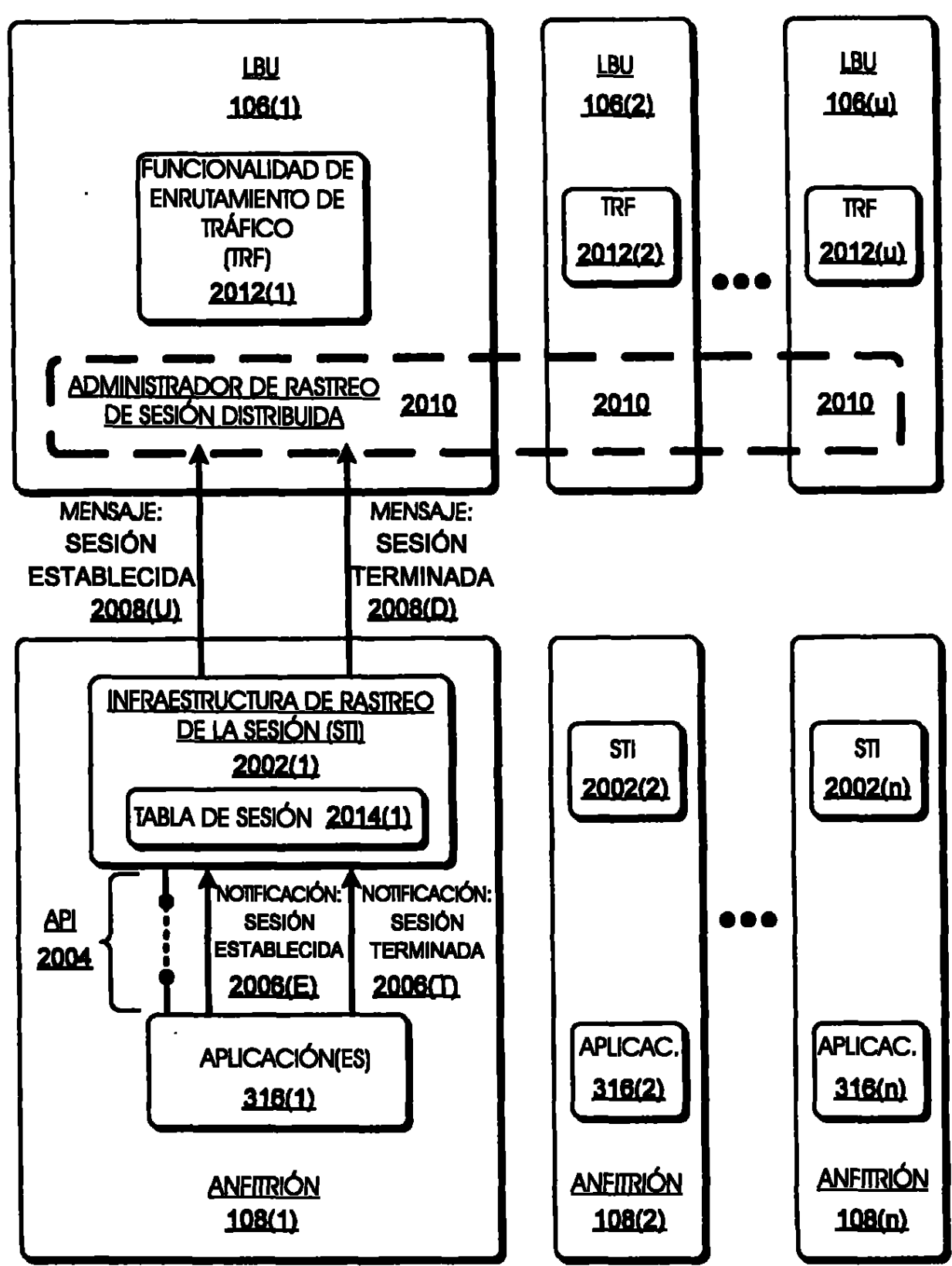


Fig. 20

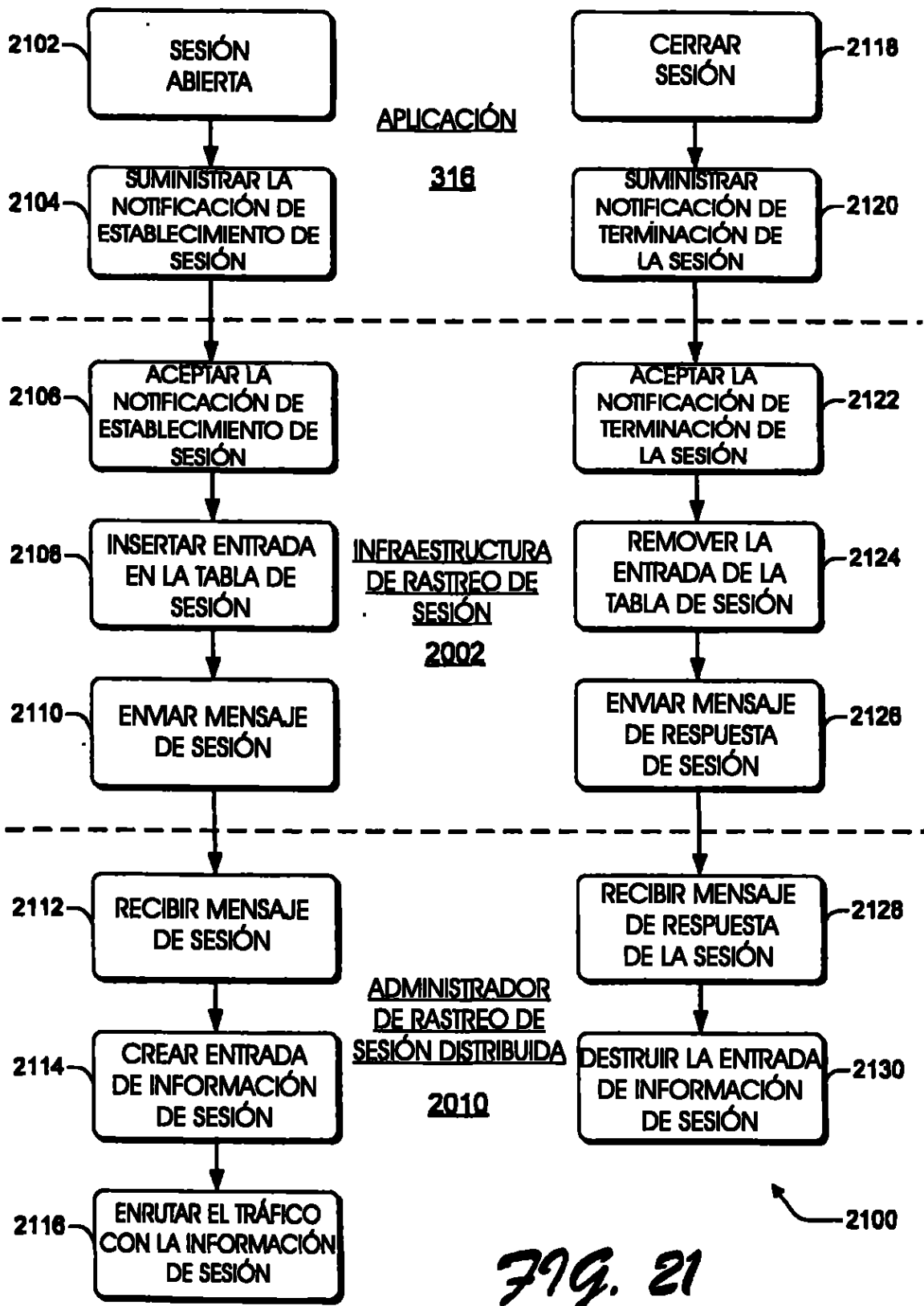


Fig. 21

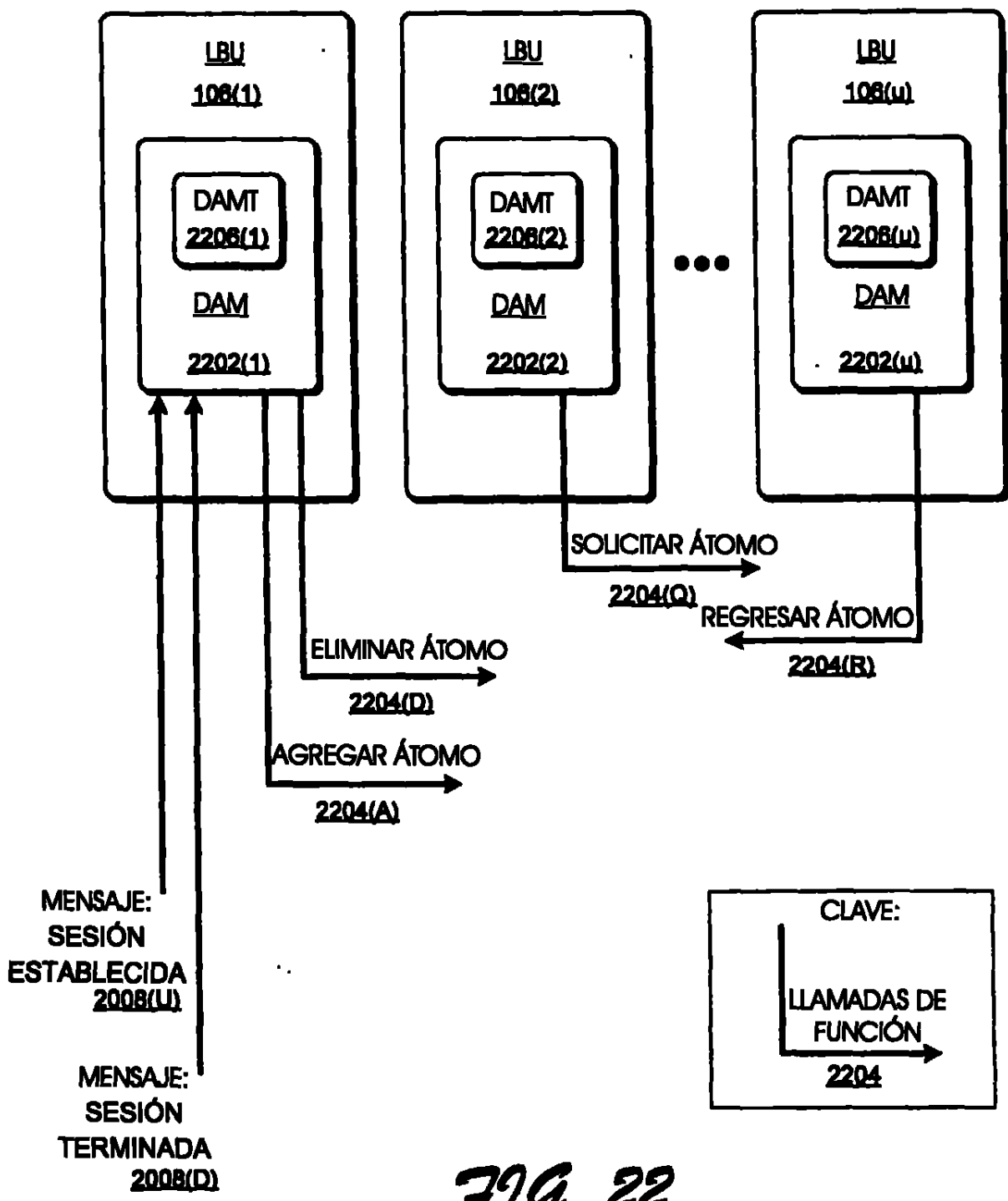


Fig. 22

TABLA DE SESIÓN		2014
2302(1)	DENTIFICADOR DE SESIÓN 2302(1I)	TIPO DE SESIÓN/APLICAC. 2302(1T)
2302(2)	DENTIFICADOR DE SESIÓN 2302(2I)	TIPO DE SESIÓN/APLICAC. 2302(2T)
⋮	⋮	
2302(v)	IDENTIFICADOR DE SESIÓN 2302(vI)	TIPO DE SESIÓN/APLICAC. 2302(vT)

Fig. 23A

TABLA DE ADMINISTRADOR DE ÁTOMO DISTRIBUIDO (DAM) (DAMT) 2206			
2304(1)	CLAVE 2304(1K)	DATOS 2304(1D)	METADATOS 2304(1M)
2304(2)	CLAVE 2304(2K)	DATOS 2304(2D)	METADATOS 2304(2M)
⋮	⋮	⋮	
2304(w)	CLAVE 2304(wK)	DATOS 2304(wD)	METADATOS 2304(wM)

Fig. 23B

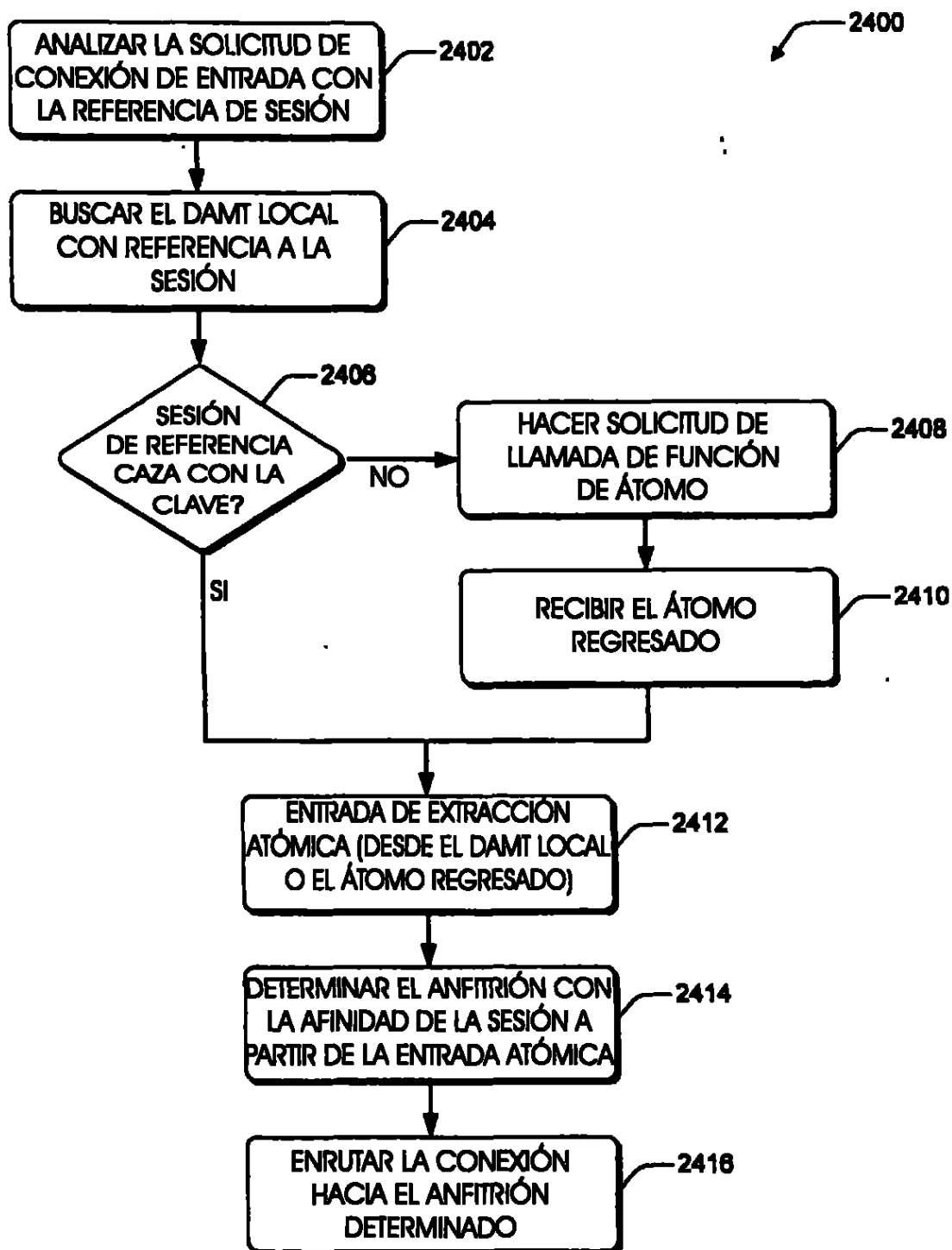


Fig. 24

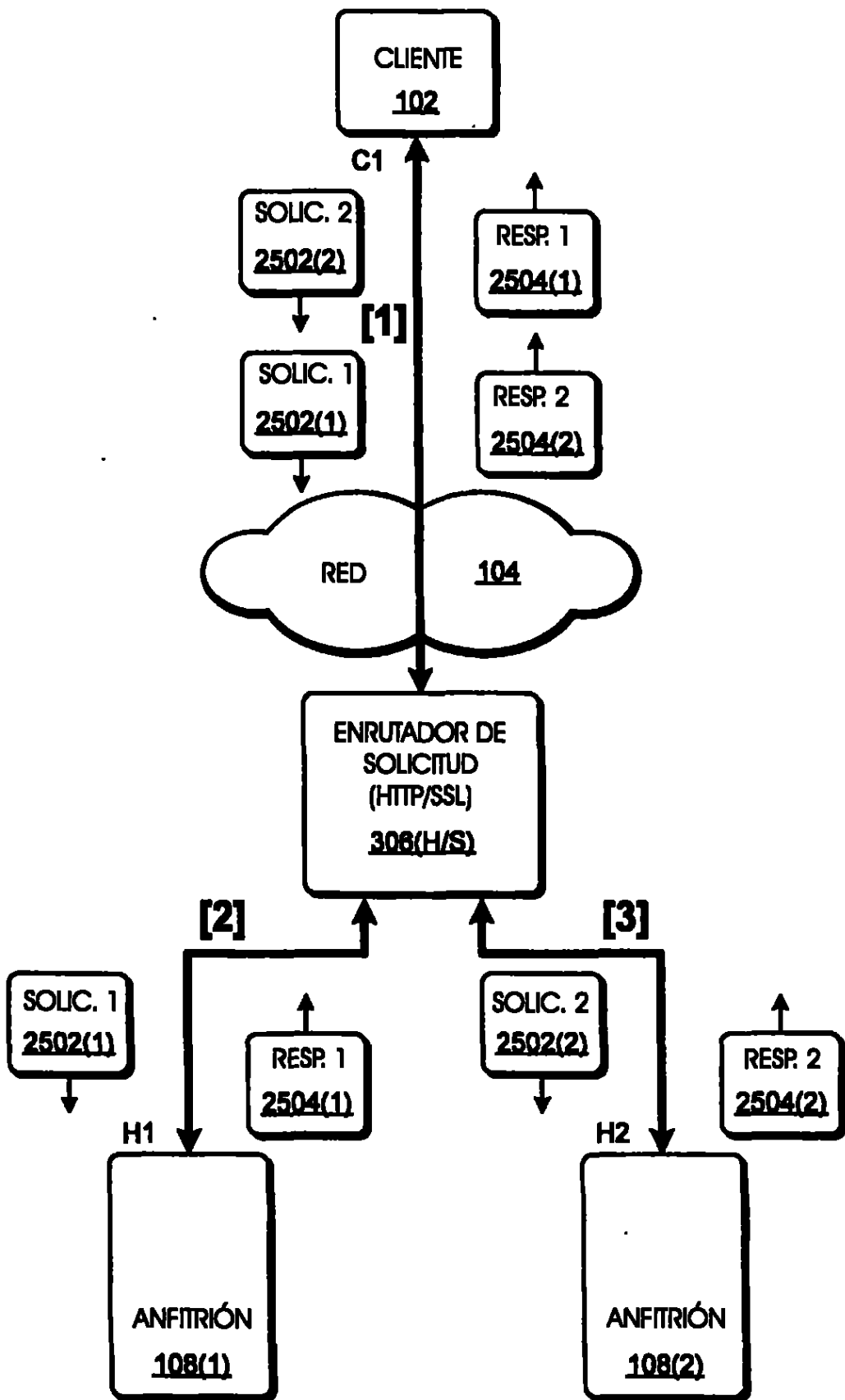


FIG. 25

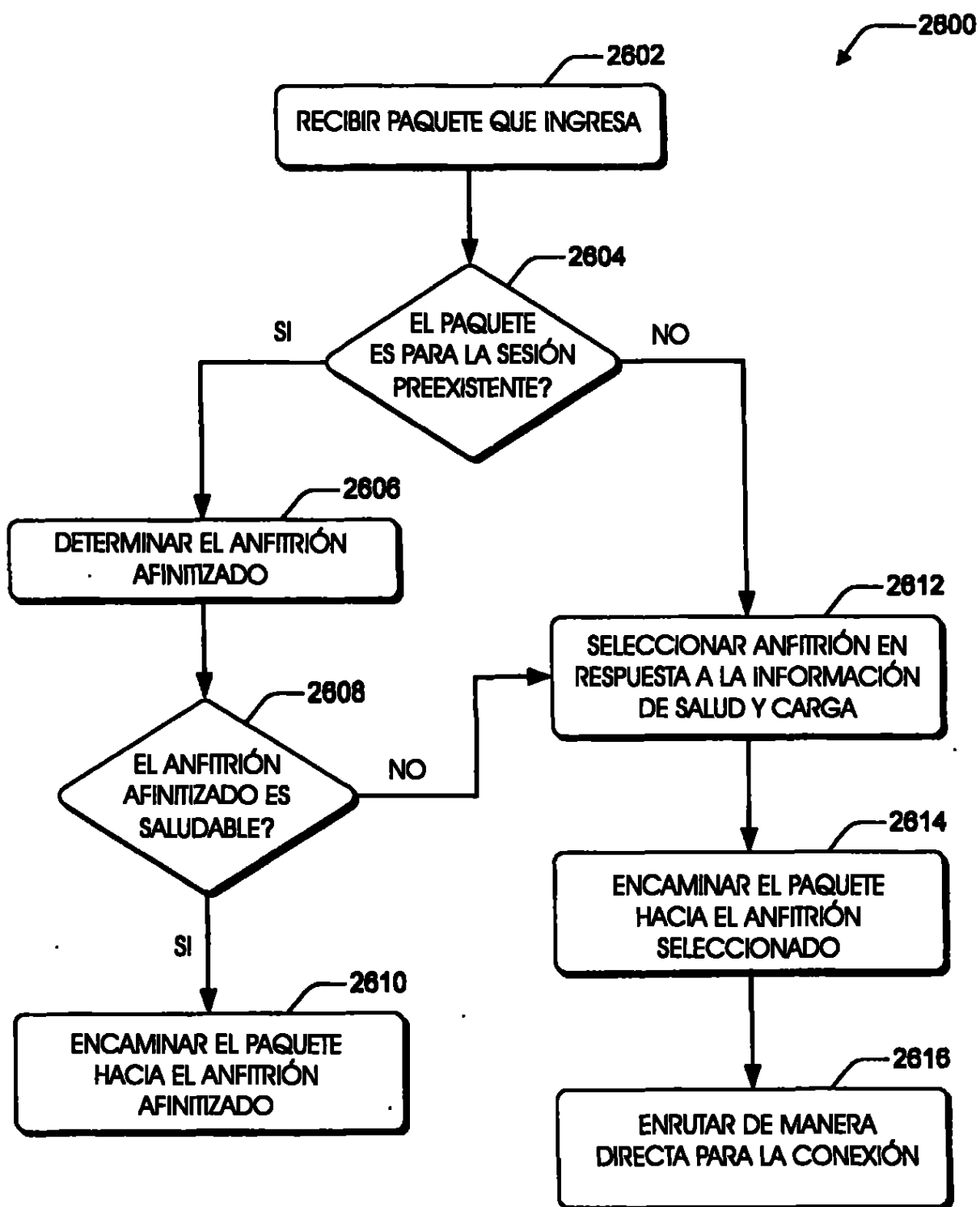


Fig. 26

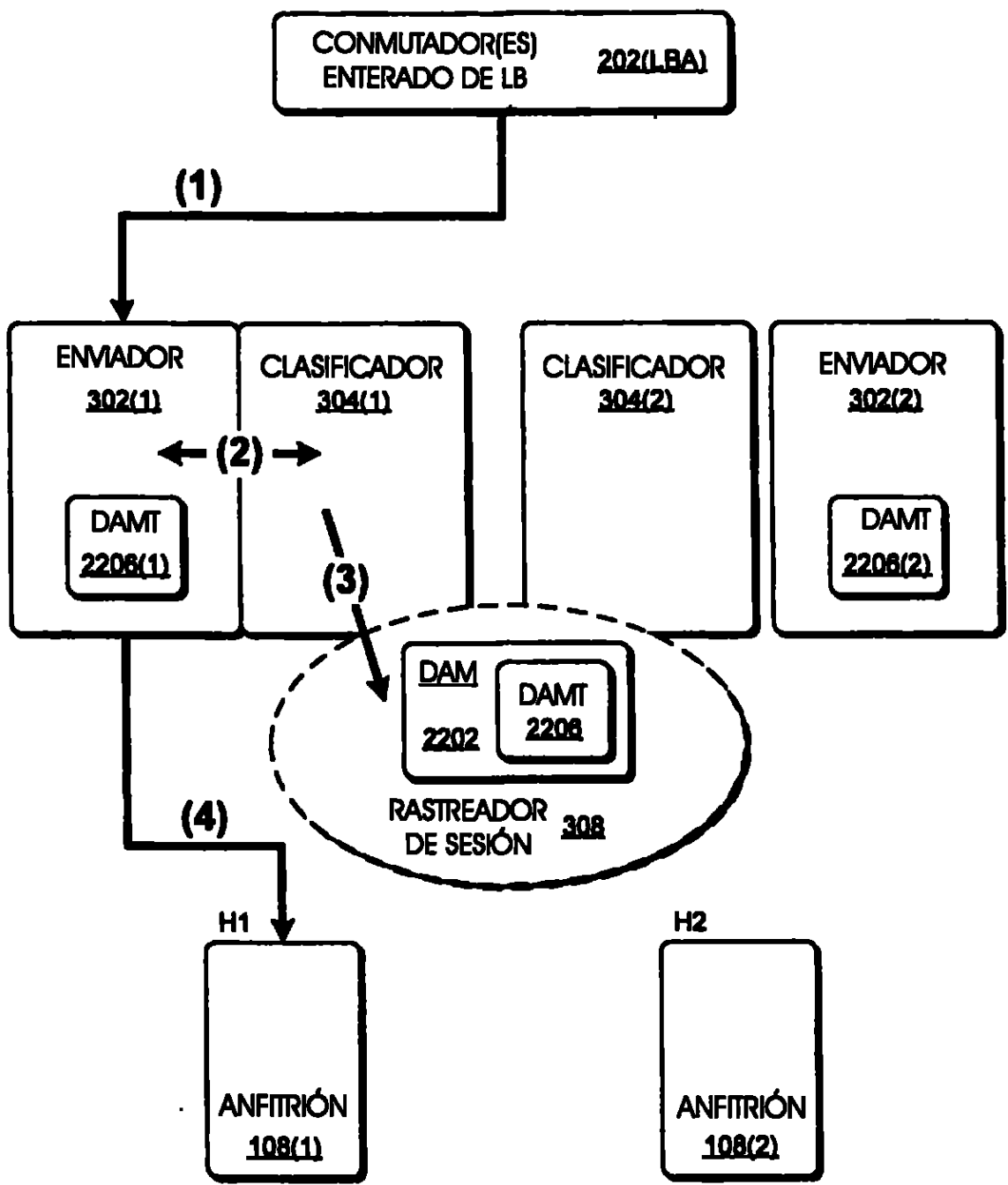


Fig. 27

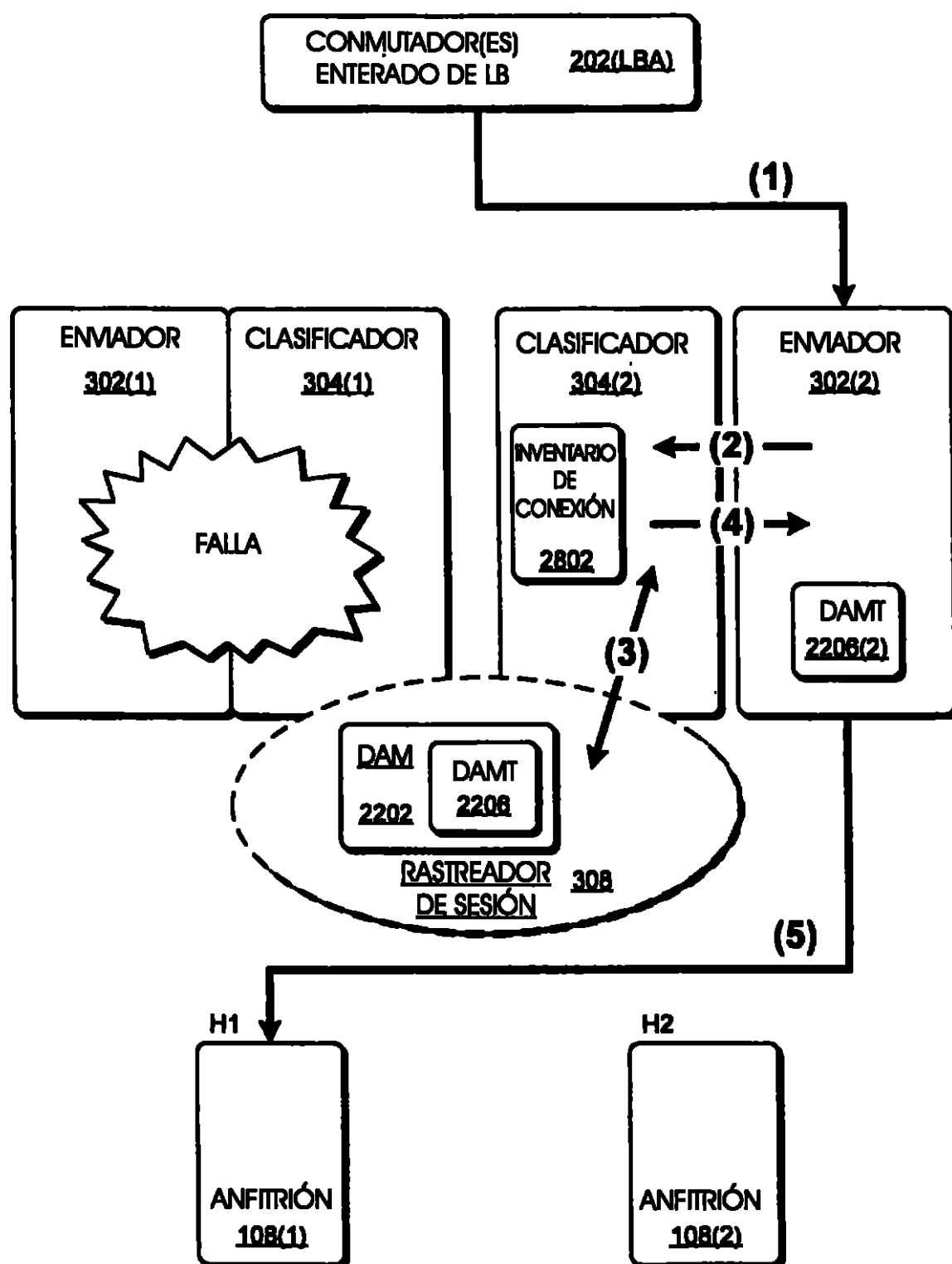


Fig. 28

+

~~283~~

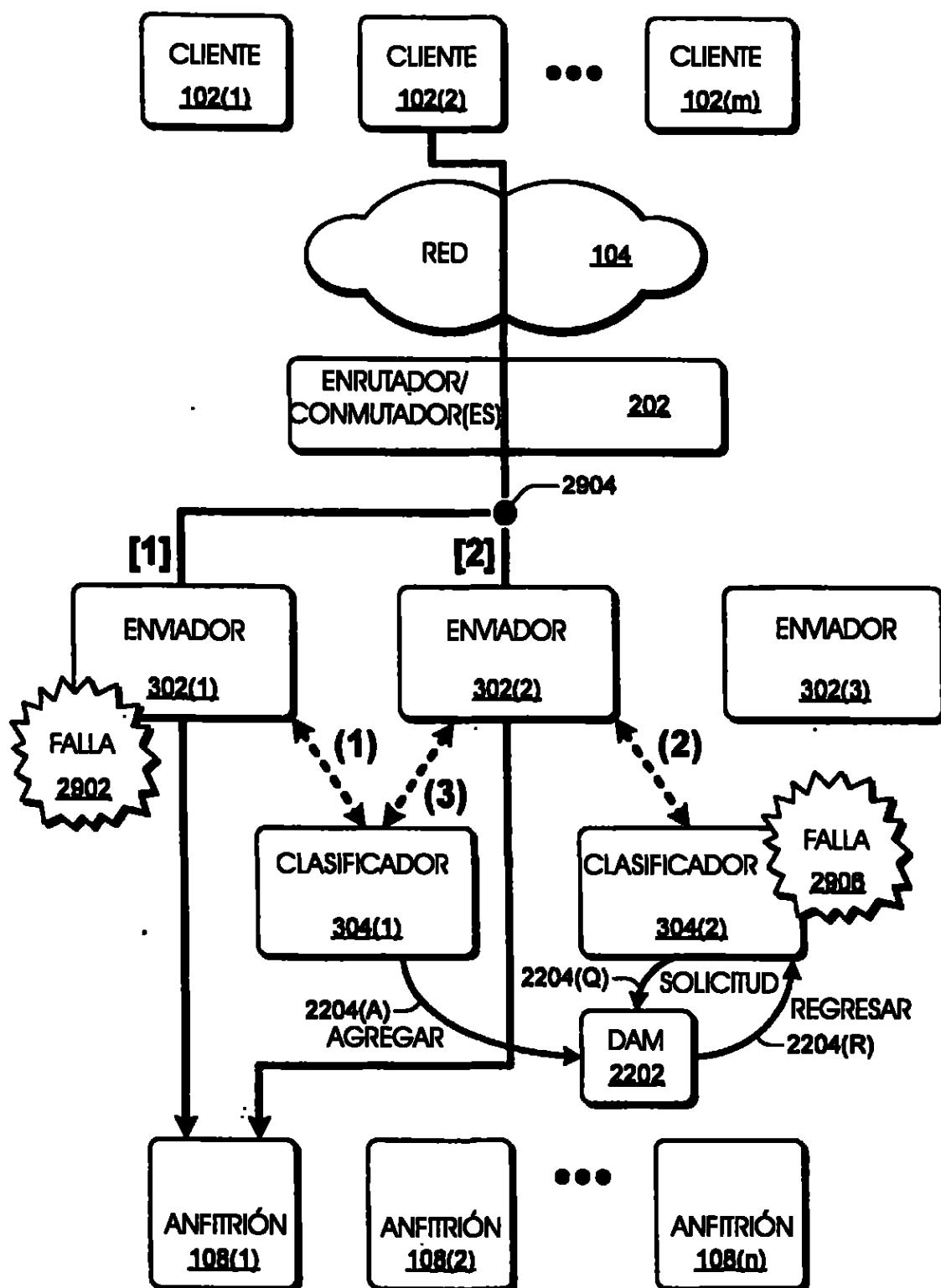


Fig. 29

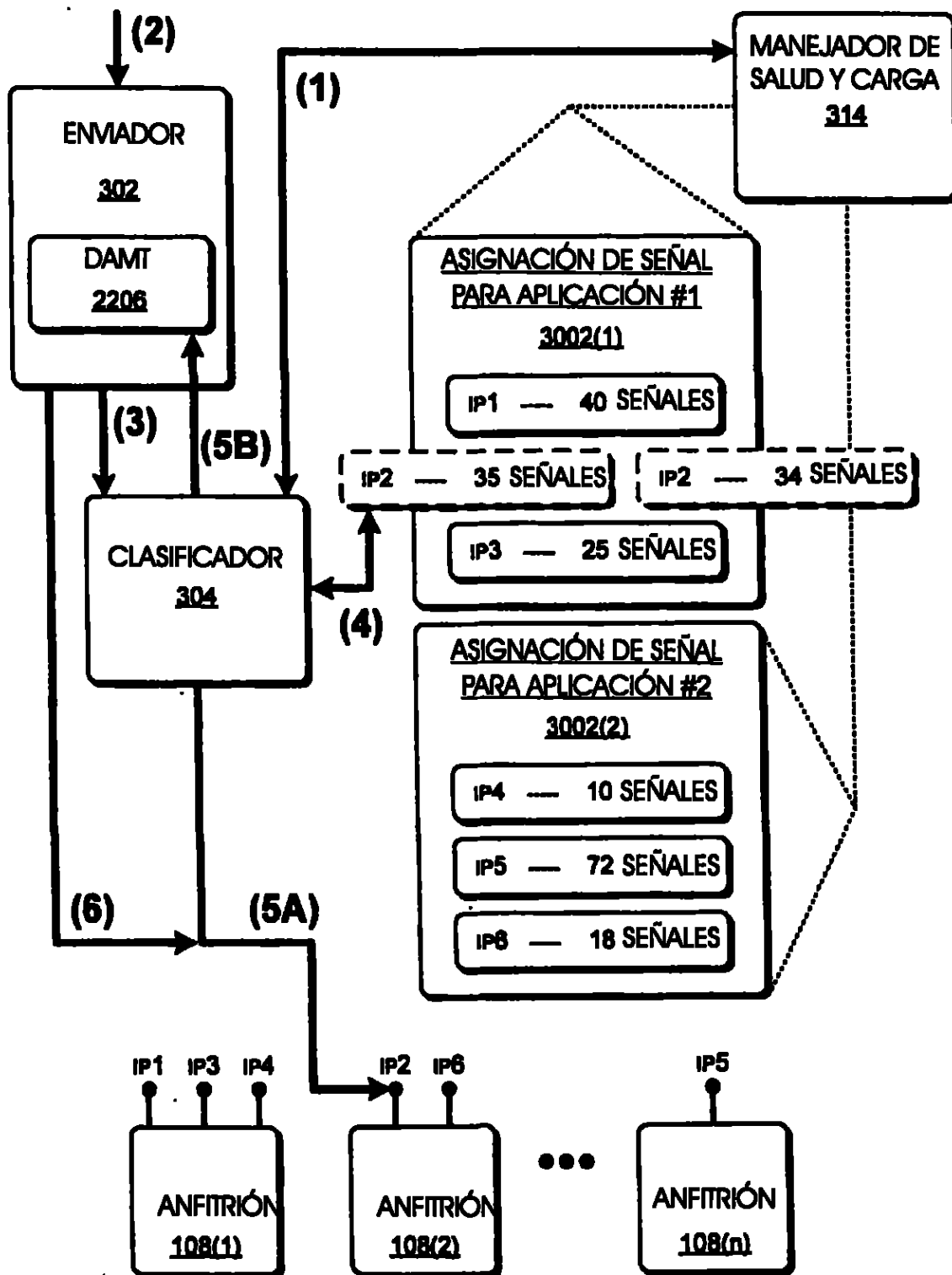


Fig. 30

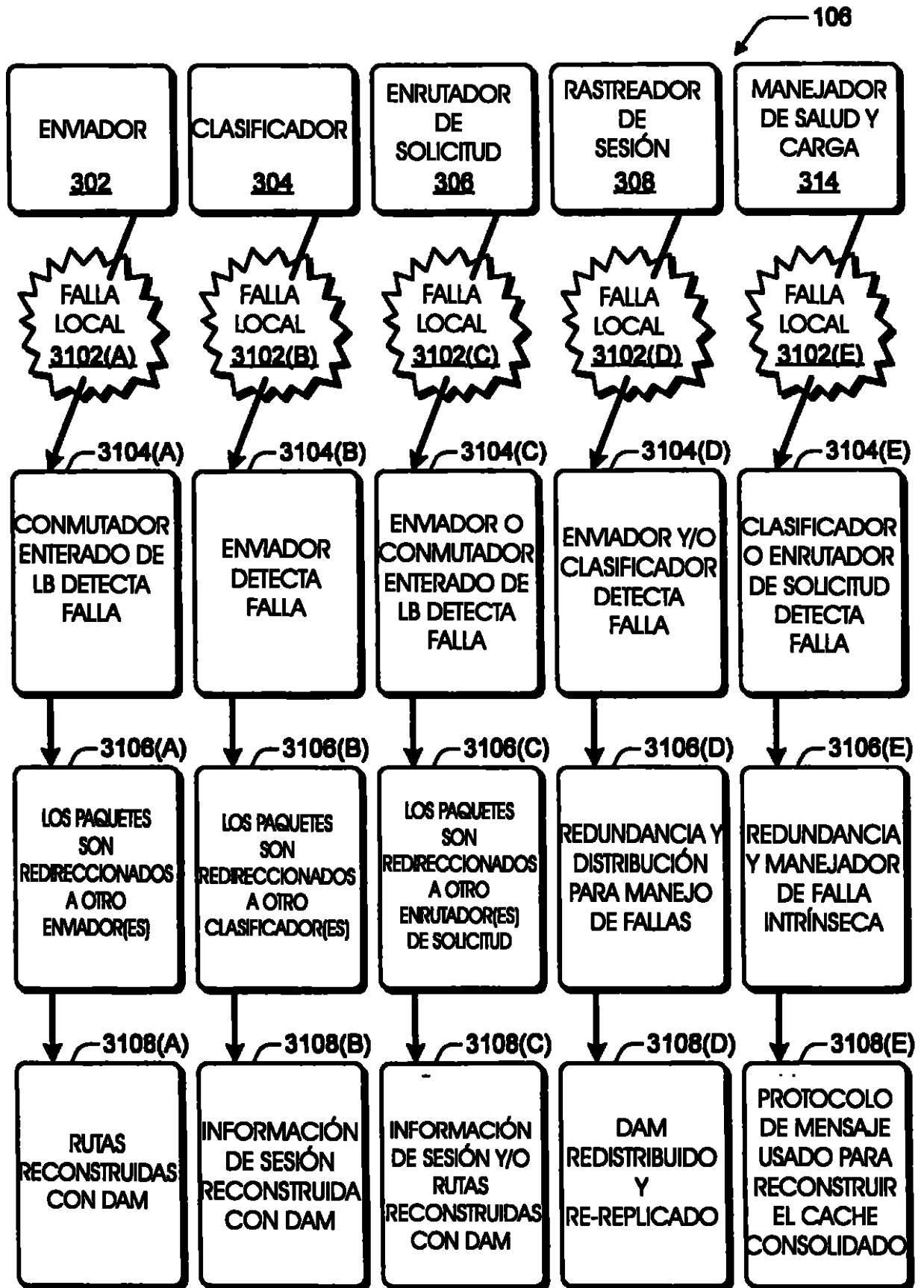


Fig. 31

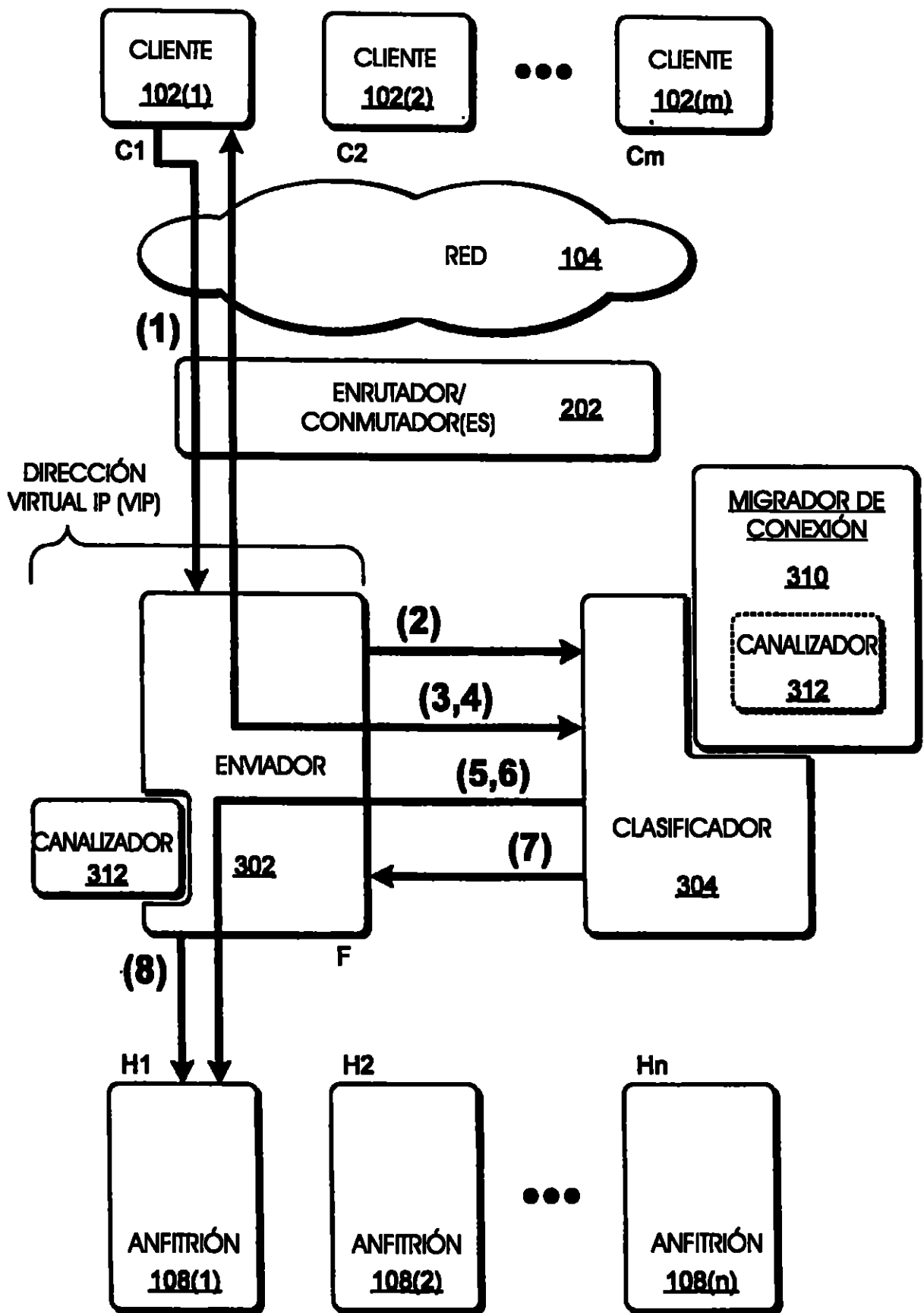


Fig. 32

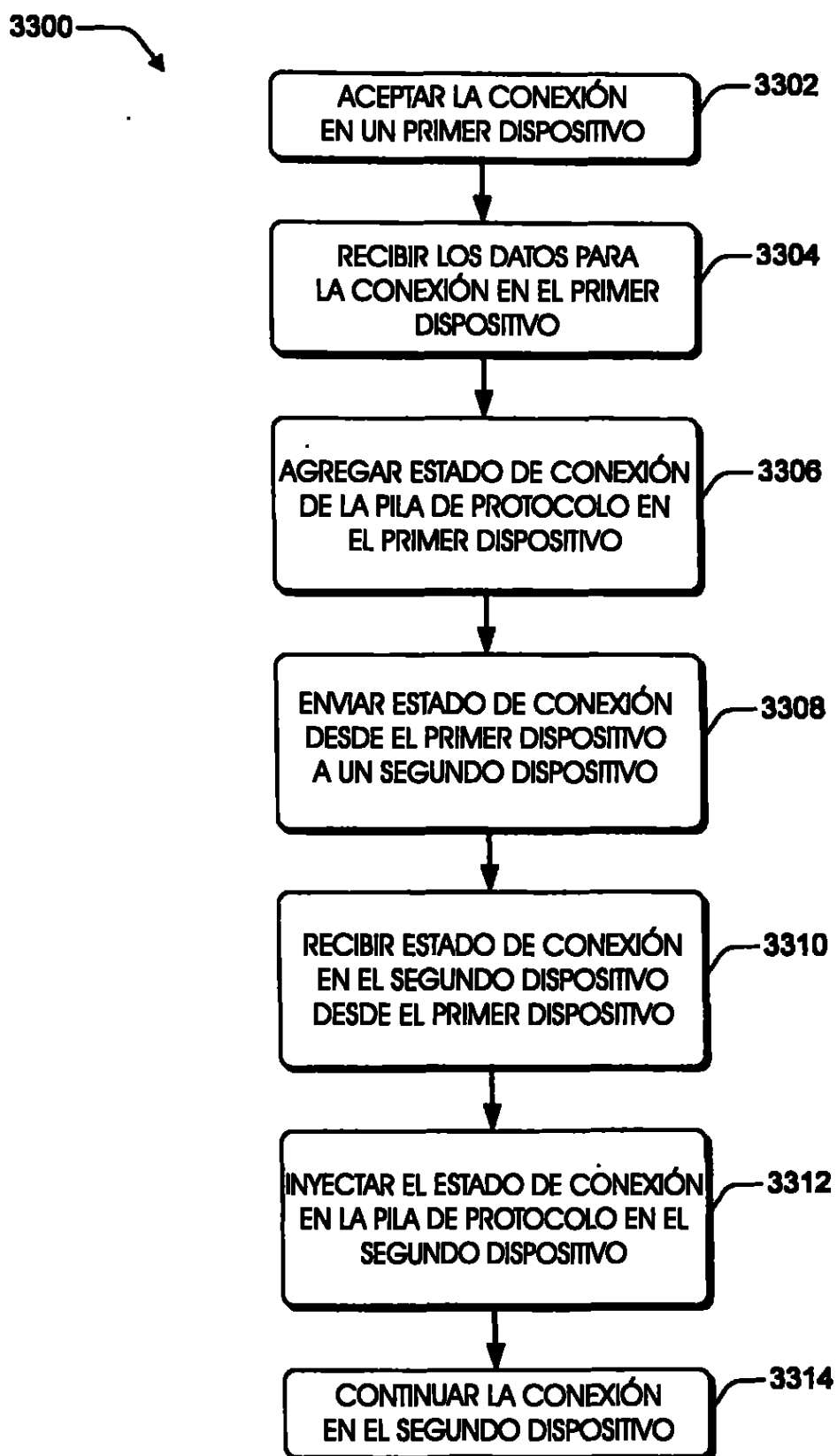


Fig. 33

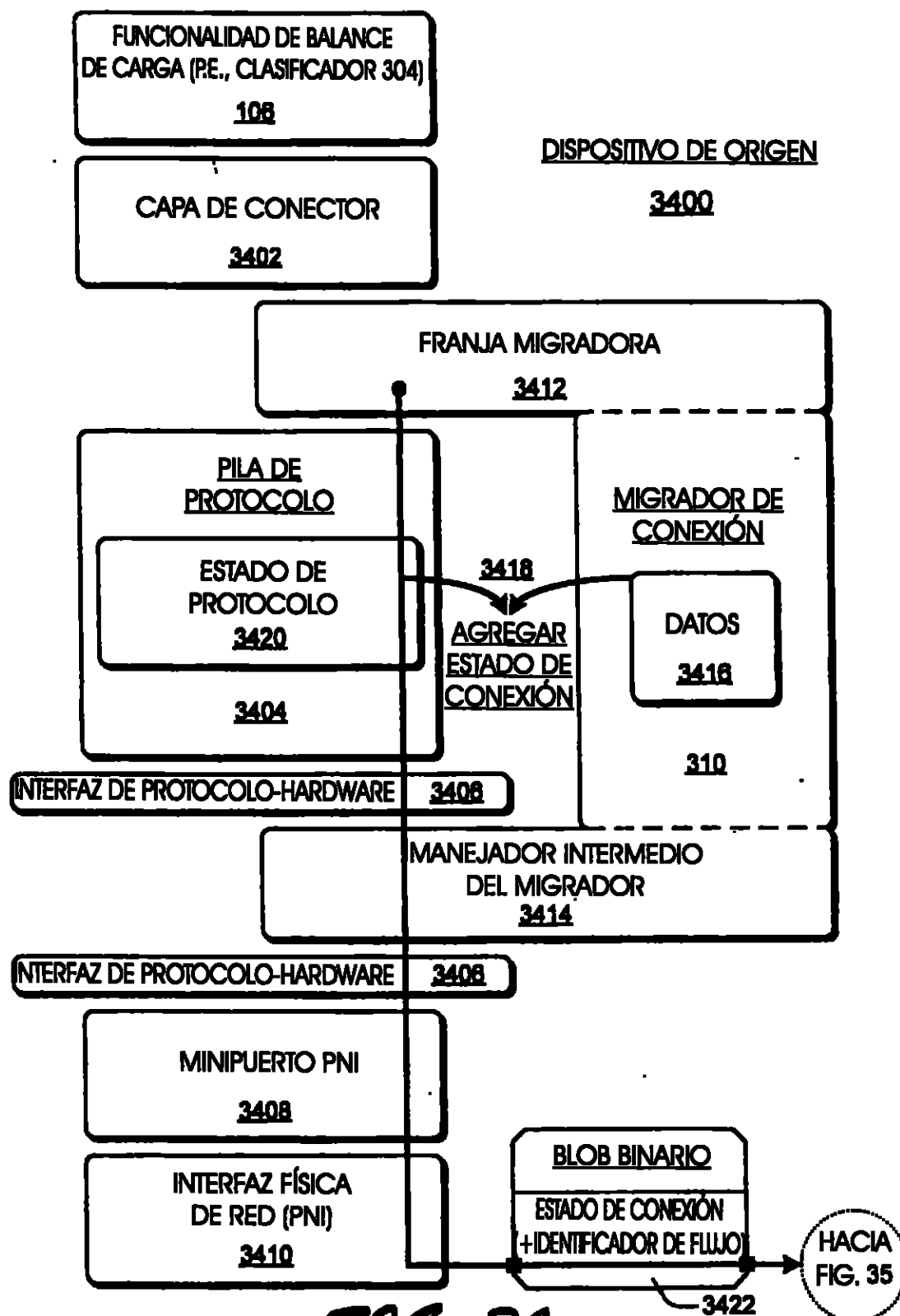


Fig. 34

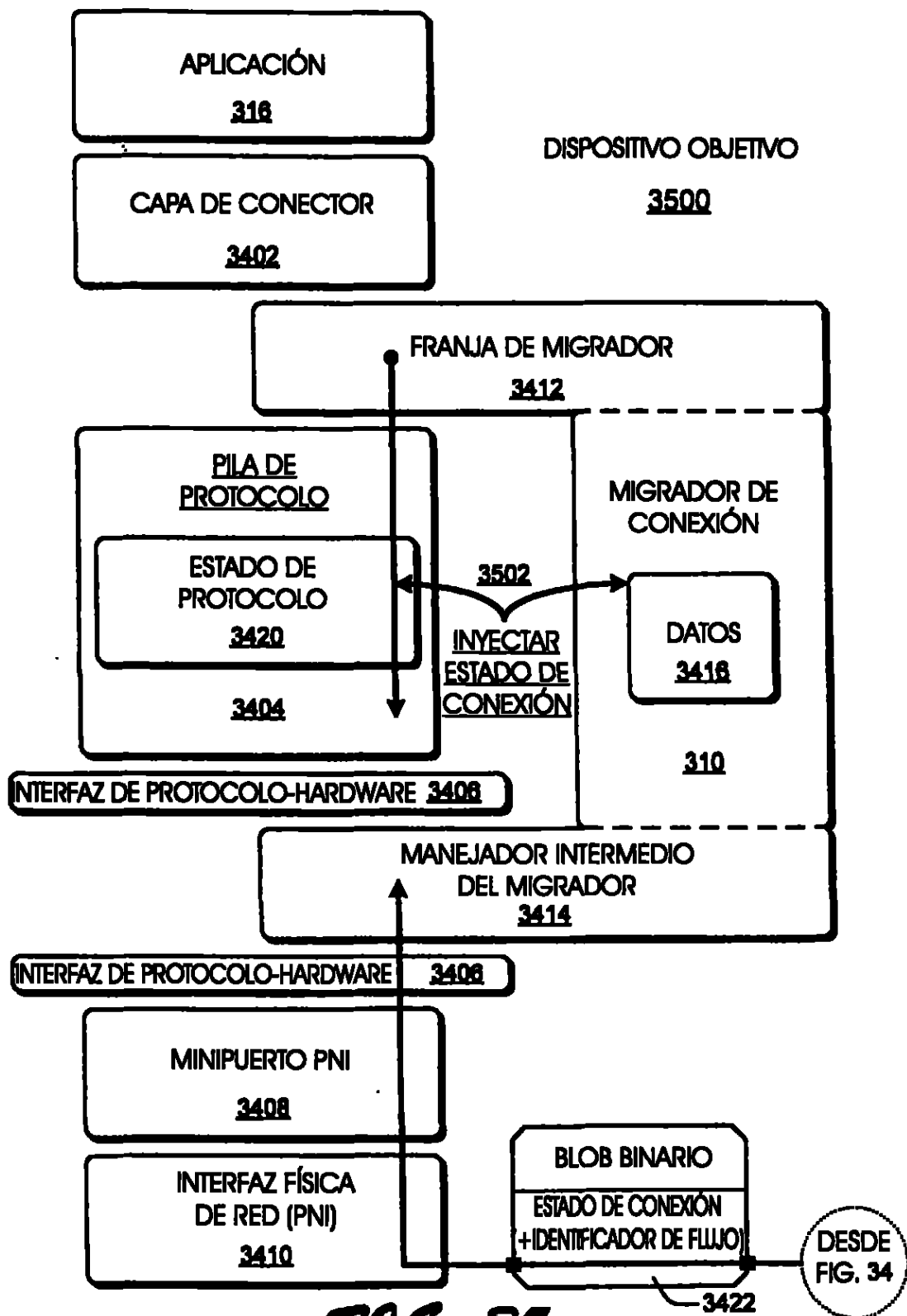


Fig. 35

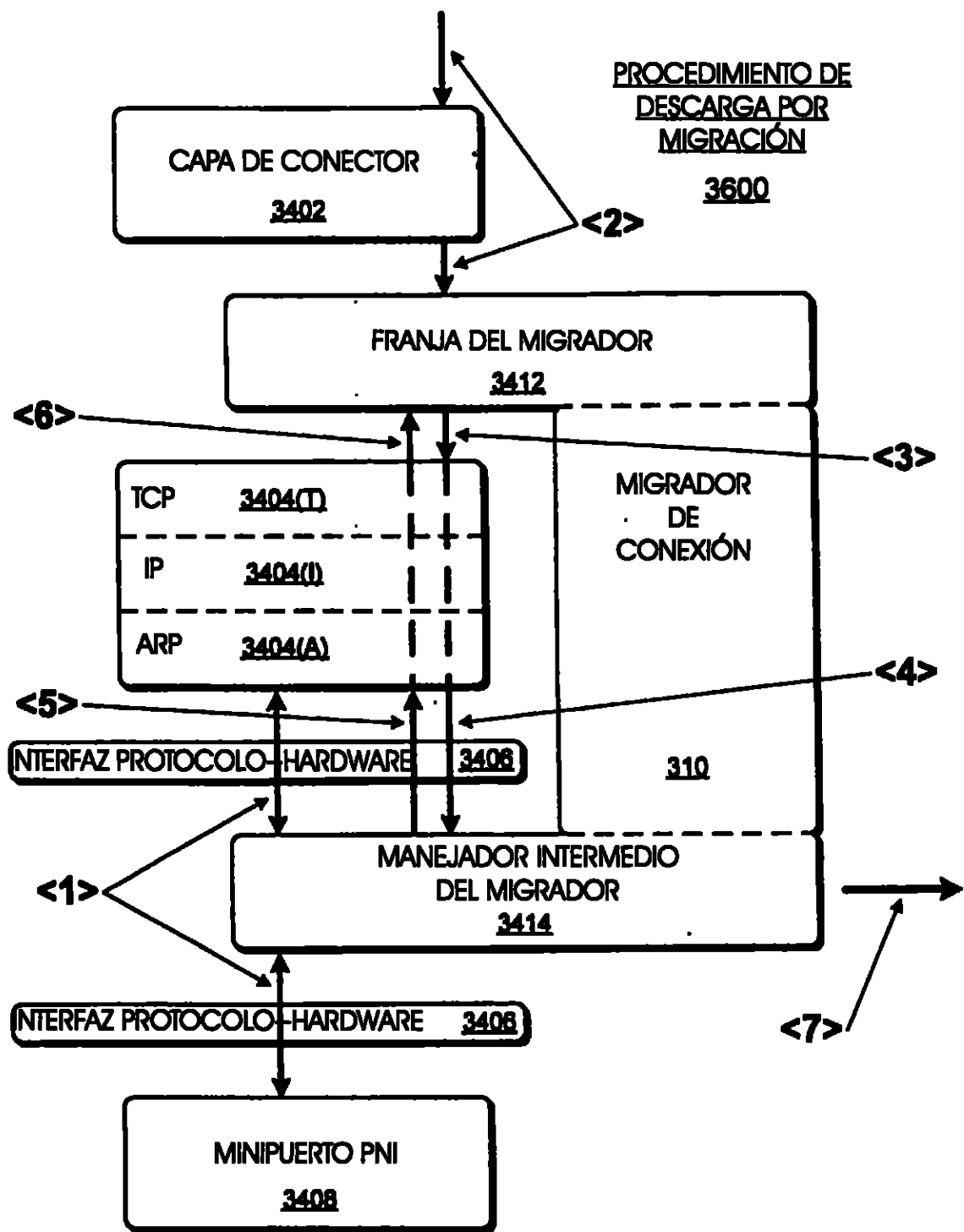


Fig. 36

49

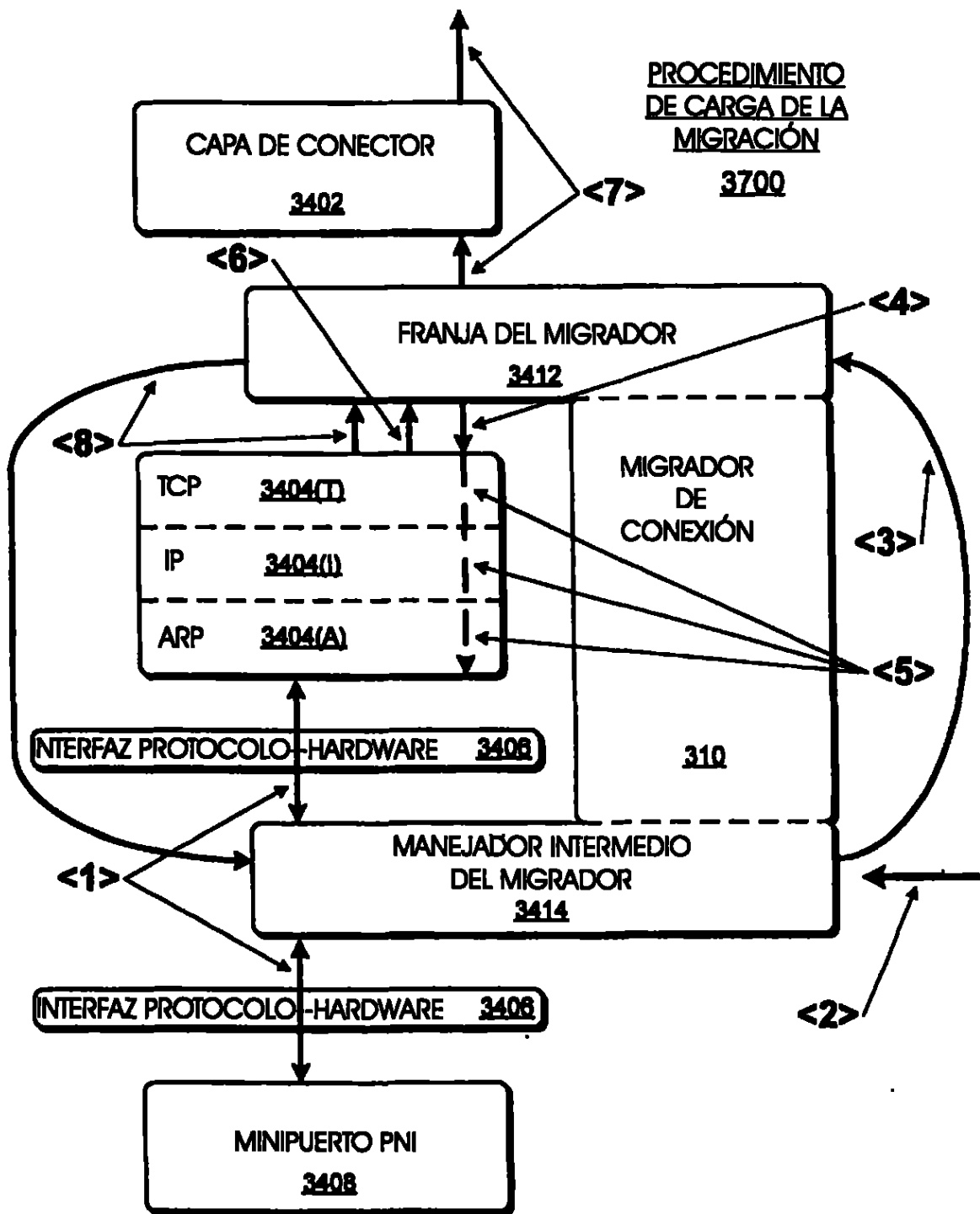


Fig. 37

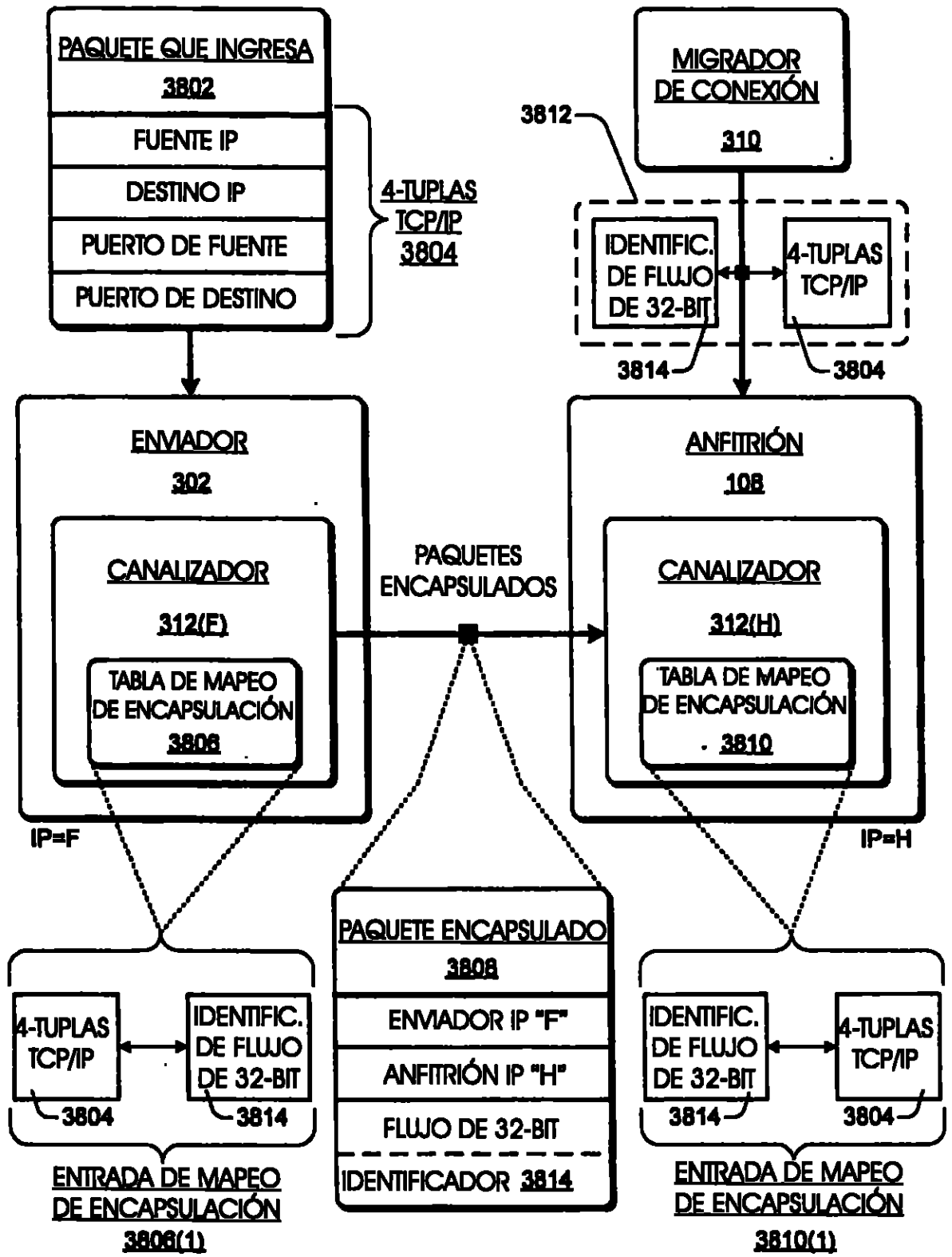


Fig. 38

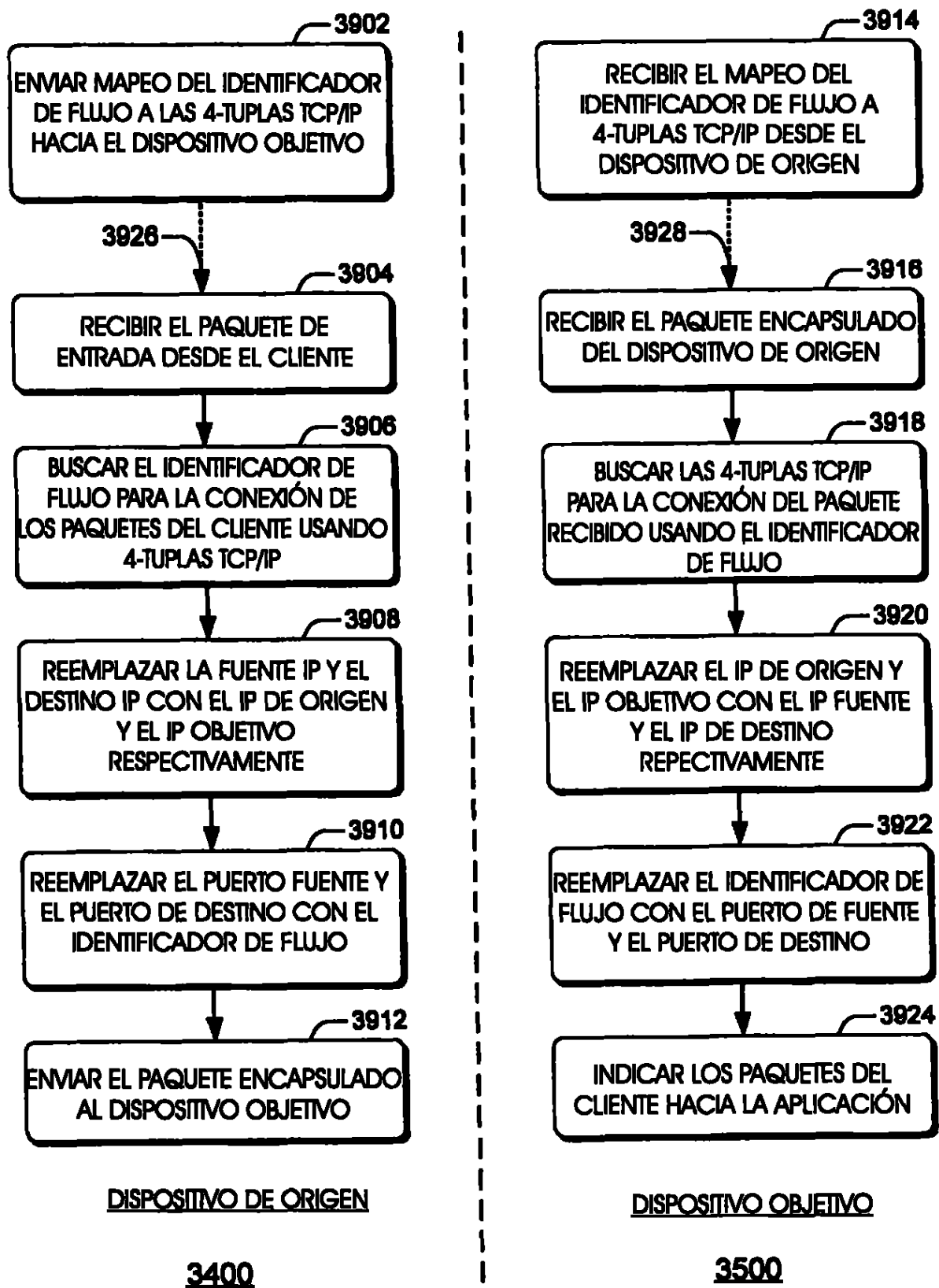
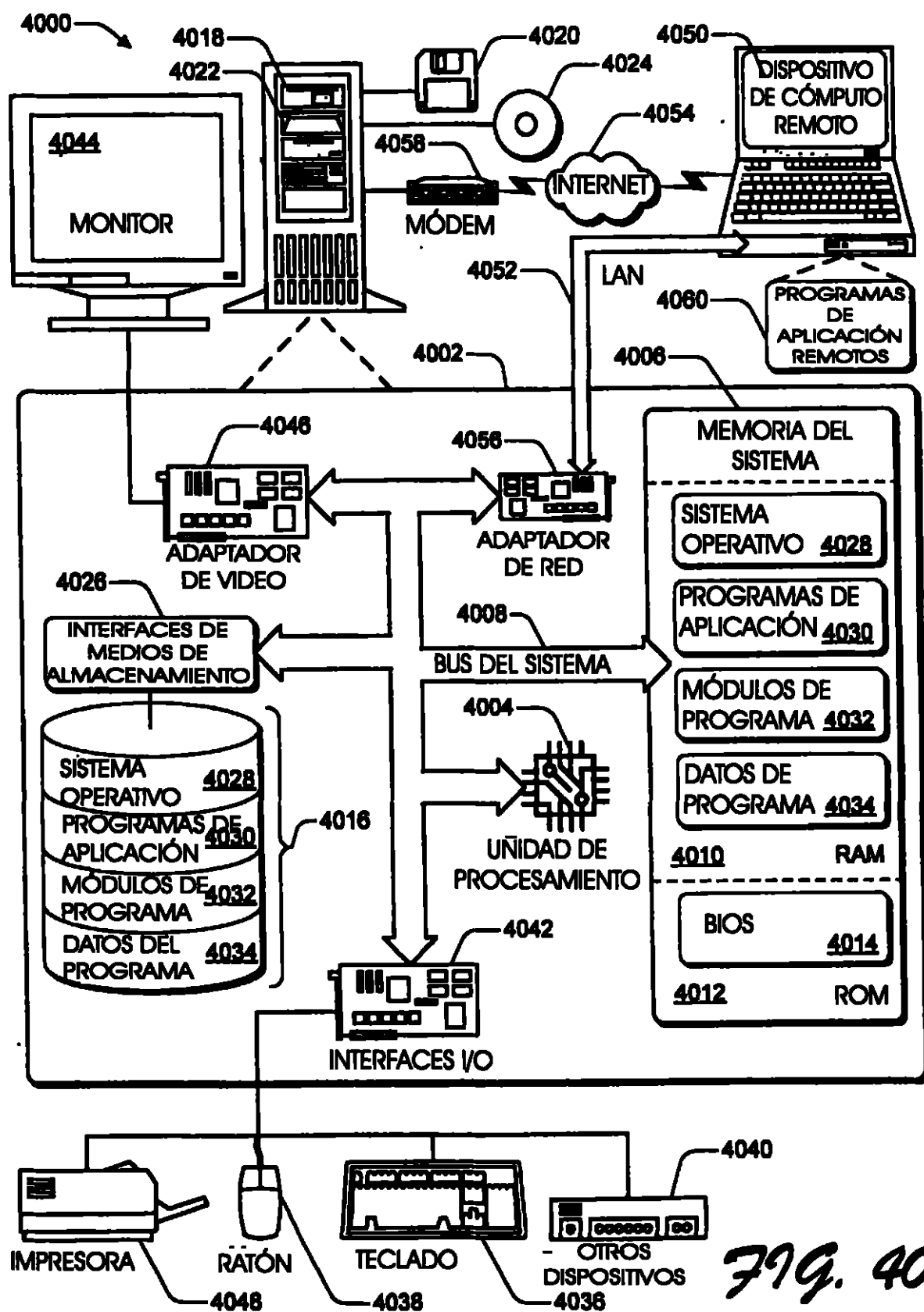
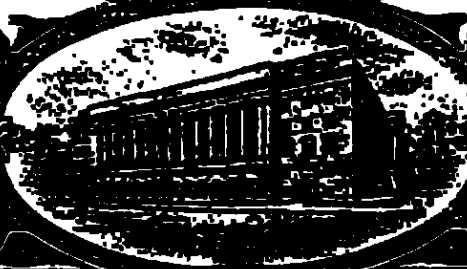


Fig. 39

3900

474





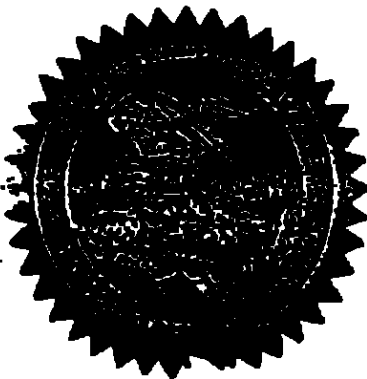
LOS ESTADOS UNIDOS DE AMÉRICA

A TODO AQUEL A QUIEN CONCIERNA LO PRESENTE
DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS
Oficina de Patentes y Marcas de los Estados Unidos

Mayo 18, 2004

LA PRESENTE ES PARA CERTIFICAR QUE ADJUNTA A LA MISMA HAY UNA COPIA FIEL DE LOS REGISTROS DE LA OFICINA DE PATENTES Y MARCAS DE LOS ESTADOS UNIDOS DE LOS DOCUMENTOS DE LA SOLICITUD DE PATENTE IDENTIFICADA EN LO QUE SIGUE, QUE CÚMPLEN LOS REQUISITOS PARA OTORGARLE UNA FECHA DE PRESENTACIÓN BAJO 35 USC 111.

NÚMERO DE SOLICITUD: 10/610,519
FECHA DE PRESENTACIÓN: Junio 30, 2003



Por la Autoridad del
COMISARIO DE PATENTES Y MARCAS

L. Edelen

L. EDELEN
Oficial Certificador

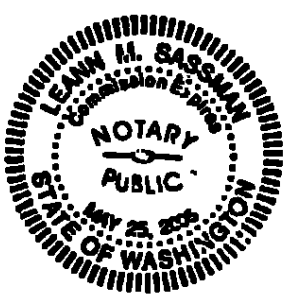
casilla → ☐

Estado de Washington)

ss:

Condado de Spokane)

Certifico que esta es una copia de un documento de
Traspaso que otorga Microsoft Corporation el derecho de
registrar solicitudes extranjeras directamente en nombre
de Microsoft Corporation



Fechado *Mayo 20 2004*

Firma del
Notario Público *Aparece firma*

Mi nombramiento expira en *5/25/05*

197



MSI-
1520US

**OFICINA DE MARCAS Y PATENTES DE LOS
ESTADOS UNIDOS**

**DE ACUERDO CON LA SECRETARÍA DE COMERCIO PARA LA
PROPIEDAD INTELECTUAL Y DIRECTOR DE LA OFICINA DE MARCAS Y
PATENTES DE LOS ESTADOS UNIDOS**

Febrero 11, 2004

PTAS

LEE & HAYES, PLLC
KEITH W. SAUNDERS
421 WEST RIVER SIDE AVENUE
SUITE 500
SPOKANE, WA 99201

**NOTIFICACIÓN
CORREGIDA**

**OFICINA DE MARCAS Y PATENTES DE LOS ESTADOS UNIDOS
NOTIFICACIÓN DE REGISTRO DE DOCUMENTO DE TRASPASO**

EL DOCUMENTO ADJUNTO HA SIDO REGISTRADO POR LA DIVISIÓN
DE TRASPASOS DE LA OFICINA DE MARCAS Y PATENTES DE LOS
ESTADOS UNIDOS. UNA COPIA DE MICROPELÍCULA COMPLETA ESTÁ
DISPONIBLE EN EL CUARTO DE BÚSQUEDA DE TRASPASOS EN EL
NÚMERO DE CARRETE Y CUADRO REFERENCIADO ADELANTE.

~~288~~

POR FAVOR REVISE TODA LA INFORMACIÓN CONTENIDA EN ESTA NOTIFICACIÓN. LA INFORMACIÓN CONTENIDA EN ESTA NOTIFICACIÓN DE REGISTRO REFLEJA LOS DATOS PRESENTADOS EN EL SISTEMA DE TRASPASO DE MARCAS Y PATENTES. SI USTED PUEDE ENCONTRAR CUALQUIER ERROR O TIENE PREGUNTAS RELACIONADAS CON ESTA NOTIFICACIÓN, USTED PUEDE CONTACTAR AL EMPLEADO CUYO NOMBRE APARECE EN ESTA NOTIFICACIÓN EN 703-308-9723. POR FAVOR ENVÍE SOLICITUD PARA CORRECCIÓN A: OFICINA DE MARCAS Y PATENTES DE LOS ESTADOS UNIDOS, DIVISIÓN DE TRASPASO, CASILLA DE TRASPASOS, CG-4, 1213 JEFFERSON DAVIS HWY, SUITE 320, WASHINGTON, D.DC. 20231.

FECHA DE REGISTRO: 02/04/2004

ROLLO/CUADRO: 014307/0355

NÚMERO DE PÁGINAS: 4

RESUMEN: TRASPASO DE INTERÉS DEL CEDENTE (VER DOCUMENTO PARA DETALLES).

CEDENTE:

DARLING CHRISTOPHER L.

FECHA DEL DOCUMENTO:

08/26/2003

299

CEDENTE:

JOY, JOSEPH M.

FECHA DEL DOCUMENTO:

08/26/2003

✓

CEDENTE:

SHRIVASTAVA, SUNITA

FECHA DEL DOCUMENTO:

08/26/2003

CEDENTE:

SUBBARAMAN, CHITTUR

FECHA DEL DOCUMENTO:

08/26/2003

CESIONARIO:

MICROSOFT CORPORATION

ONE MICROSOFT WAY

REDMOND, WASHINGTON 98052

014307/0355 PÁGINA 2

NÚMERO DE SERIE: 10610519

FECHA DE PRESENTACIÓN: 06/30/2003

NÚMERO DE PATENTE:

FECHA DE PUBLICACIÓN:

DOROTHY RILEY, PARALEGAL

DIVISIÓN DE TRASPASOS

OFICINA DE REGISTROS PÚBLICOS

501

02/04/2004

700064305

PORTADA FORMULARIO DE REGISTRO

SOLO PATENTES

Del PTO-1595

(Rev. 03/01)

OMB No. 0651-0027 (exp. 5/31/2002)

DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS
Oficina de Marcas y Patentes de los Estados Unidos

Configuración de tabuladores⇒ ⇒ ⇒ ▼ ▼ ▼ ▼ ▼ ▼ ▼

Al Honorable Comisionado de Marcas y Patentes: Por favor
registre los documentos originales adjuntos o copia de
estos.

1. Nombre de la(s) parte(s) que traspasa(n):

Christopher I. Darling

Joseph M. Joy

Sunita Shrivastava

Chittur Subbaraman

**Nombre(s) adicional(es) de la(s) parte(s) que
traspasa(n) adjunto(s)**

☐ Si ☒ No

2. Nombre de la(s) parte(s) que recibe(n)

Nombre: Microsoft Corporation

Dirección Interna:

Dirección: One Microsoft Way

Ciudad: Richmond

Estado: Washington

Zip: 98052

Nombre(s) adicional(es) & dirección(es) adjuntas?

☐ Si ☒ No

3. Naturaleza de la transmisión:

☒ Traspaso

☐ Acuerdo de seguridad

☐ Fusión

☐ Cambio de Nombre

Otro _____

Fecha de Legalización 08/26/2003

4. Número de la(s) solicitud(es) o número(s) de
patente(s): 10/610,519

Si este documento está siendo presentado junto con
una nueva publicación, la fecha de formalización de
la solicitud es: _____

A. Solicitud de Patente No.(s)

B. Patente No.(s)

Números adicionales adjuntos?

☐ Si ☒ No

5. Nombre y dirección de la parte a quien los
documentos de correspondencia concernientes deban
ser enviados:

Nombre: Keith W. Saunders, Reg. No. 41,462

Dirección Interna: Lee & Hayes, PLLC
Dirección: 421 West Riverside Avenue
Suite 500
Ciudad: Spokane
Estado: WA
Zip: 99201

6. Número total de solicitudes y patentes involucradas: 1

7. Tasa Total (37 CFR 3.41)\$40.00

☐ Incluido

☒ Autorizado a ser cargado a cuenta de depósito

8. Número de cuenta de depósito
12-0769

(adjunte copia duplicado de esta página si paga en cuenta de depósito)

NO UTILICE ESTE ESPACIO

9. Declaración y Firma. .

~~305~~

A mi leal saber y entender, la anterior información es correcta y verdades y cualquier copia adjunta es una copia verdades del documento original.

Keith W. Saunders, Reg. No. 41,482

Nombre de la persona que firma

(Aparece firma)

firma

02/04/2004

Fecha

Número total de páginas que incluyen esta portada, adjuntos y documentos: 4

Documentos de correspondencia a ser registrados con la información de portada requerida a:

Commissioner for Patents & Trademarks,

Mail Stop Assignments

Alexandria, Virginia 22313-1450

~~518~~

Registro de Abogado No. MS1-1520US
Registro MS No. 304011.1
No. de Serie 10/610,519

EN LA OFICINA DE MARCAS Y PATENTES DE LOS ESTADOS UNIDOS

Número de Serie: 10/610,519
Fecha de Presentación: Junio 30, 2003
Inventores Darling, et al.
Solicitante Microsoft Corporation
Certificación de Abogado No. MS1-1520US
Registro MS No. 304011.1
Título: **BALANCEO DE CARGA DE RED CON INFORMACIÓN DE
ESTADO DE HUÉSPED**

TRASPASO DE PATENTE

PARTES EN EL TRASPASO

Cedente(s):

Christopher L. Darling
2721 233rd Place, NE
Sammamish, WA 98074

Joseph M. Joy

17219 NE 32nd Street

Redmond, WA 98052

Sunita Shrivastava

14924 NE 76th Ct.

Redmond, WA 98052

Chittur Subbaraman

5037 326th Place NE

Carnation, WA 98014

Cesionario:

Microsoft Corporation

Corporation in the State of Washington

One Microsoft Way

Redmond, WA 98052

ACUERDO

CONSIDERANDO, QUE LOS CEDENTES (listados anteriormente)
son inventores de la invención titulada "BALANCEO DE
CARGA DE RED CON INFORMACIÓN DE ESTADO DE HUÉSPED", como

se describió y reivindicó en la especificación que forma parte de la solicitud para cartas de patente de Estados Unidos referenciadas anteriormente;

CONSIDERANDO, que Microsoft Corporation (llamado de aquí en adelante como CESIONARIO), una corporación del Estado de Washington que tiene lugar de negocio en One Microsoft Way, Redmond, WA 98052, deseosa de adquirir los derechos completos, título e interés en y a la invención y en cualquier carta de patente que por lo tanto pueda ser otorgada en los Estados Unidos y en cualquiera y todos los países extranjeros;

POR LO TANTO, AHORA, en intercambio de buena y valiosa consideración, el recibo del cual se reconoce aquí, los CEDENTES por medio del presente venden, traspasan y transfieren al CESIONARIO, los derechos completos, título, beneficio en y a dicha invención, dicha solicitud y cualquiera y todos los títulos de patente que puedan ser otorgadas a dicha invención en los Estados Unidos de América, y sus posesiones territoriales y en cualquiera de los países extranjeros, y en cualquiera y todas las reotorgamientos y continuaciones de la misma, incluyendo el derecho a presentar solicitudes en el extranjero

~~521~~

directamente a nombre del CESIONARIO y a reivindicar los derechos de propiedad derivados de dicha solicitud Estadounidense al cual dichas solicitudes extranjeras tienen derecho por virtud de la convención, tratado internacional, o de otra manera, dicha invención, solicitud y todos los títulos de patente sobre dicha invención a ser poseídos y disfrutados por los CESIONARIOS y sus sucesores y cede en su uso y beneficio y el de sus sucesores y cede tan completa y enteramente como si la misma hubiera sido poseída y disfrutada por el cedente si este traspaso o venta no hubiera sido hecho. Por medio de la presente los CEDENTES autorizan y solicitan al Comisionado de Marcas y Patentes a expedir todos los títulos de patente sobre esta invención al CESIONARIO. Los CEDENTES están de acuerdo para firmar todos los instrumentos y documentos requeridos par efectuar y tramitar las solicitudes para los títulos de patente en los Estados Unidos y en el exterior, para litigar con respecto a dichos títulos de patente, o con el propósito de proteger el título a dicha invención o título de patente.



26 de Agosto 2003

(aparece firma ilegible)

Fecha

Christopher L. Darling

Agosto 26, 2003

(aparece firma ilegible)

Fecha

Joseph M. Joy

8/26/2003

(aparece firma ilegible)

Fecha

Sunita Shrivastava

8/26/2003

(aparece firma ilegible)

Fecha

Chittur Subbaraman

524

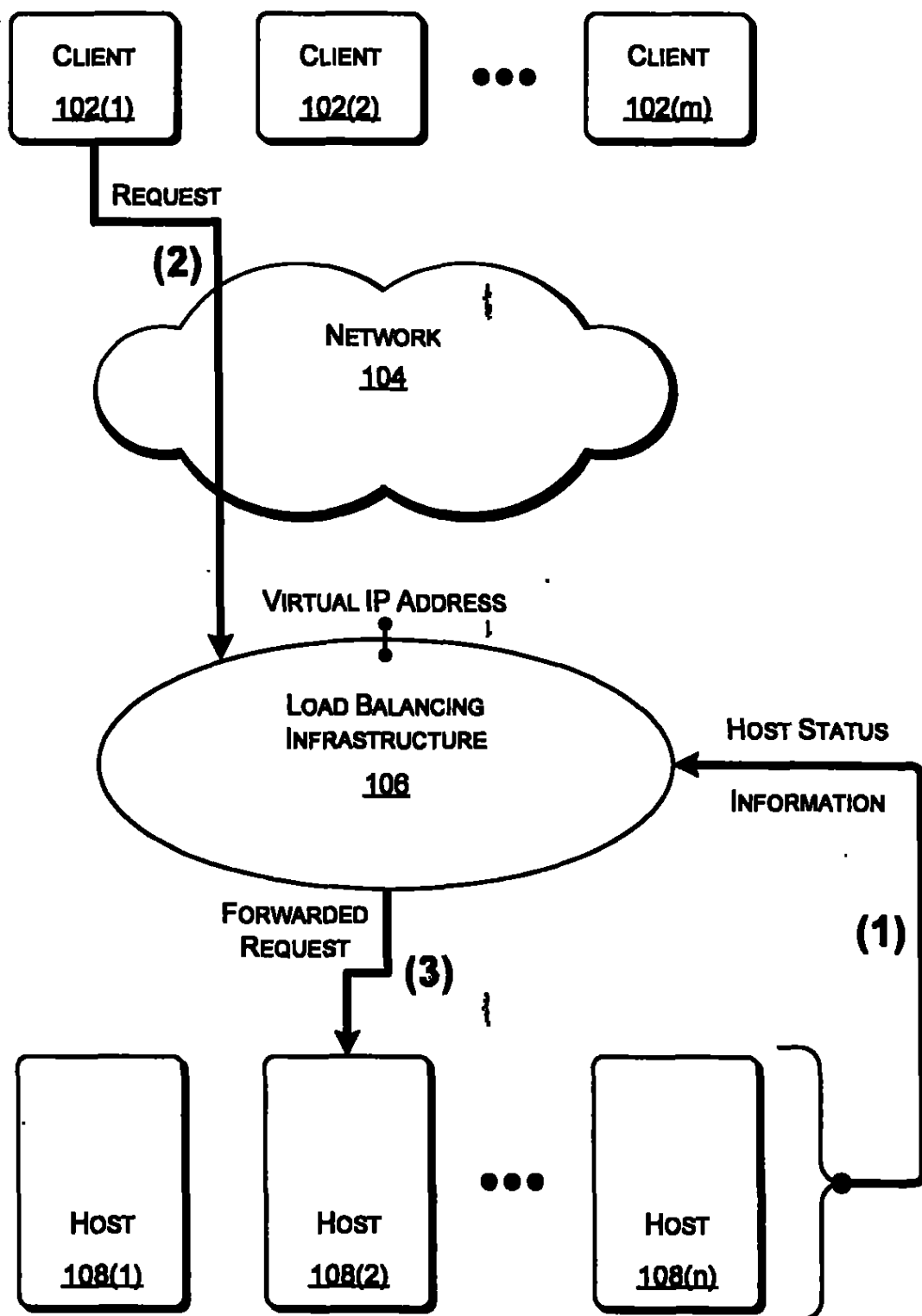


FIG. 1

518

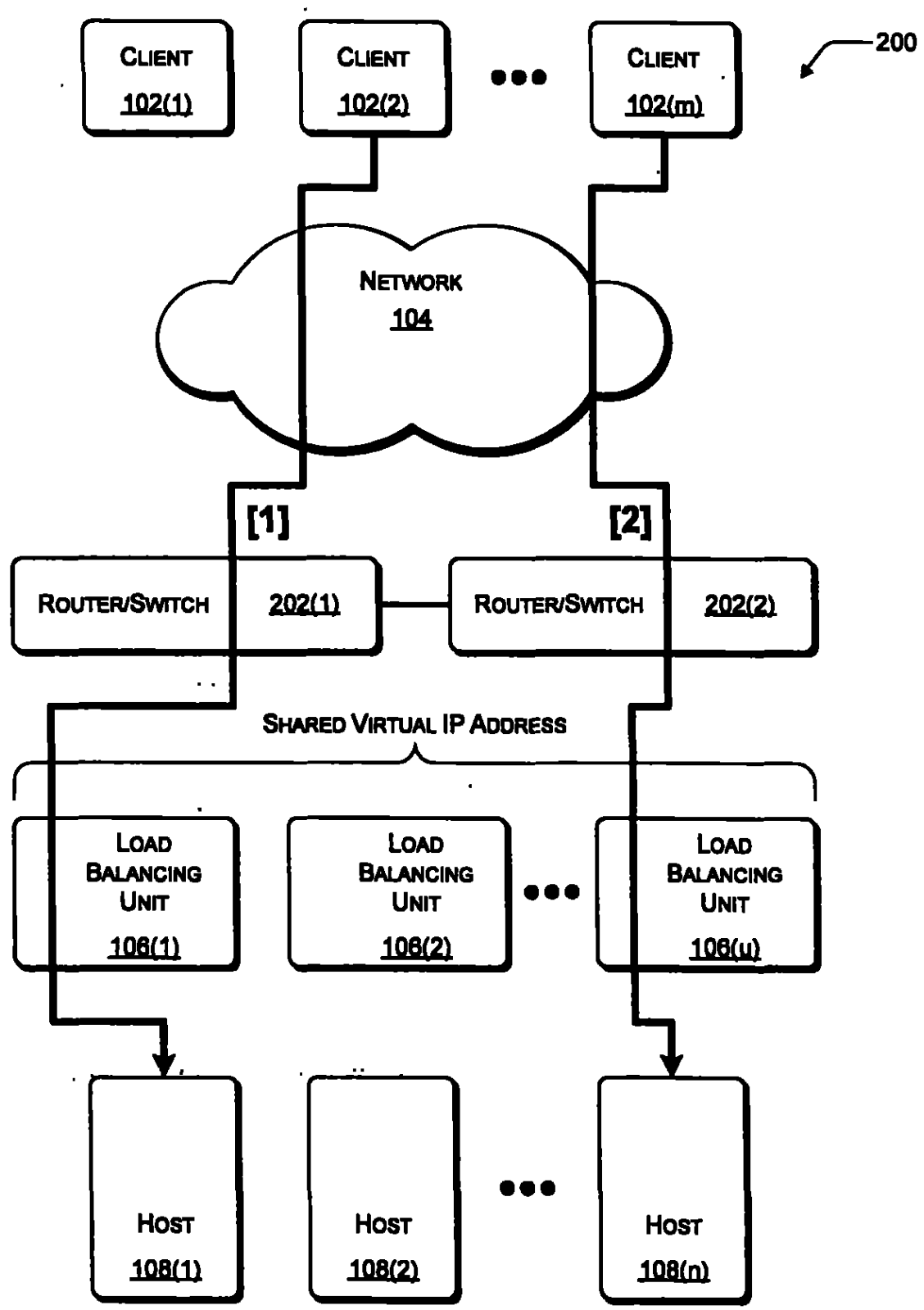


FIG. 2

+

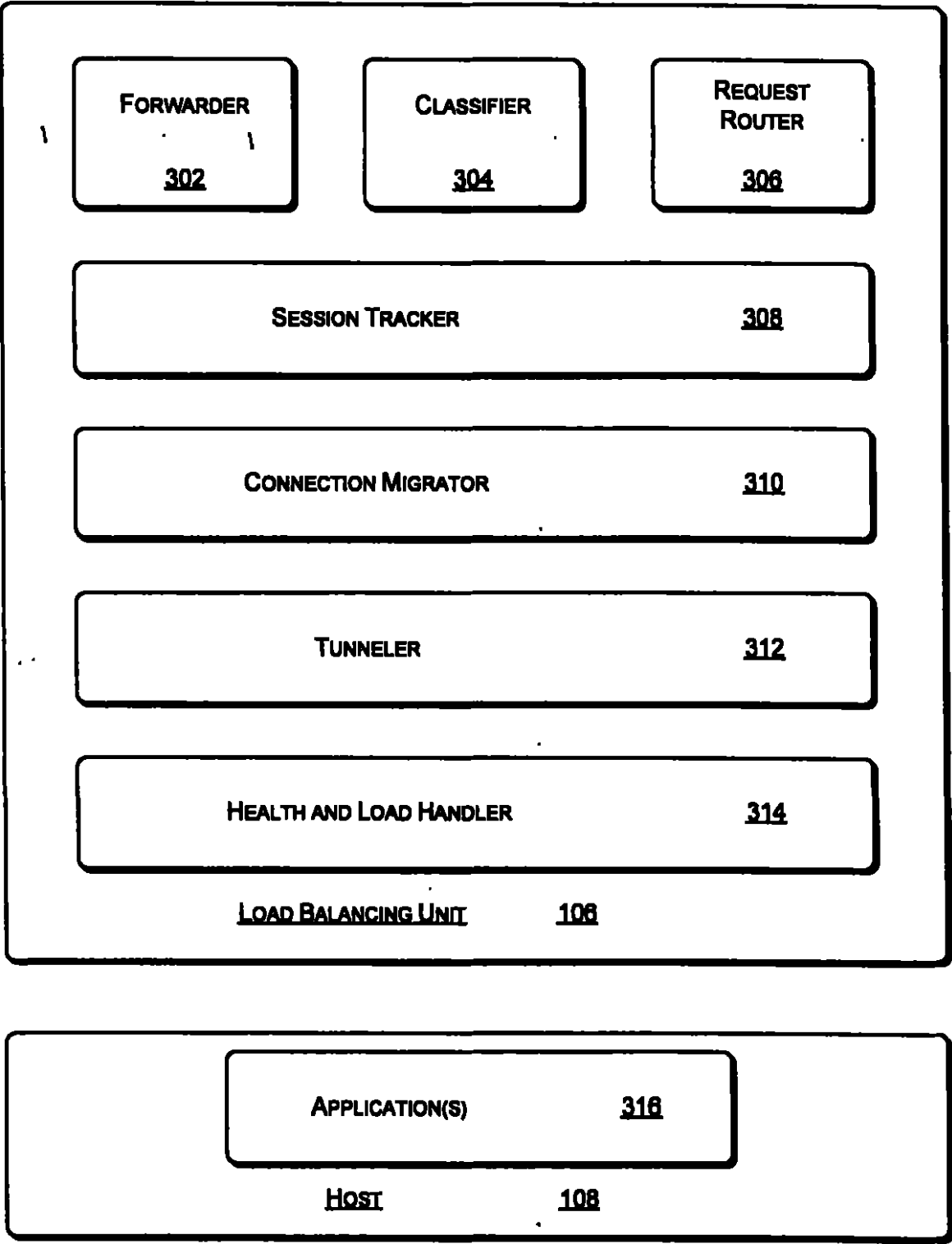


Fig. 3

+

~~519~~

527

+

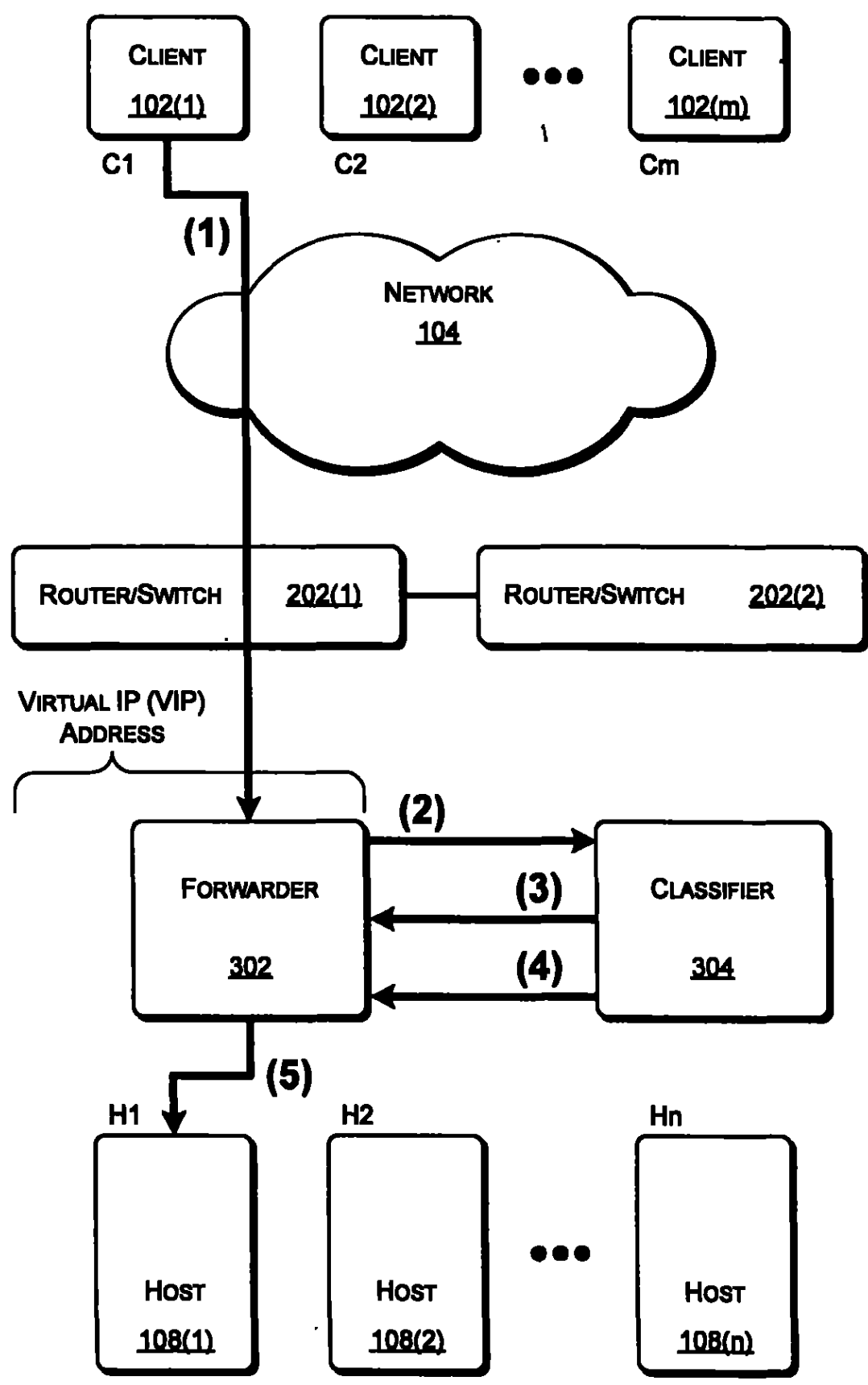


FIG 1

515

+

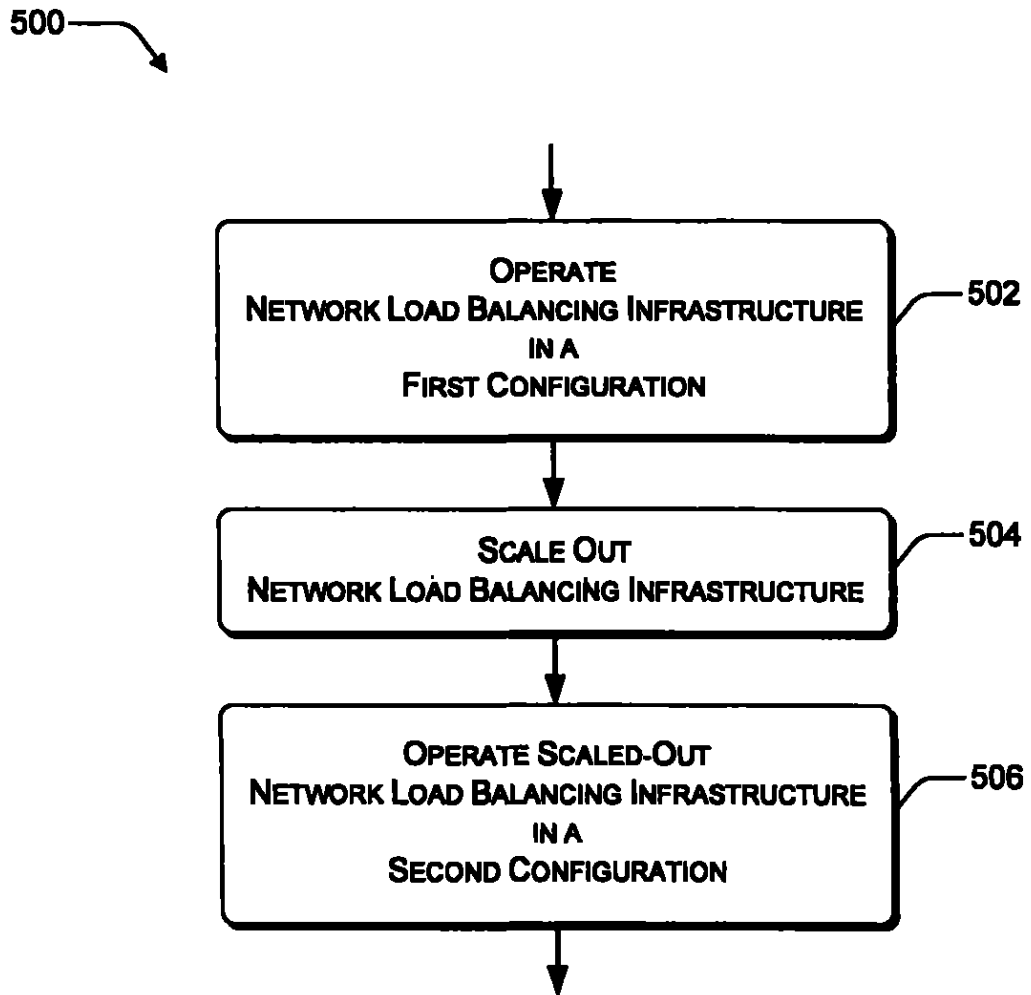


Fig. 5

+

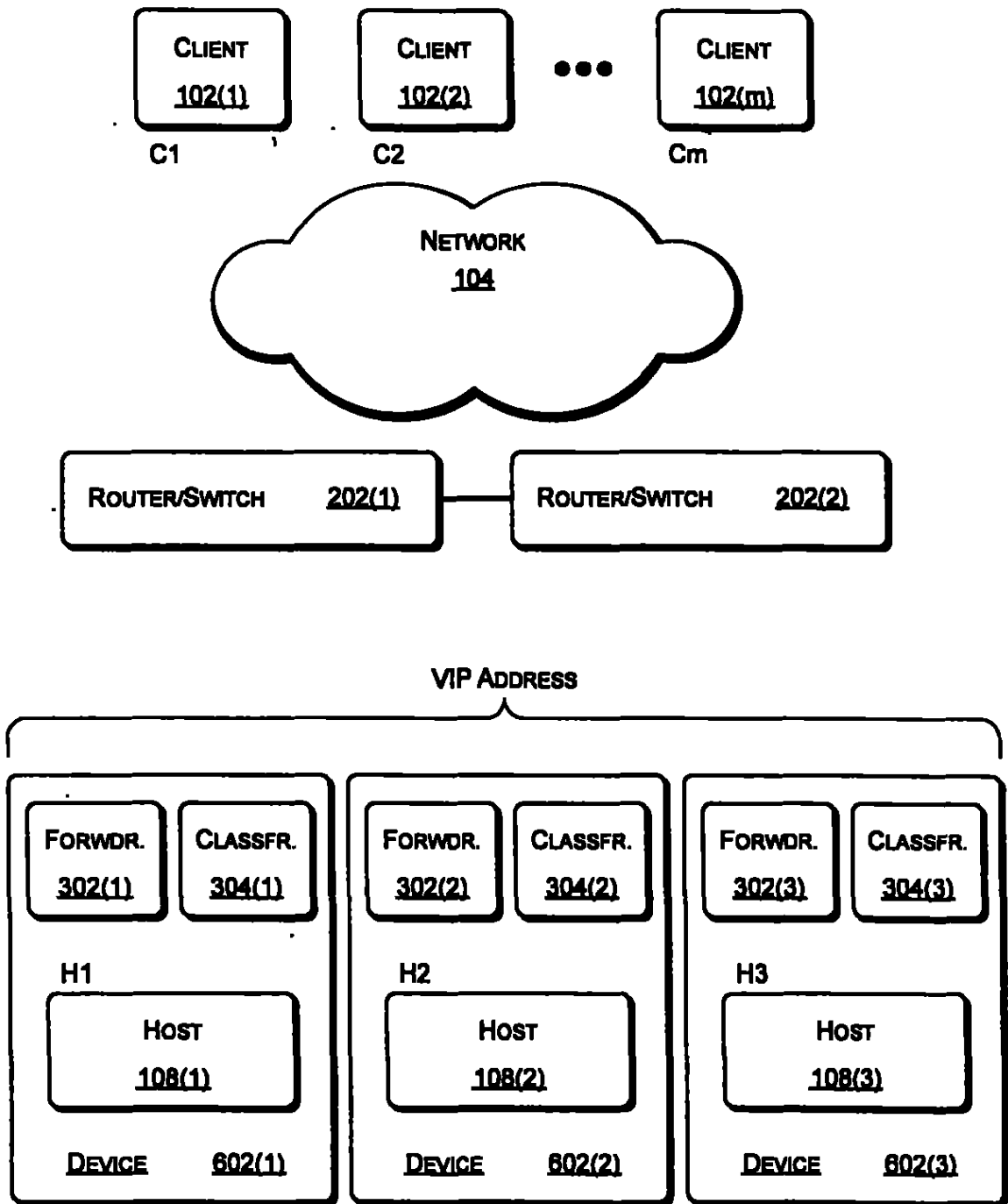


Fig. 6

+

+

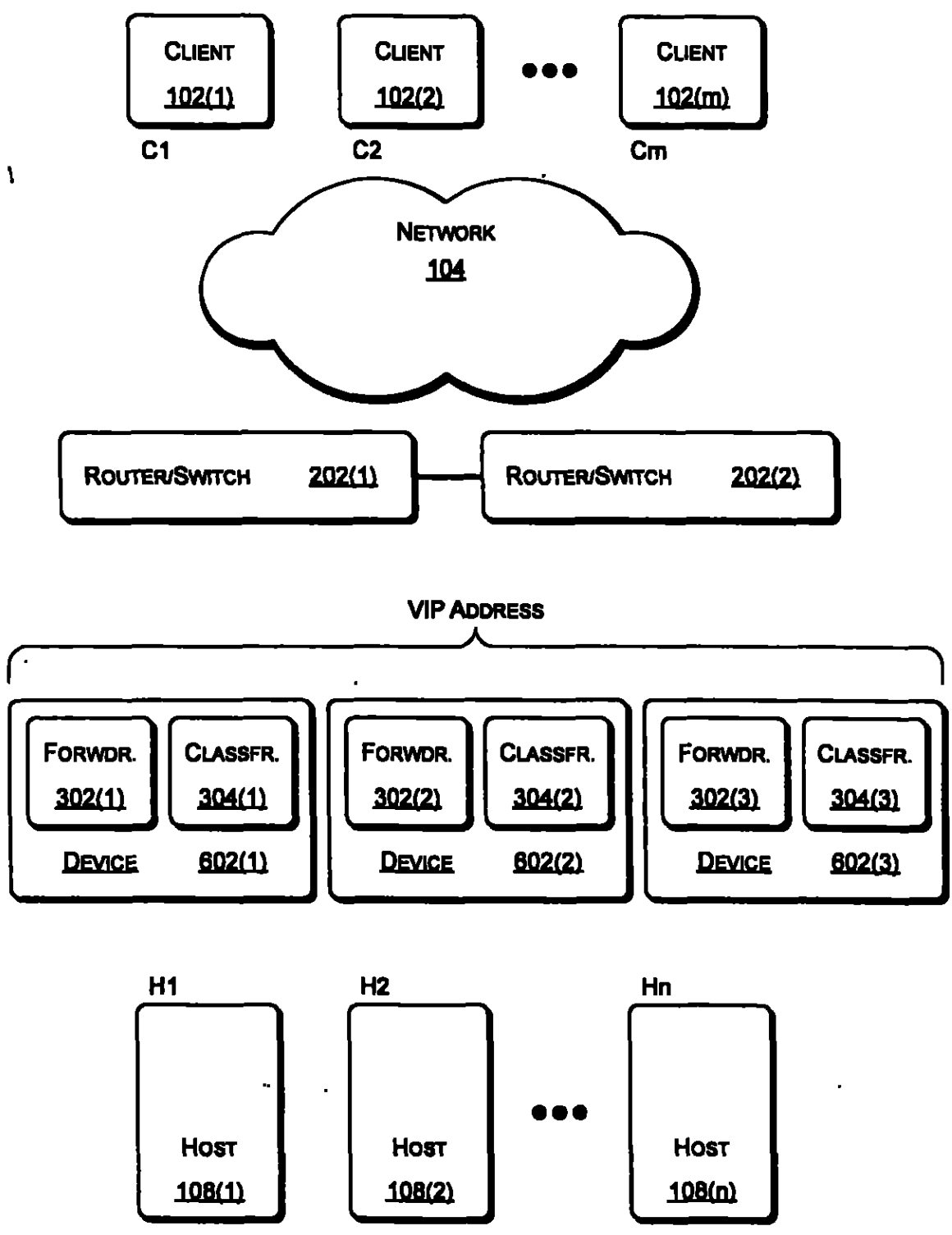


Fig. 7

+

518

+

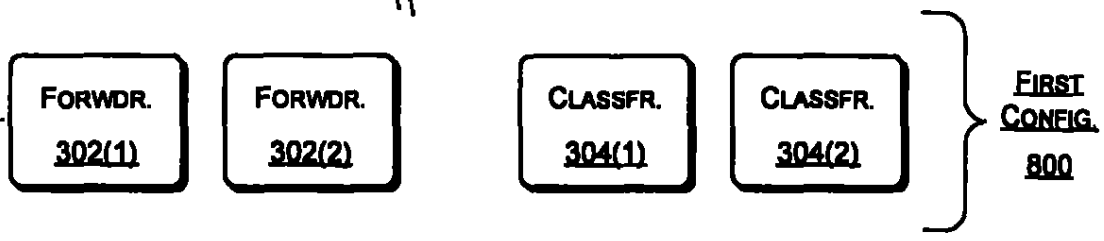


FIG. 8A

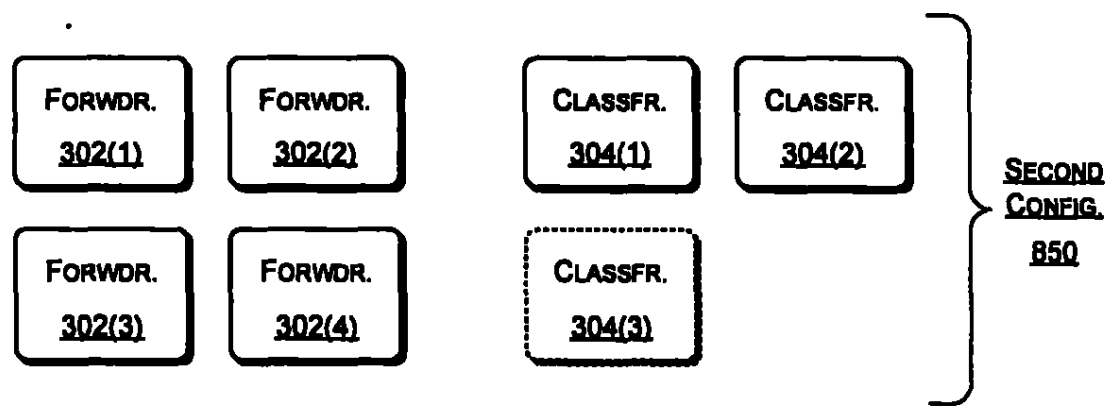


FIG. 8B

+

549

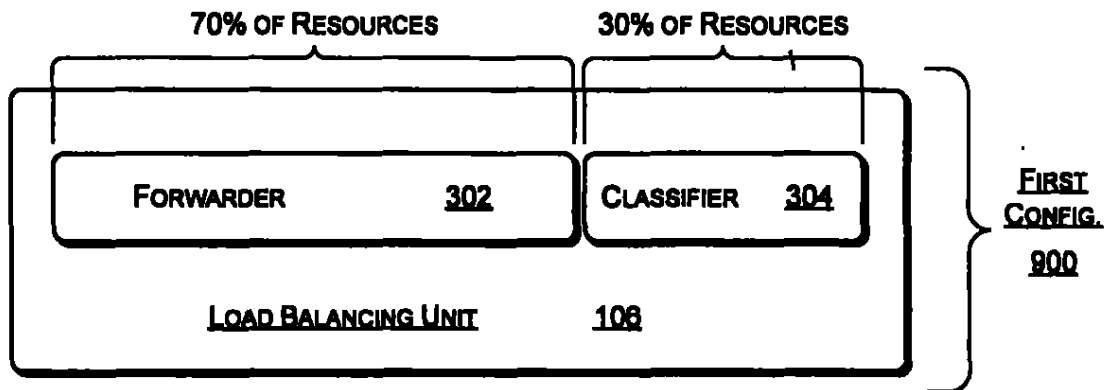


FIG. 9A

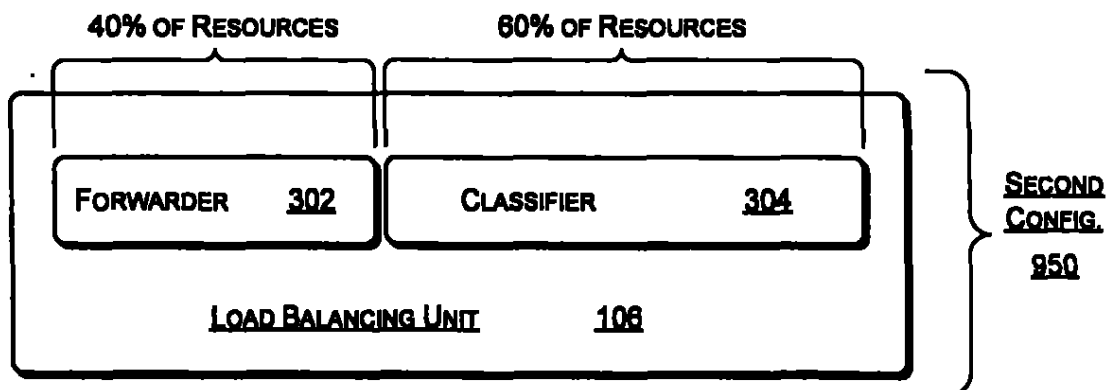


FIG. 9B

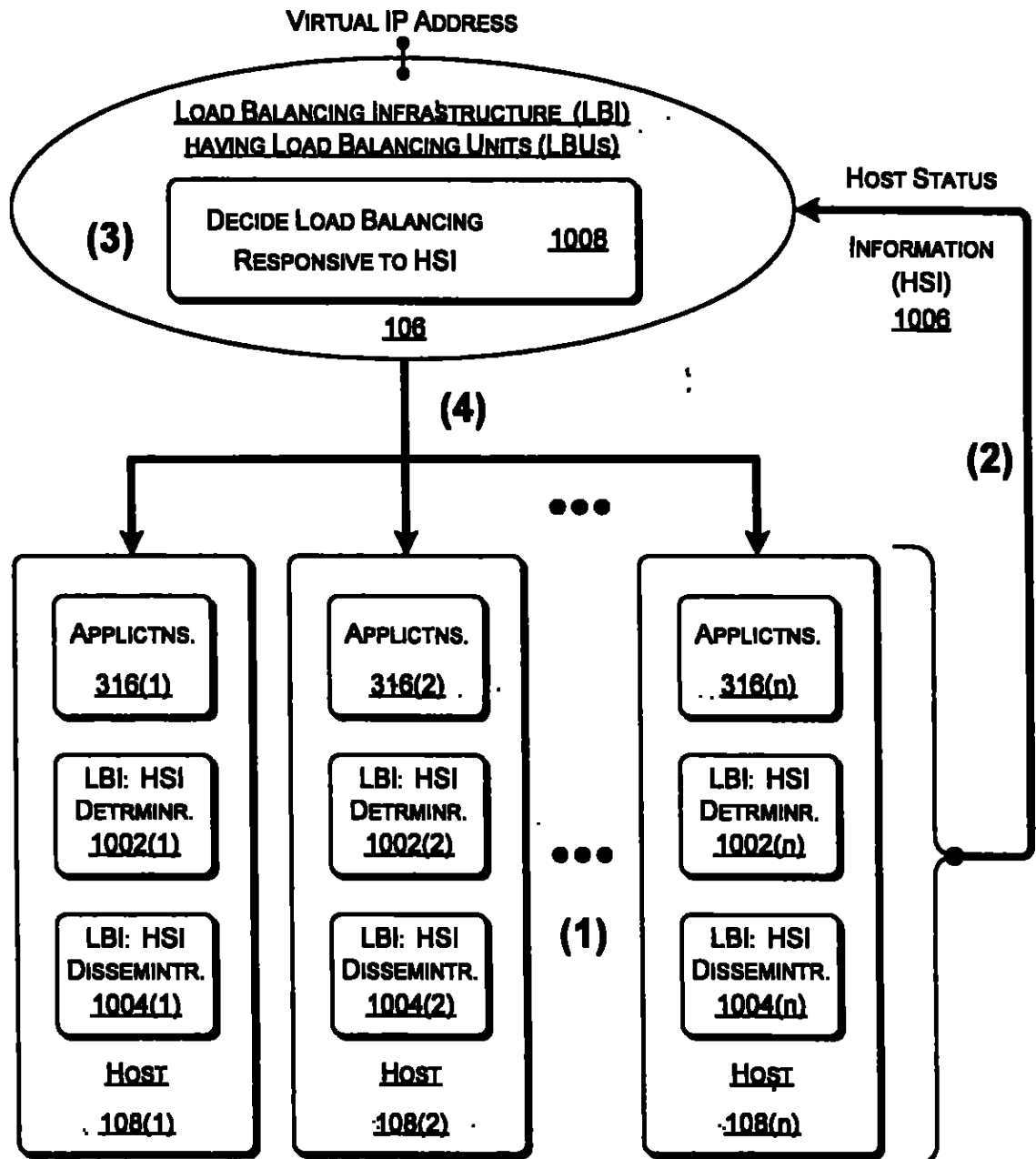


Fig. 10

+

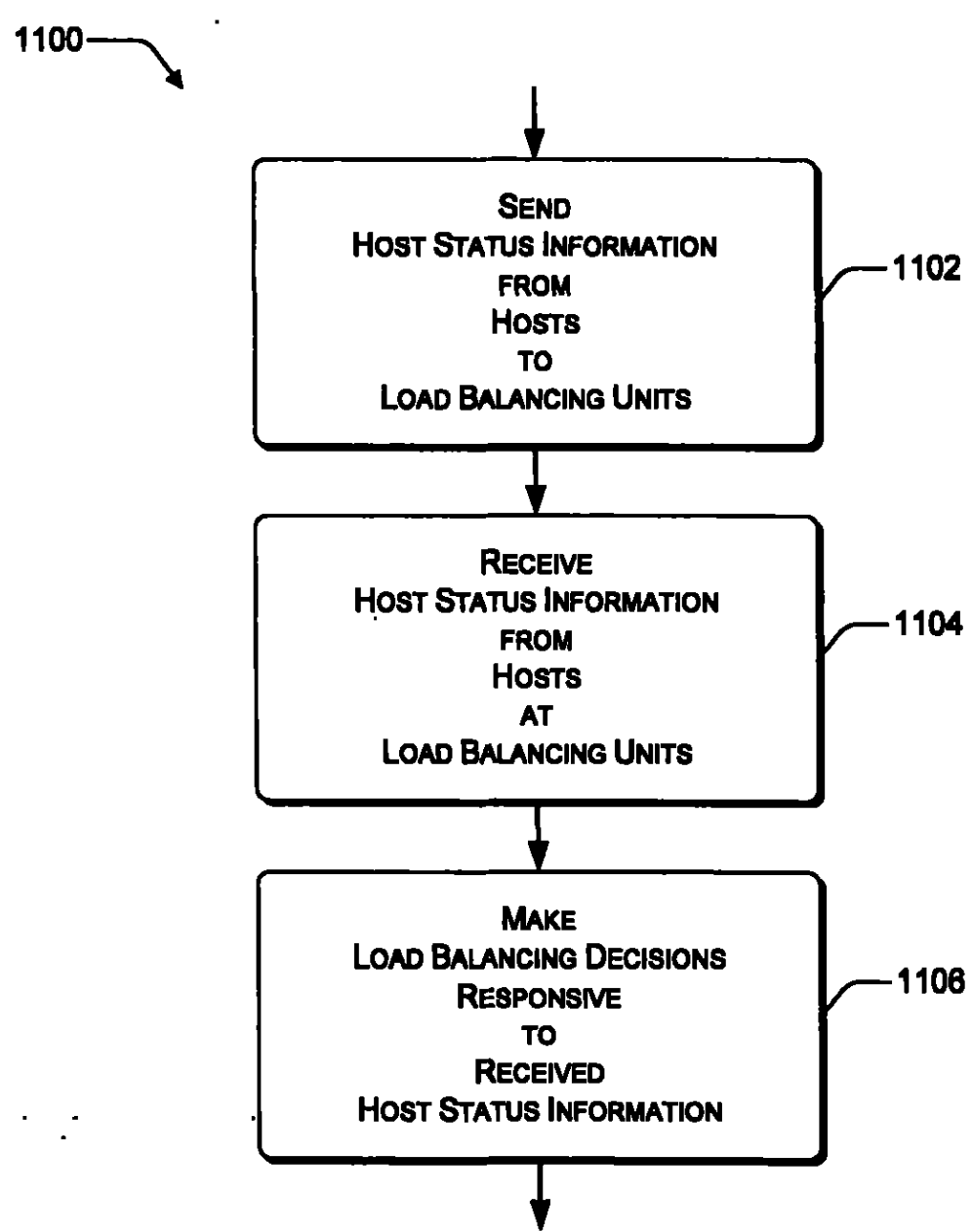


FIG. 11

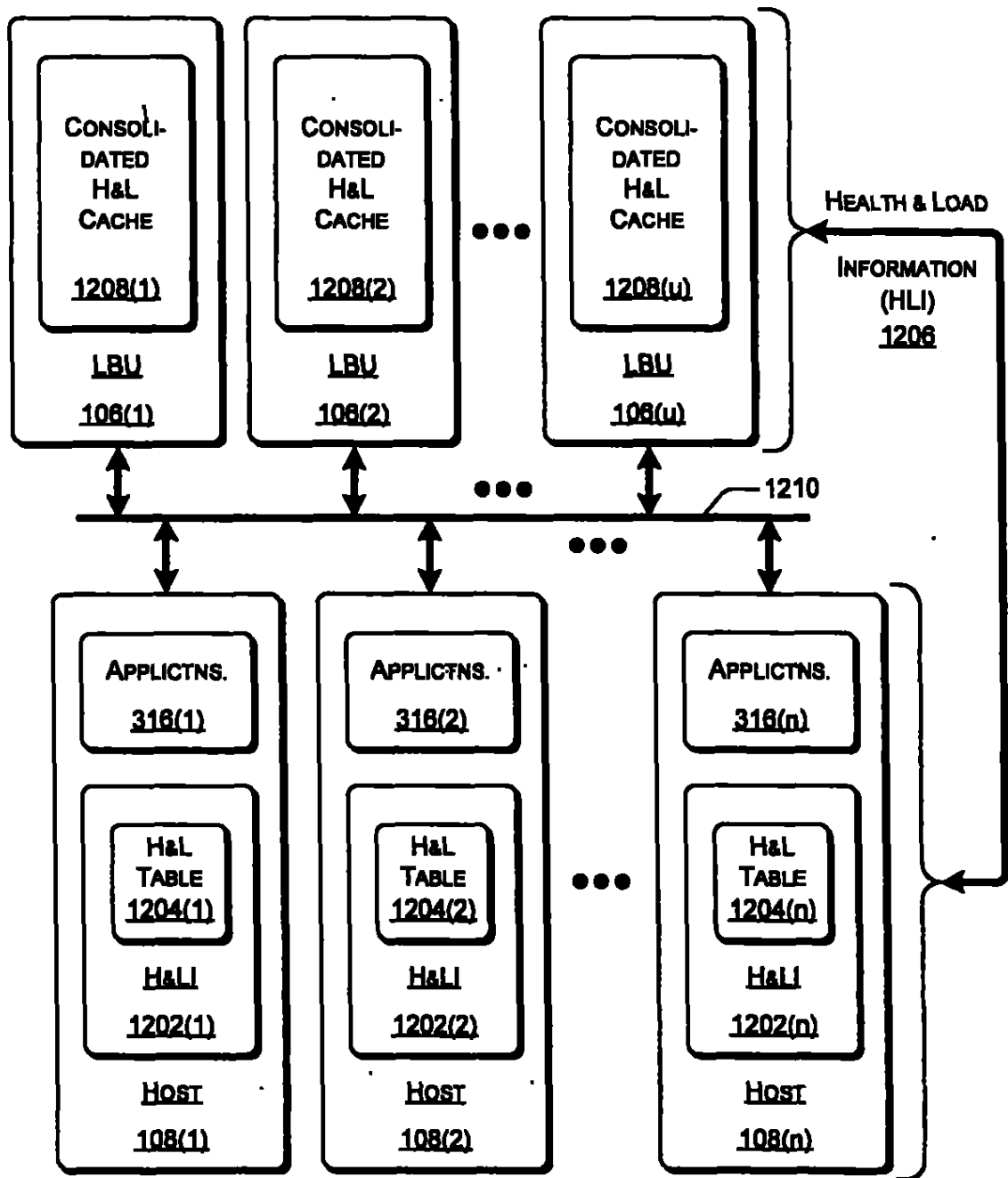


Fig. 12

+

H&L TABLE			1204
APPLICATION IDENTIFIER (ID)	APPLICATION STATUS CHARACTERIZATION	LOAD BALANCER DIRECTIVE	
1302(A)	1302(B)	1302(C)	
VIRTUAL IP ADDR. & PORT	APPLICATION HEALTH	TARGET LOAD BALANCING STATE	
PHYSICAL IP ADDR. & PORT	APPLICATION LOAD		
PROTOCOL	APPLICATION CAPACITY	CURRENT LOAD BALANCING STATE	
PROTOCOL-SPECIFIC INFO.			

Fig. 13A

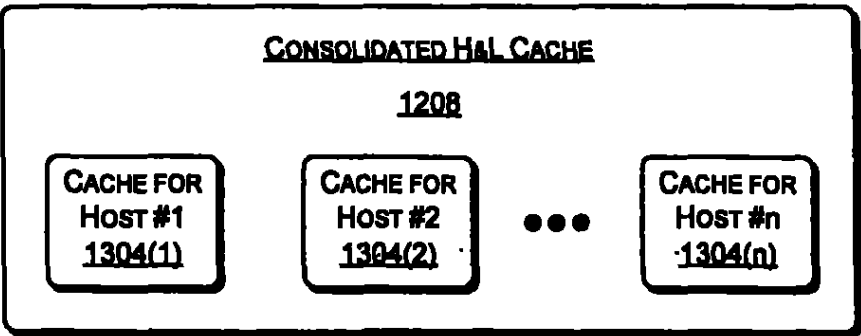


Fig. 13B

+

364

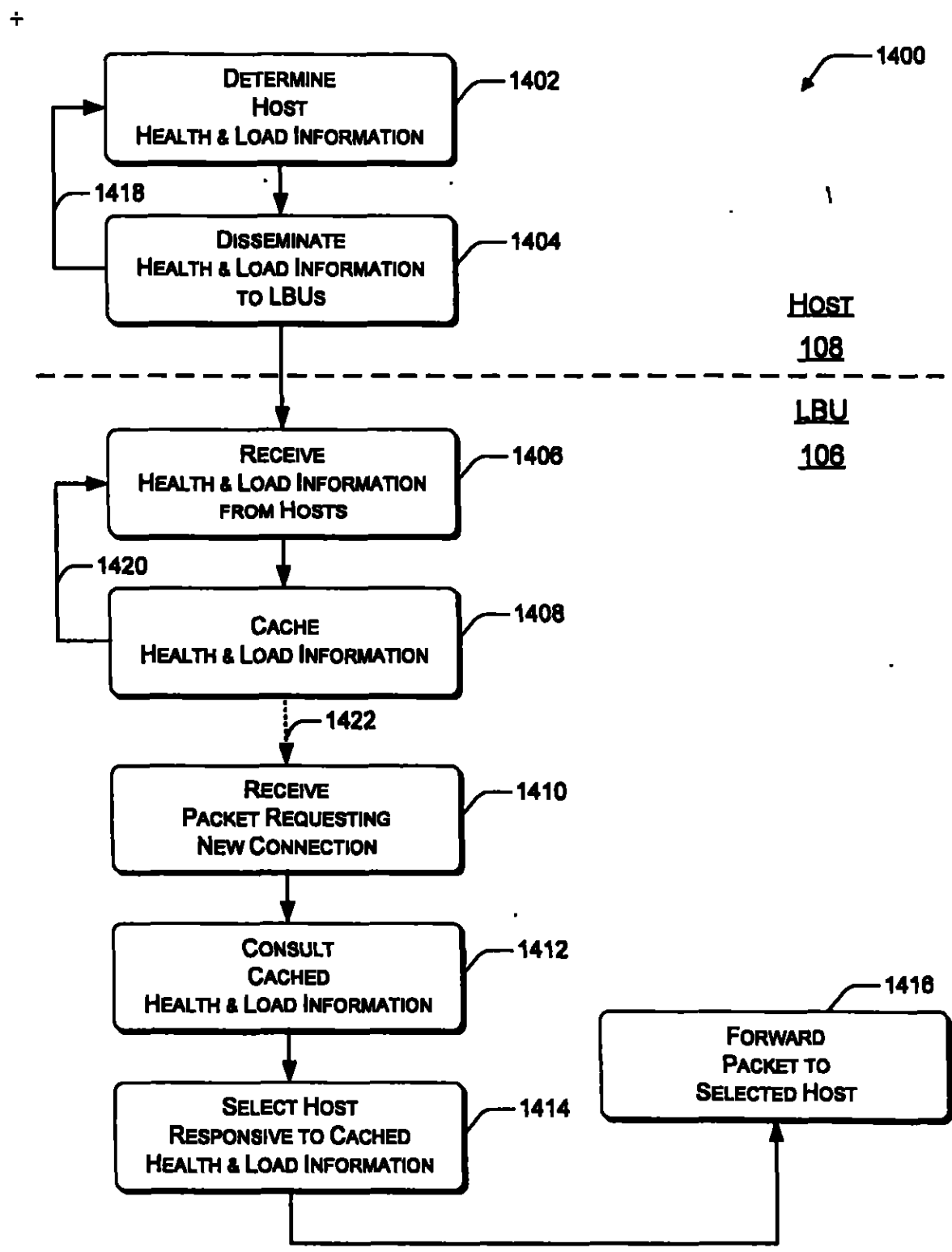


FIG. 14

+

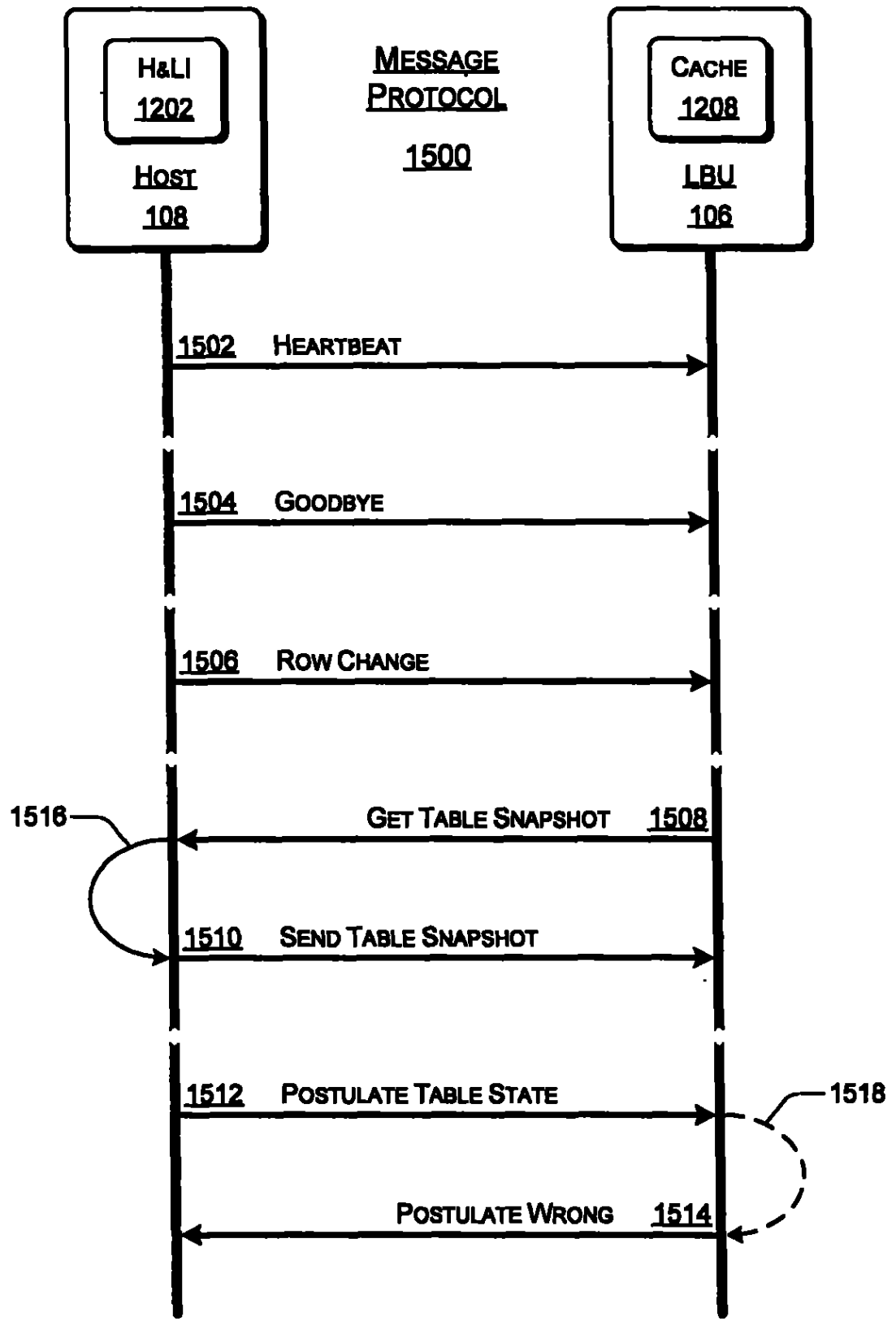


FIG. 15

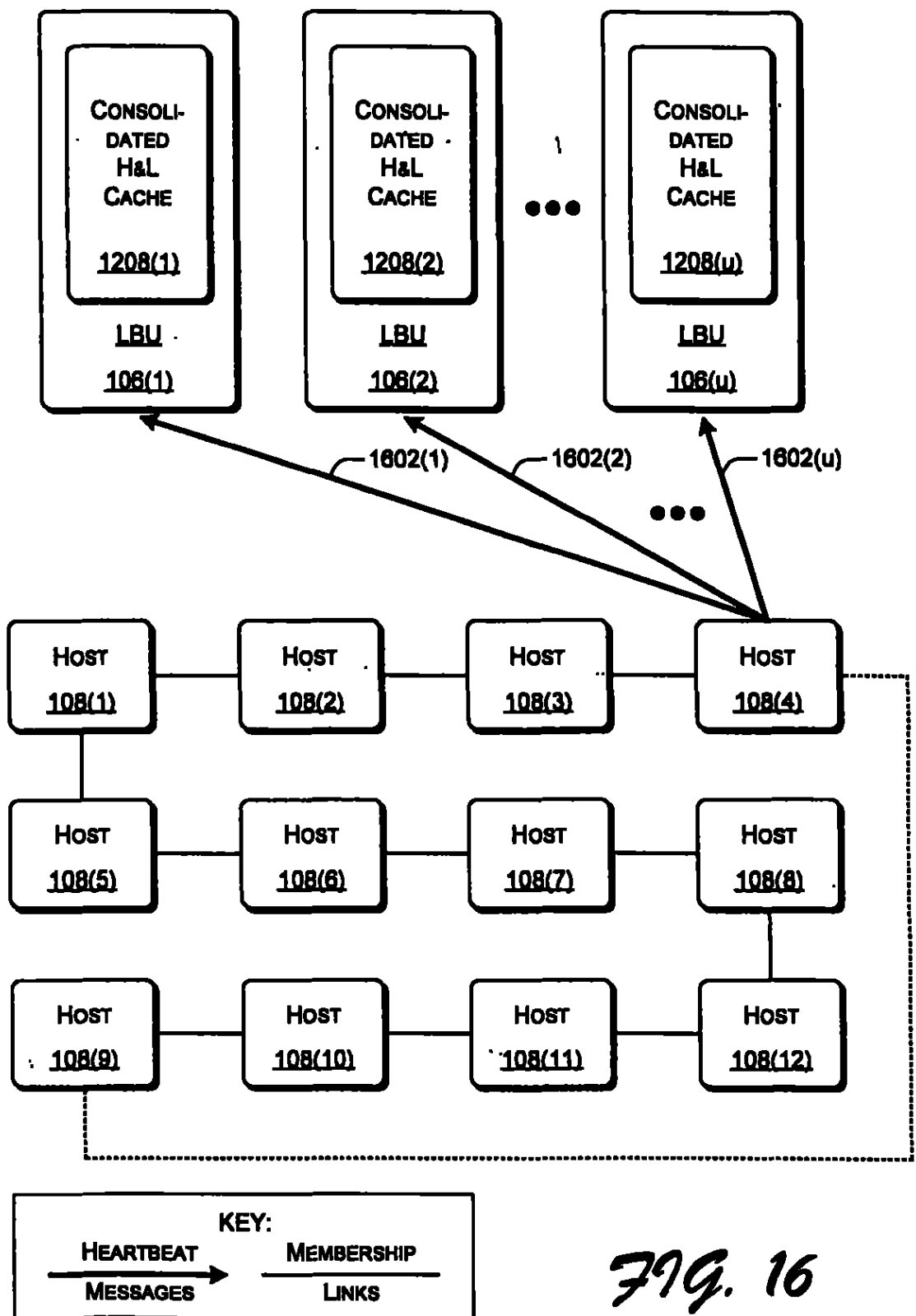


Fig. 16

+

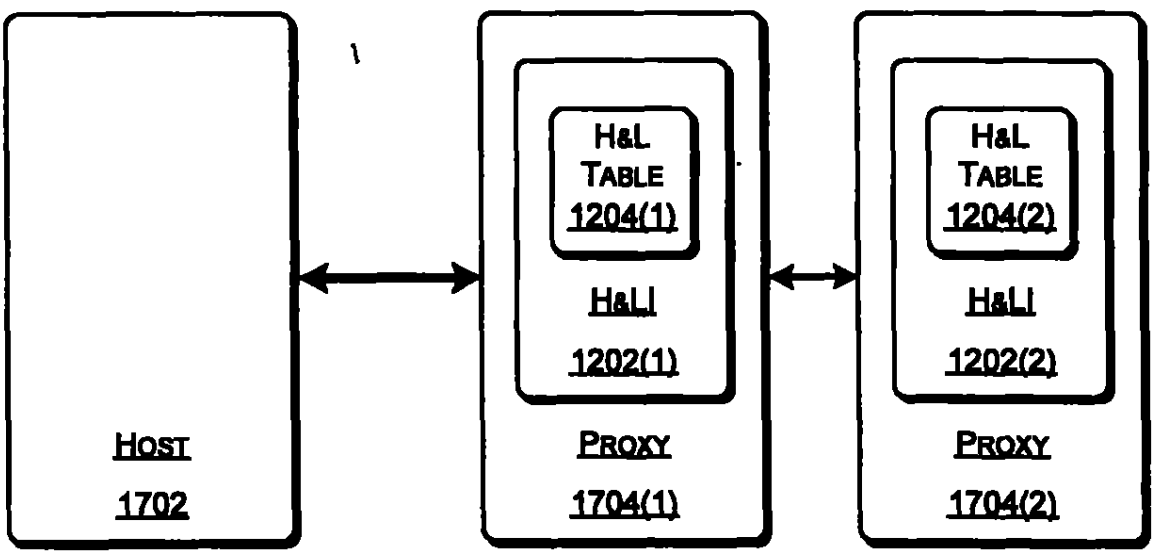


FIG. 17A

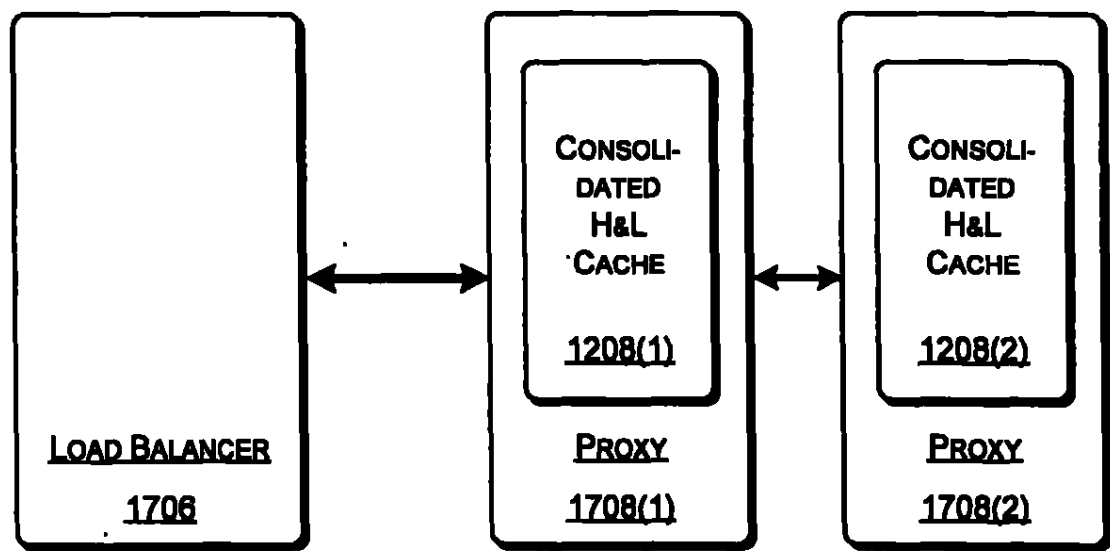


FIG. 17B

528

+

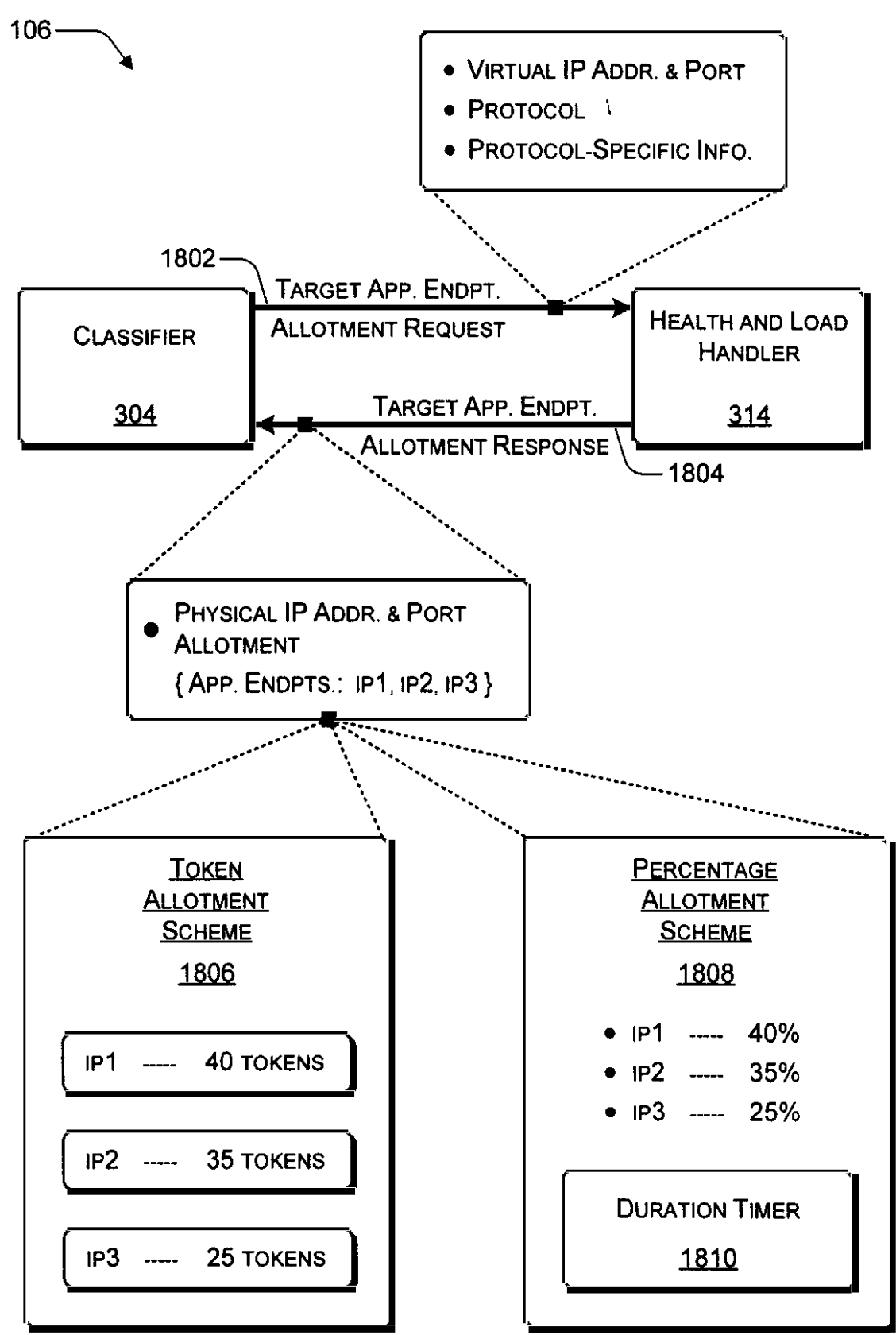


FIG 18

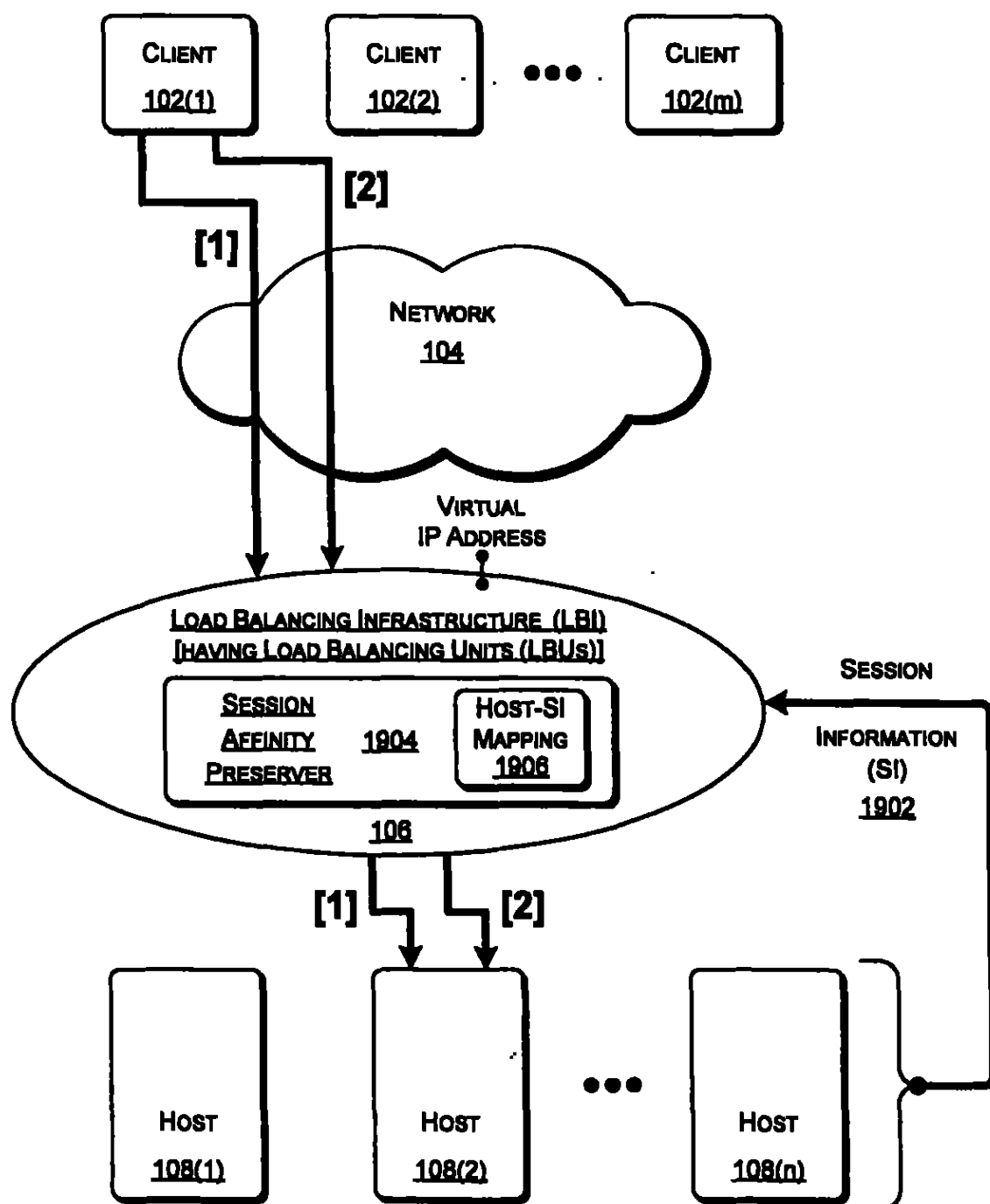


FIG. 19

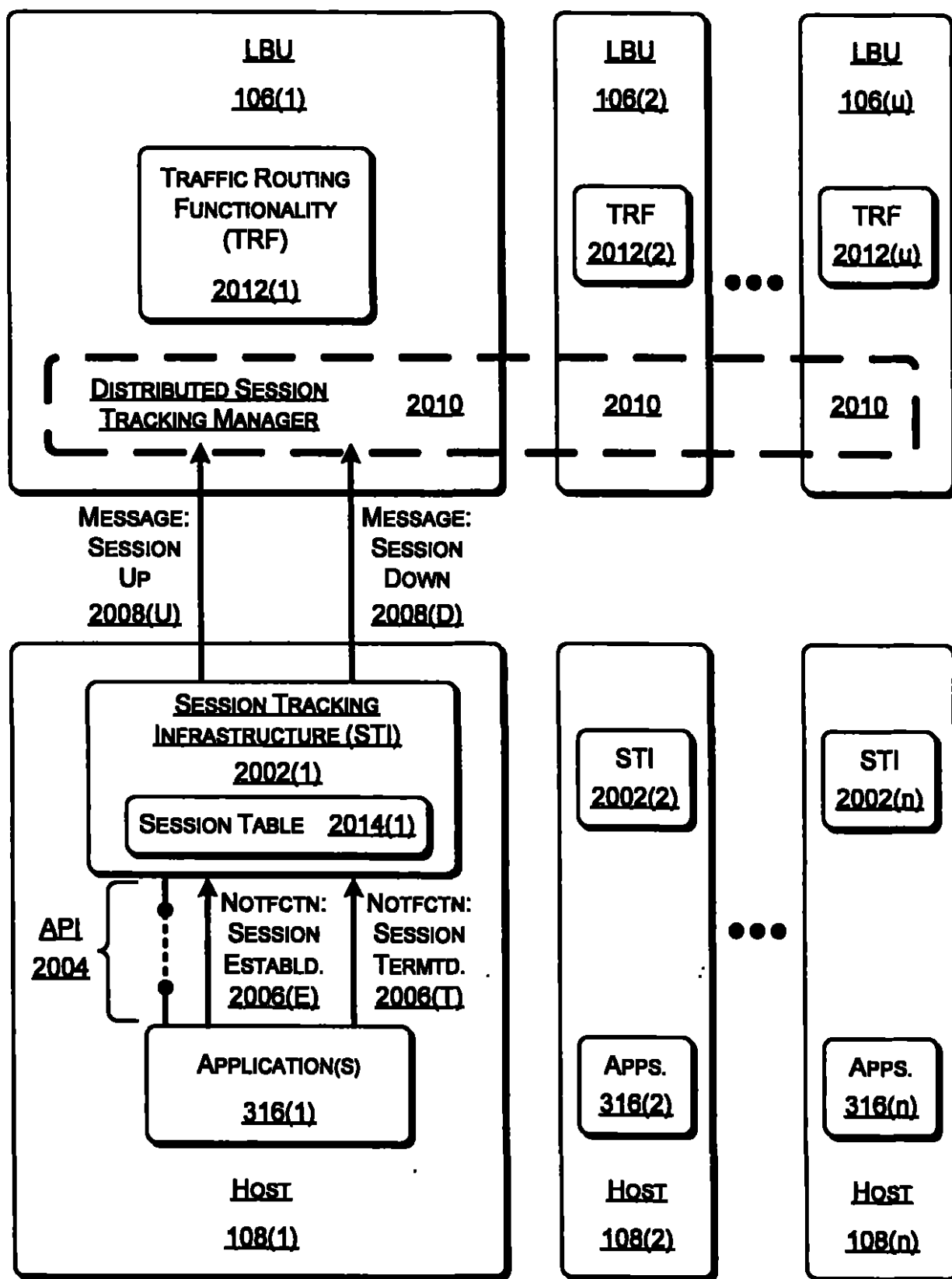


FIG. 20

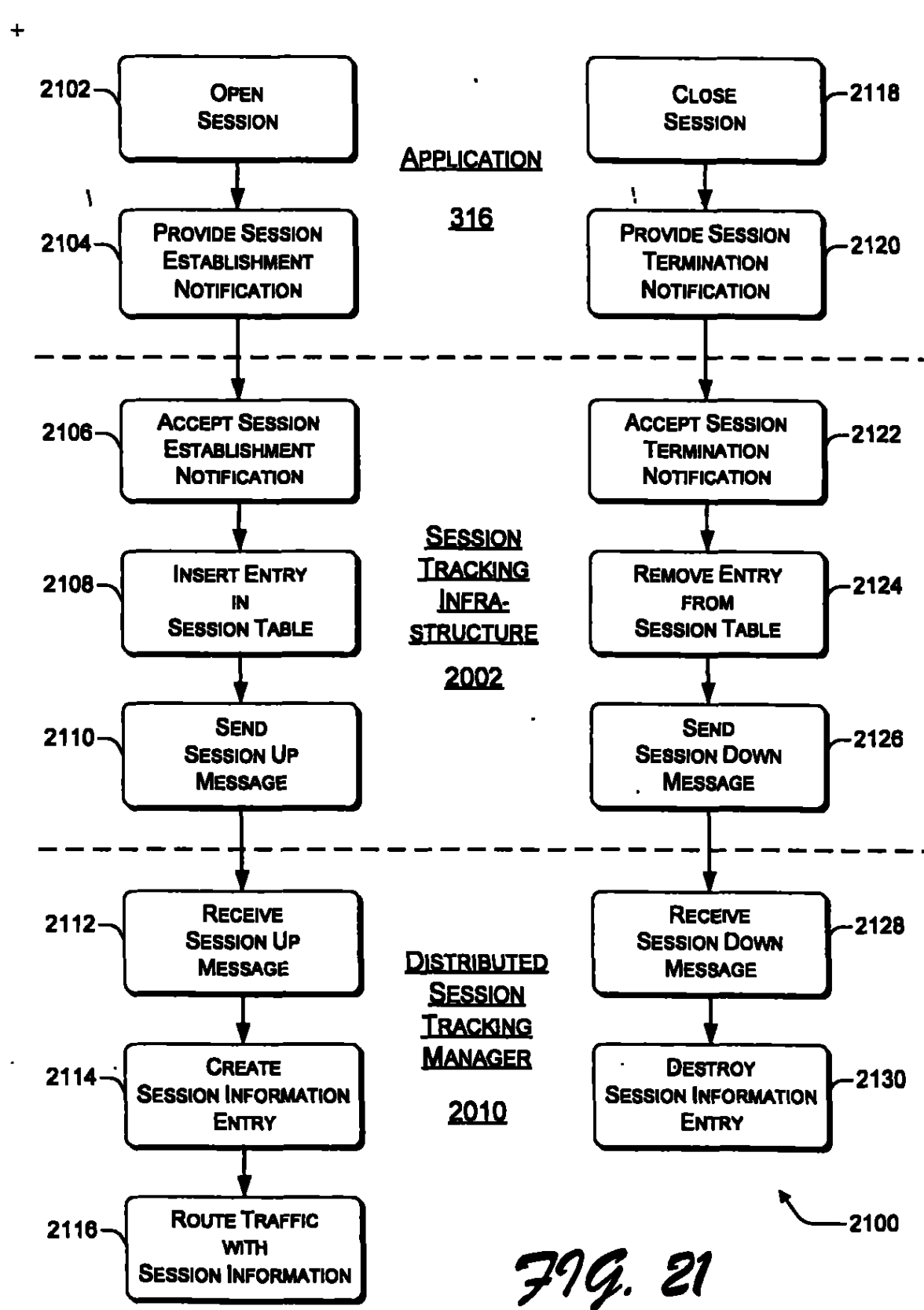


Fig. 21

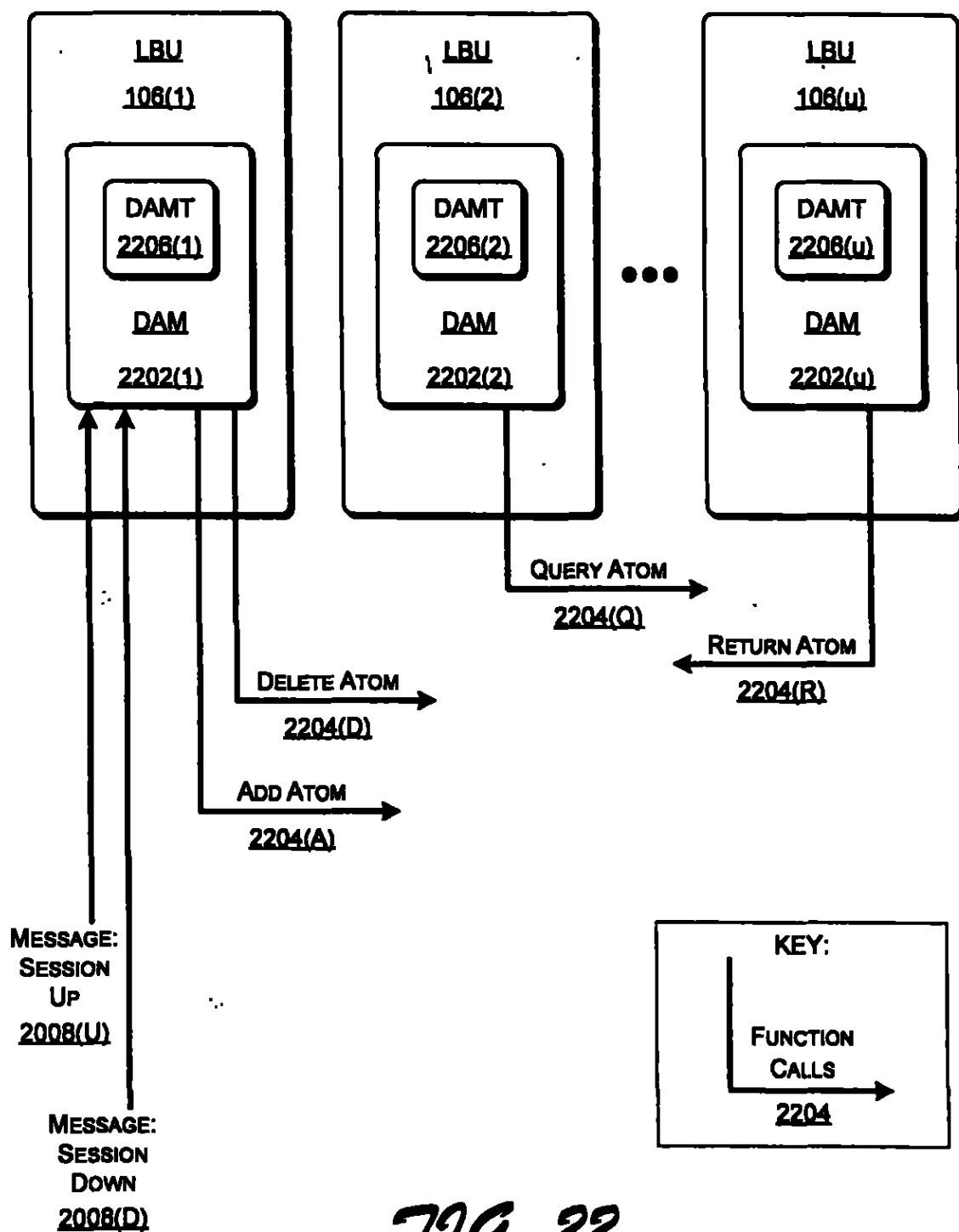


Fig. 22

+

SESSION TABLE		2014
2302(1)	SESSION IDENTIFIER 2302(1I)	SESSION TYPE/APPLIC. 2302(1T)
2302(2)	SESSION IDENTIFIER 2302(2I)	SESSION TYPE/APPLIC. 2302(2T)
⋮	⋮	
2302(v)	SESSION IDENTIFIER 2302(vI)	SESSION TYPE/APPLIC. 2302(vT)

FIG. 23A

DISTRIBUTED ATOM MANAGER (DAM) TABLE (DAMT)			2206
2304(1)	KEY 2304(1K)	DATA 2304(1D)	METADATA 2304(1M)
2304(2)	KEY 2304(2K)	DATA 2304(2D)	METADATA 2304(2M)
⋮	⋮	⋮	
2304(w)	KEY 2304(wK)	DATA 2304(wD)	METADATA 2304(wM)

FIG. 23B

534

+

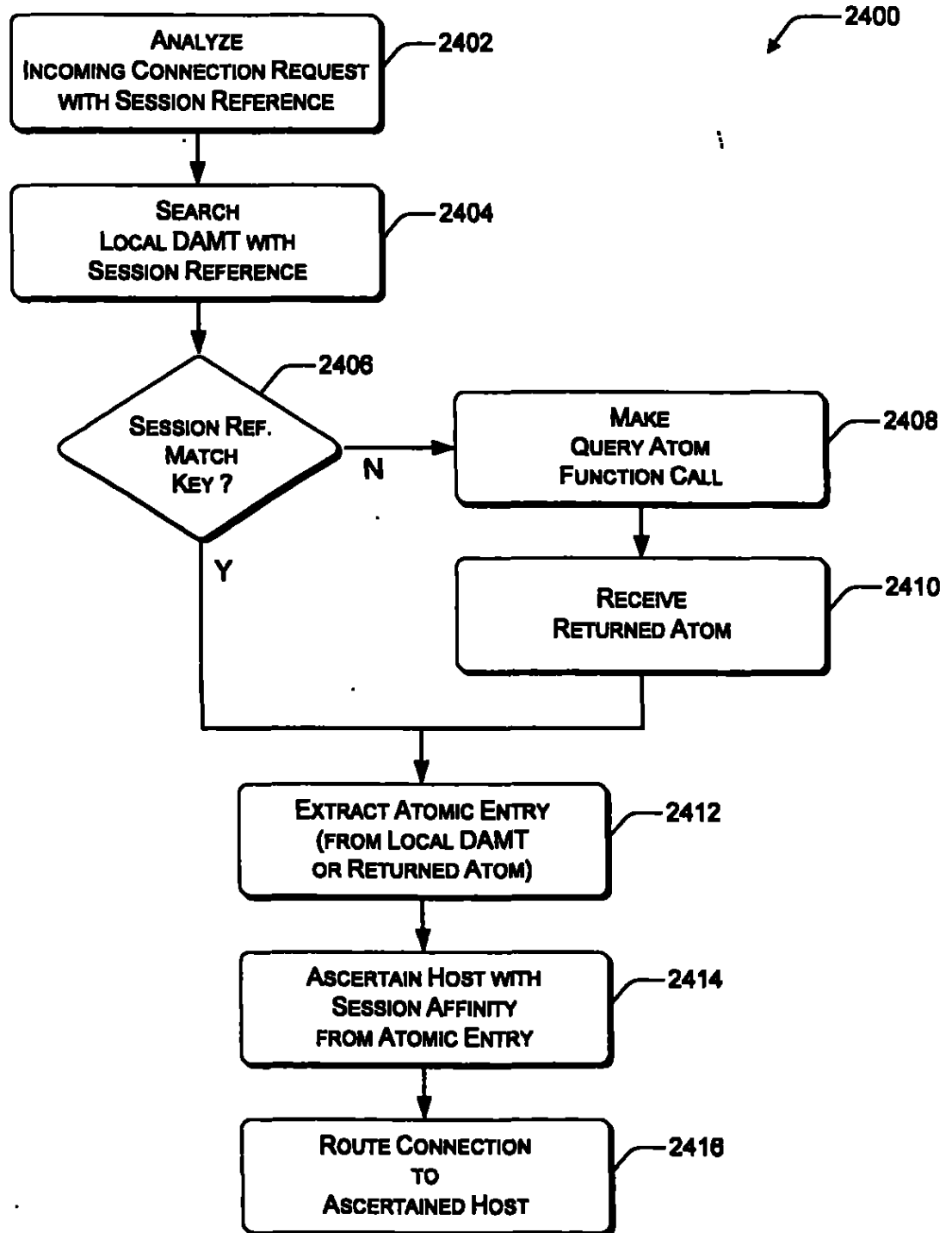
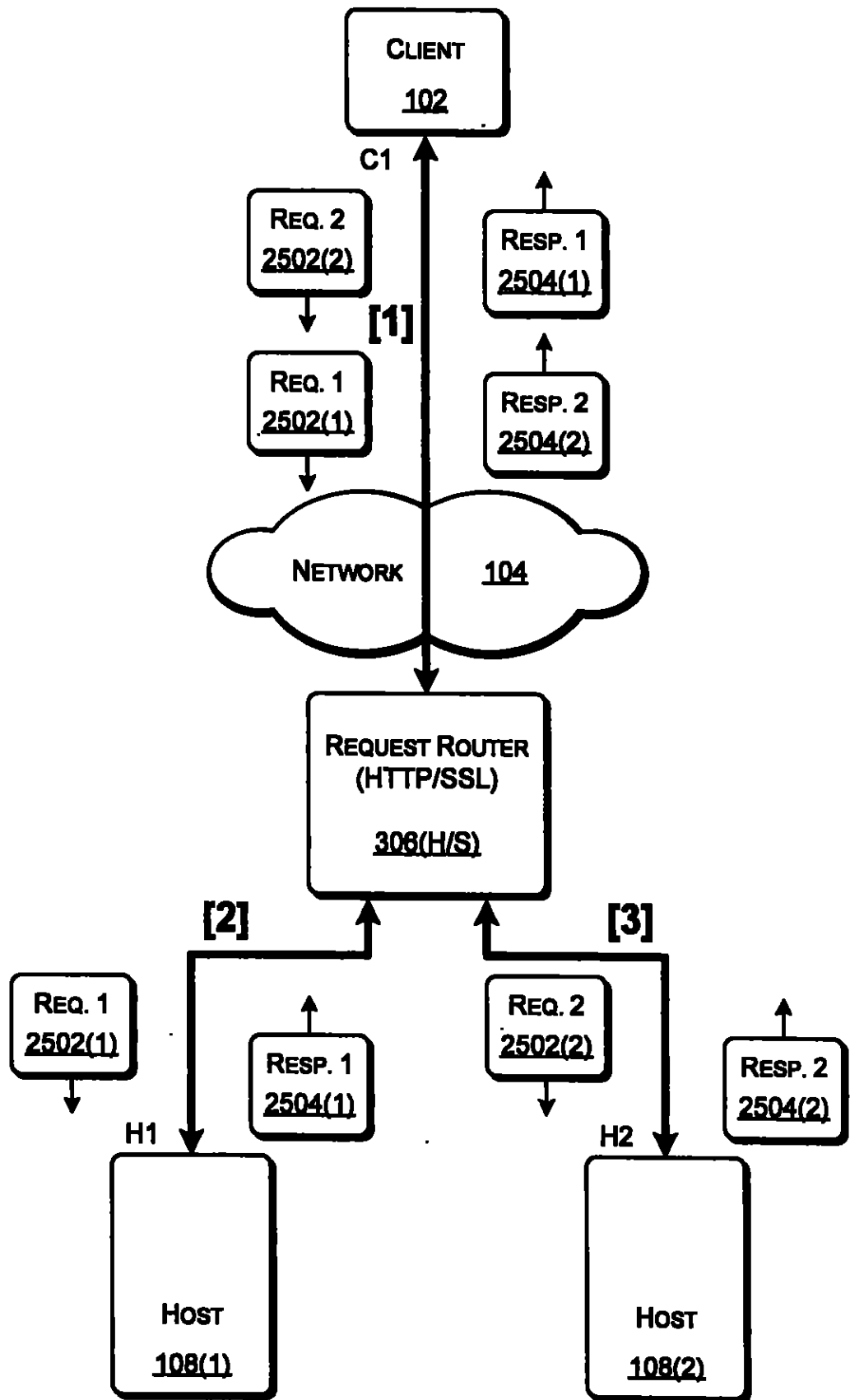


FIG. 24

+

~~335~~



796 25

+

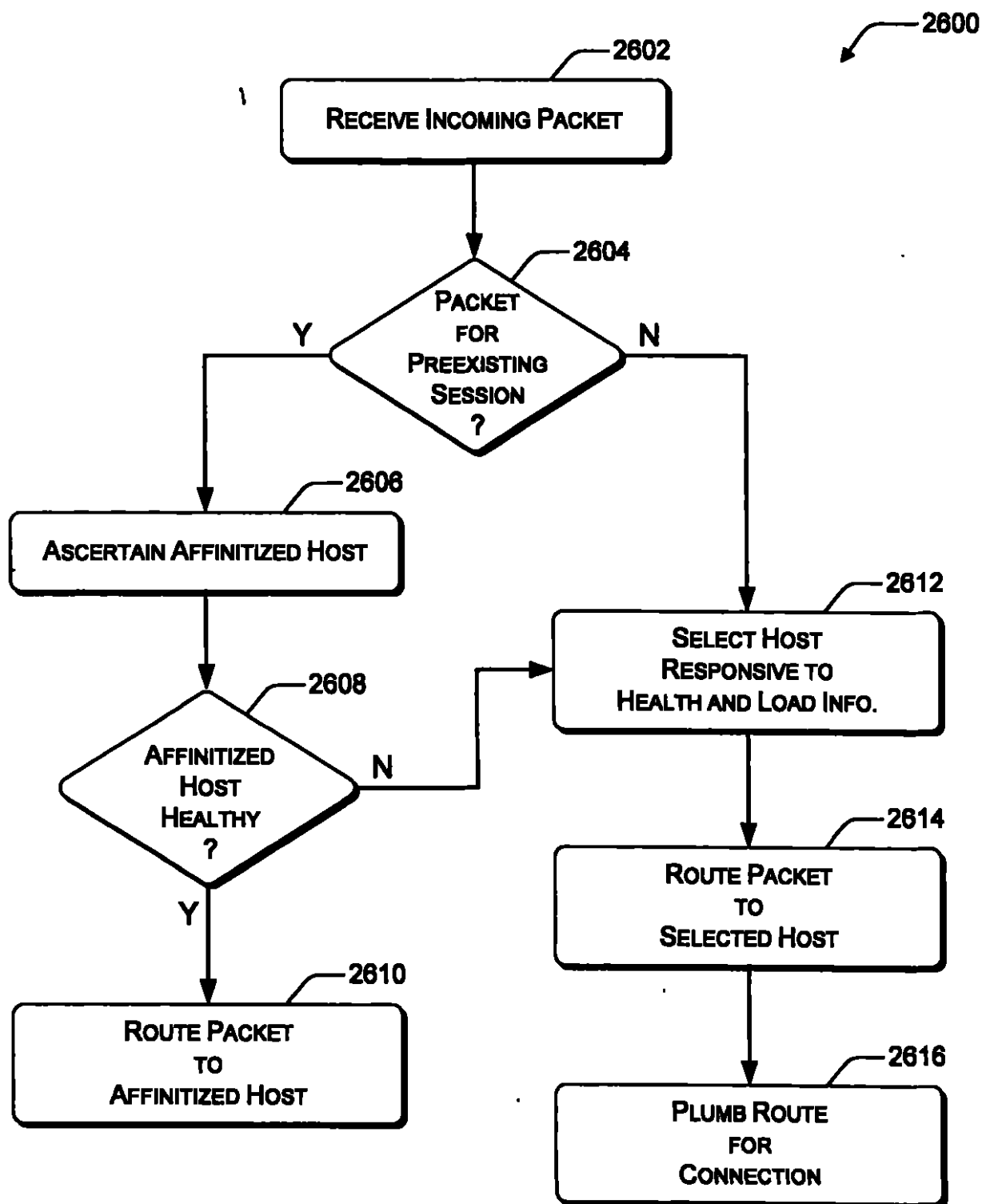


FIG. 26

+

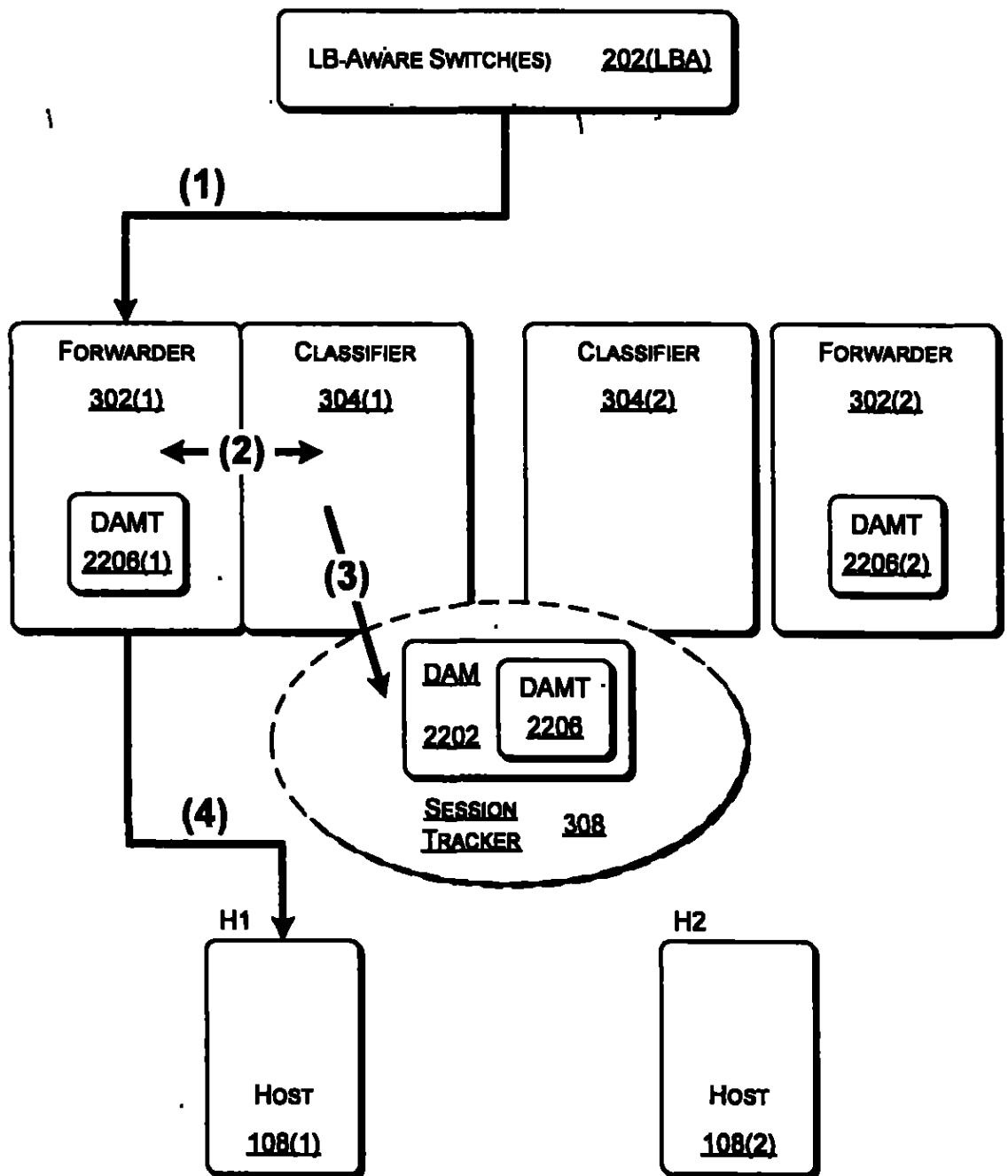


Fig. 27

+

558

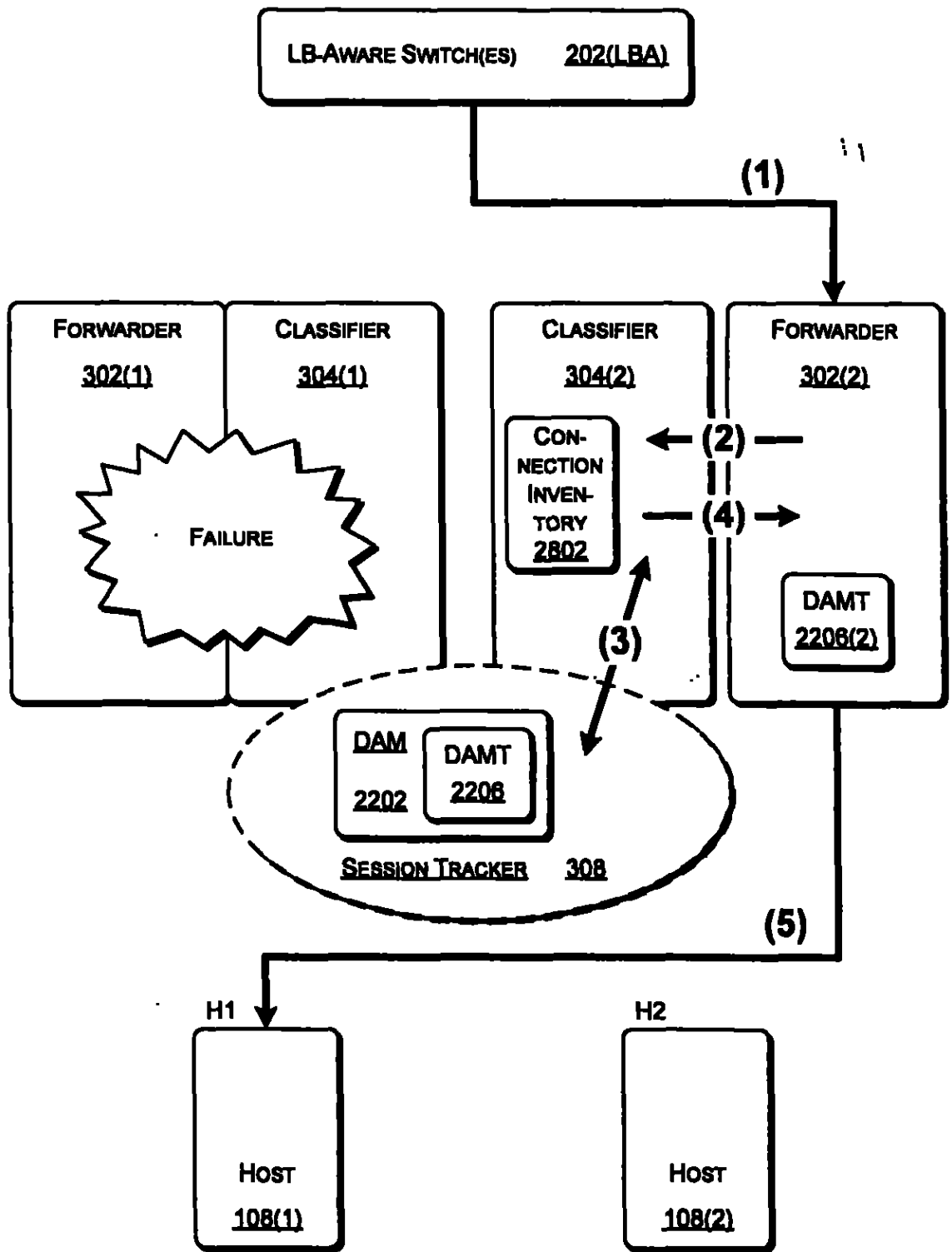


FIG. 28

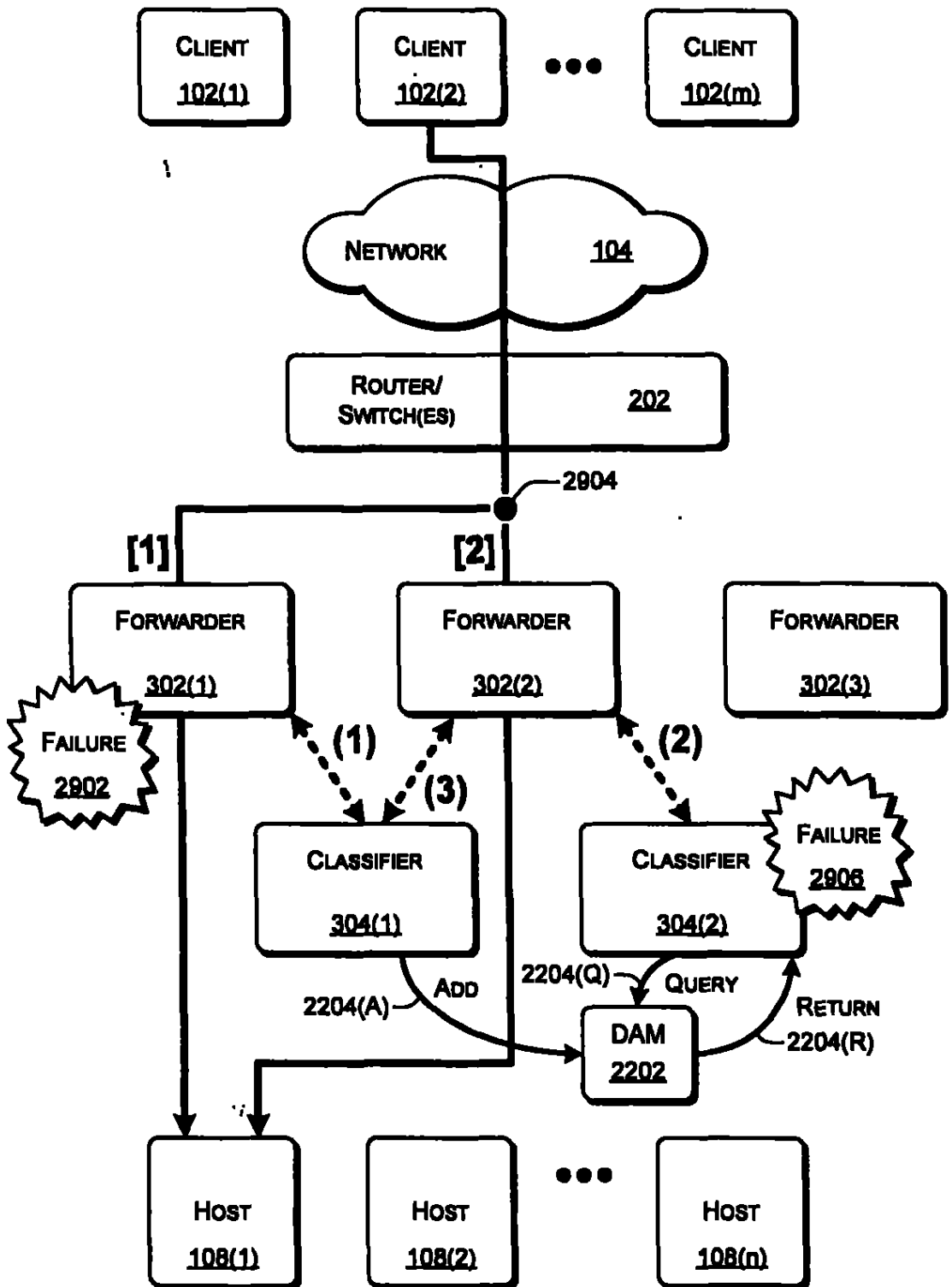


FIG. 29

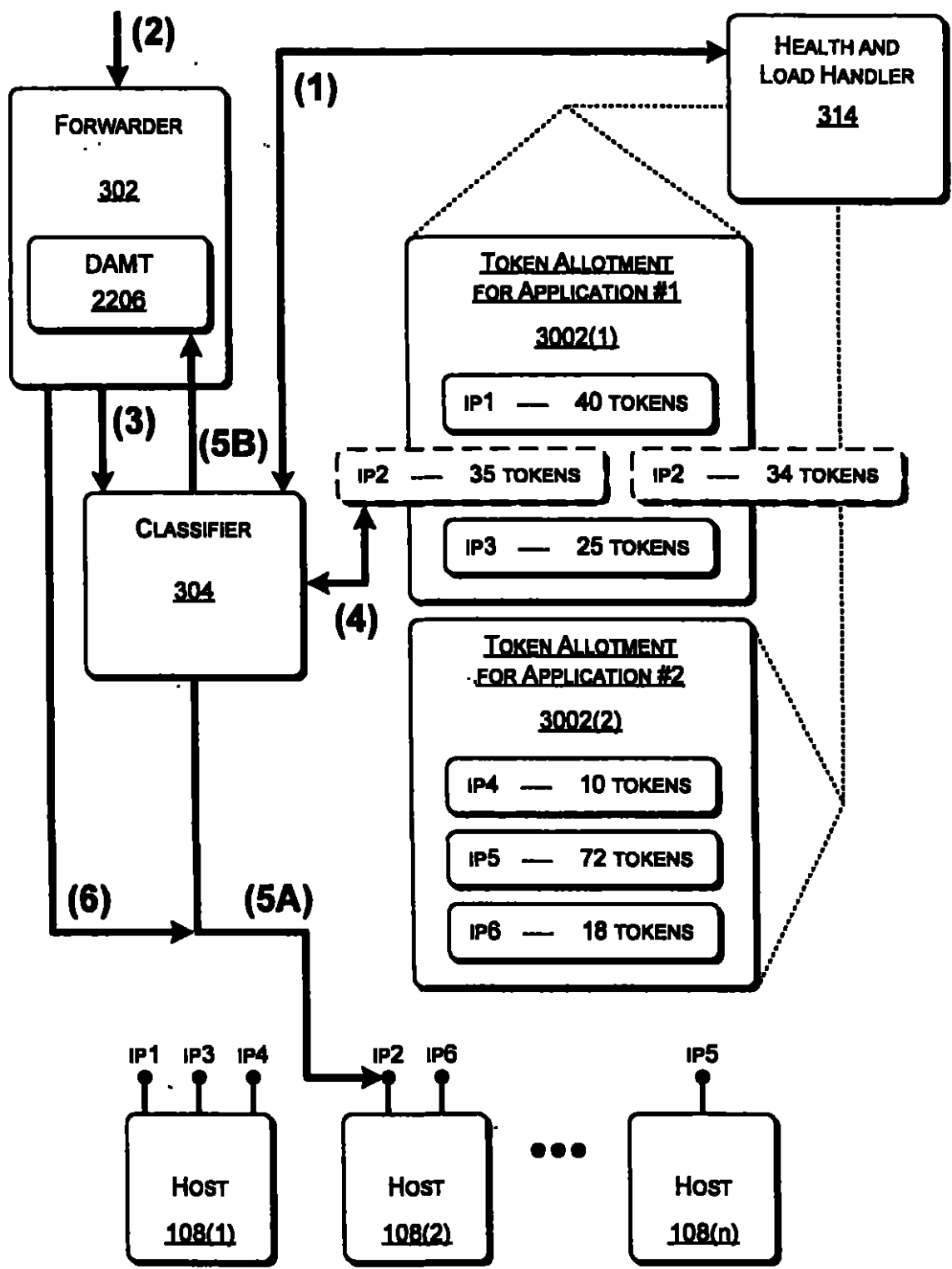


Fig. 30

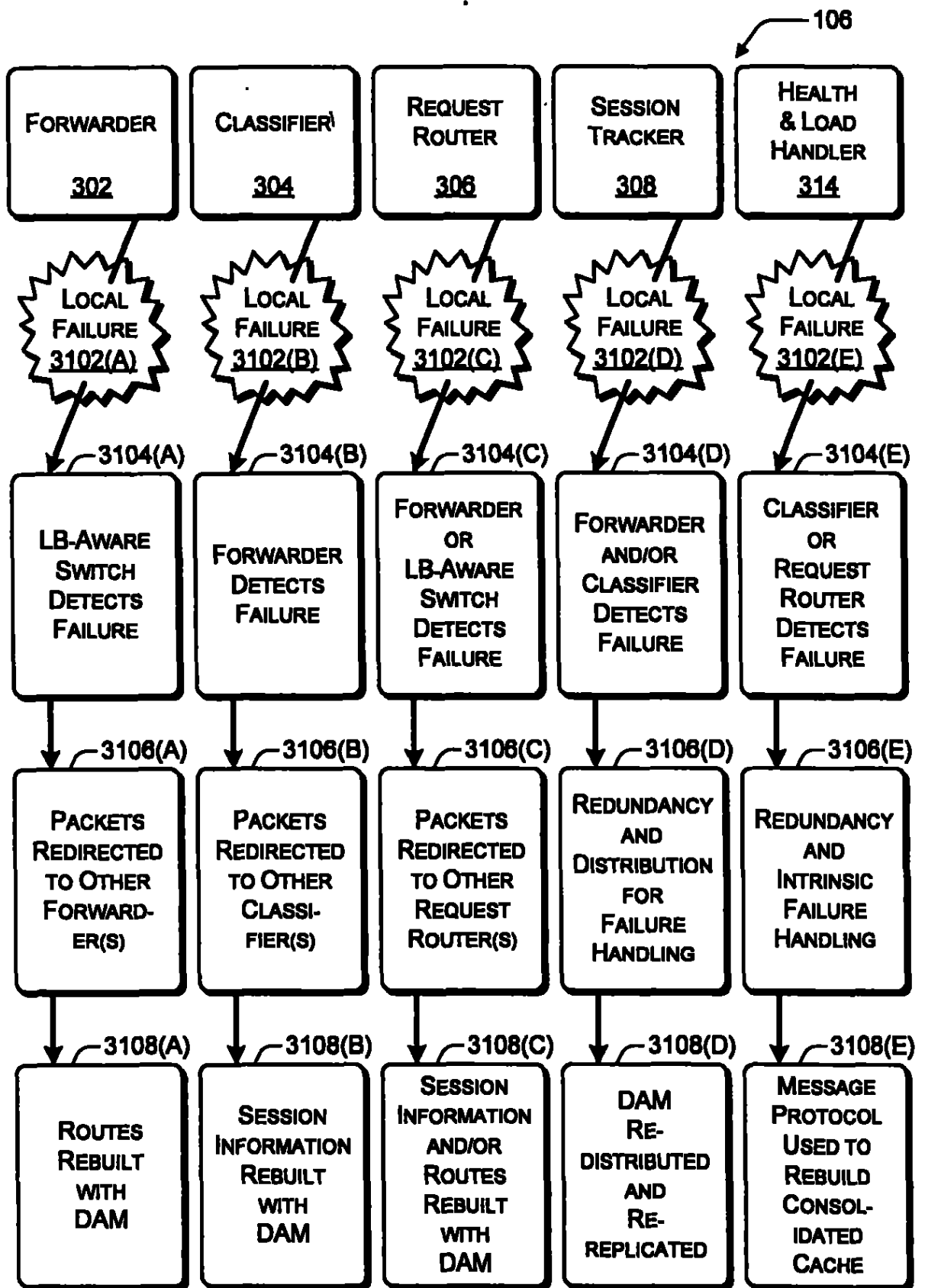


FIG. 21

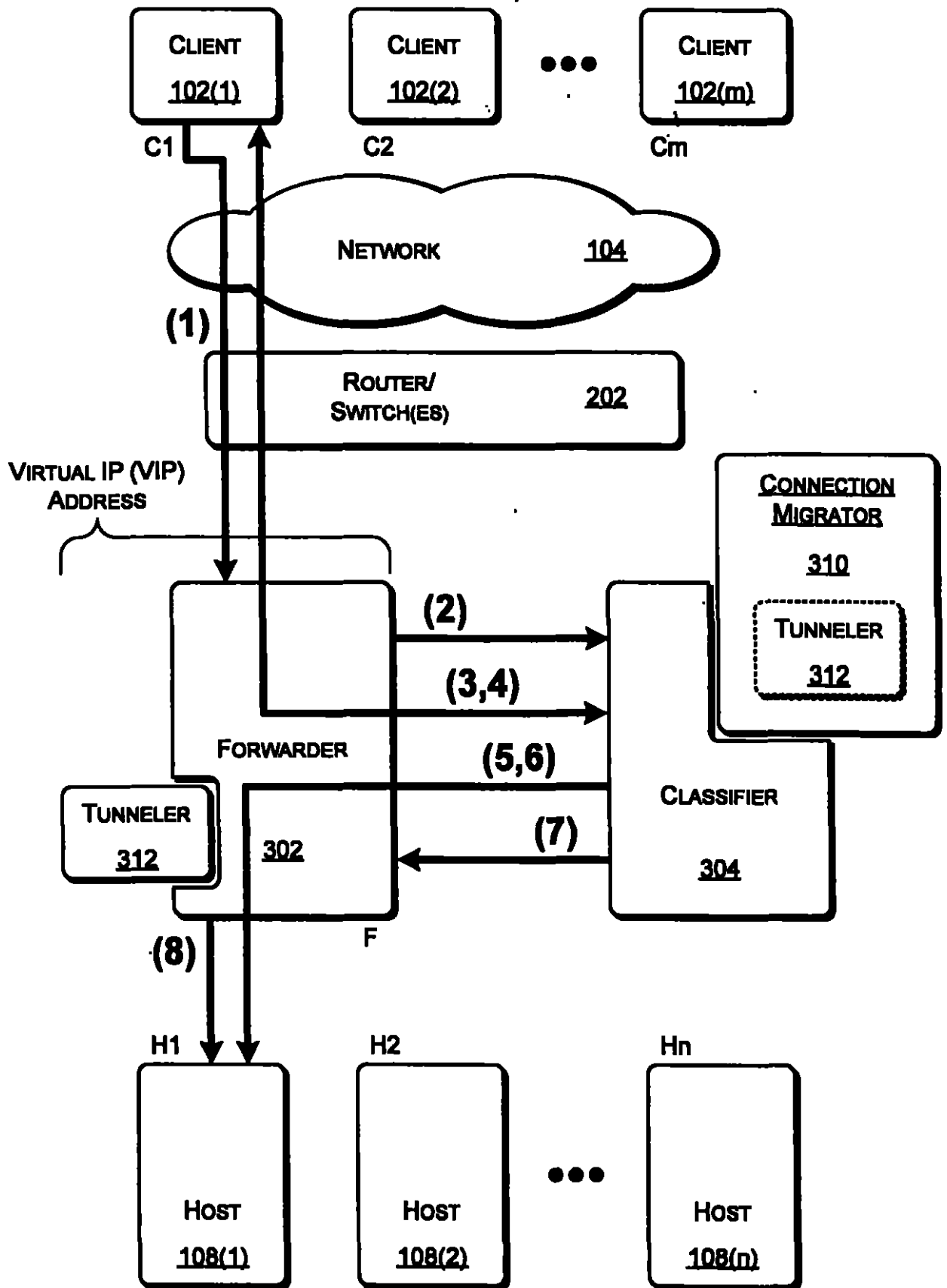


FIG. 29

343

+

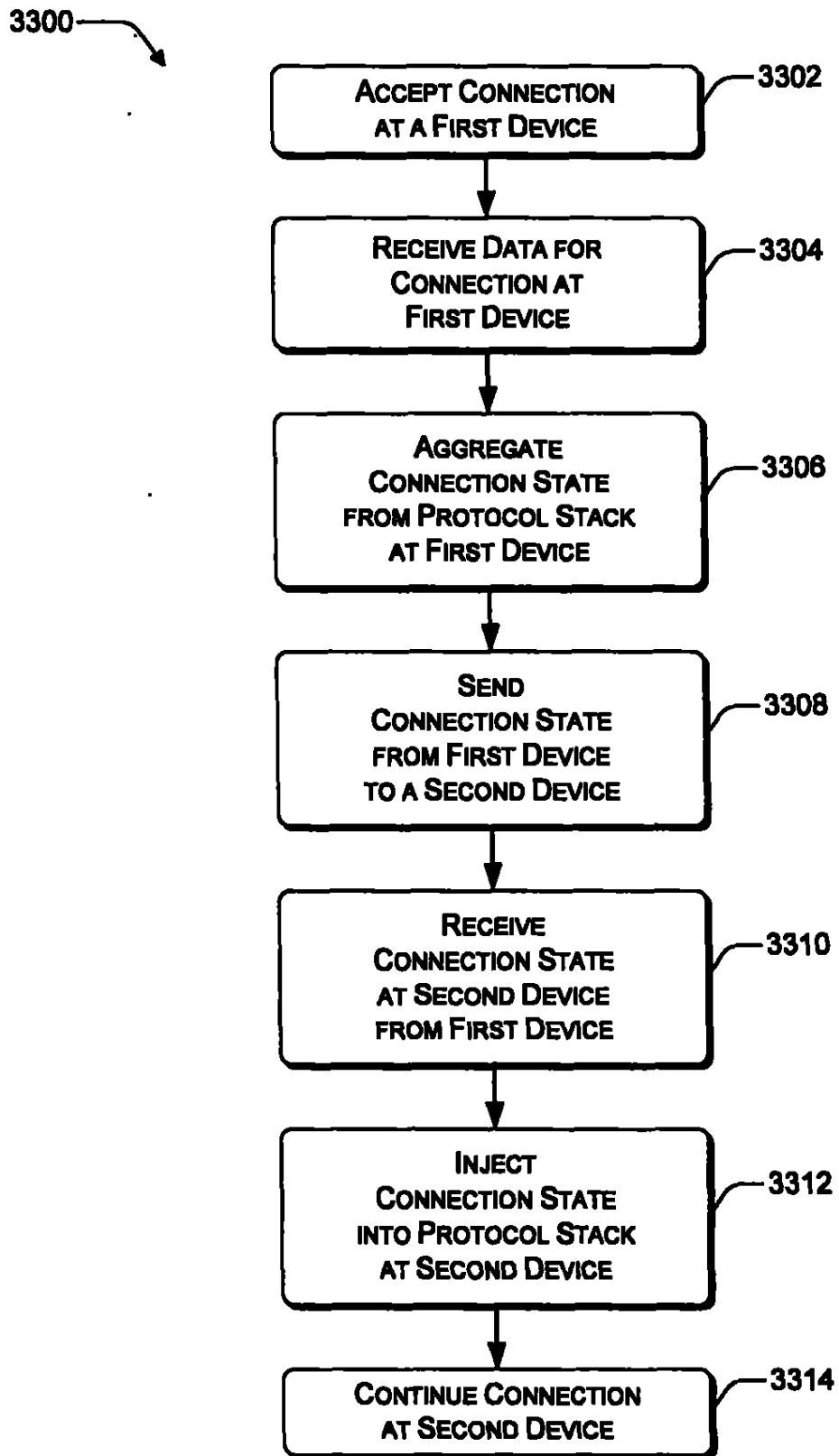


FIG. 33

394

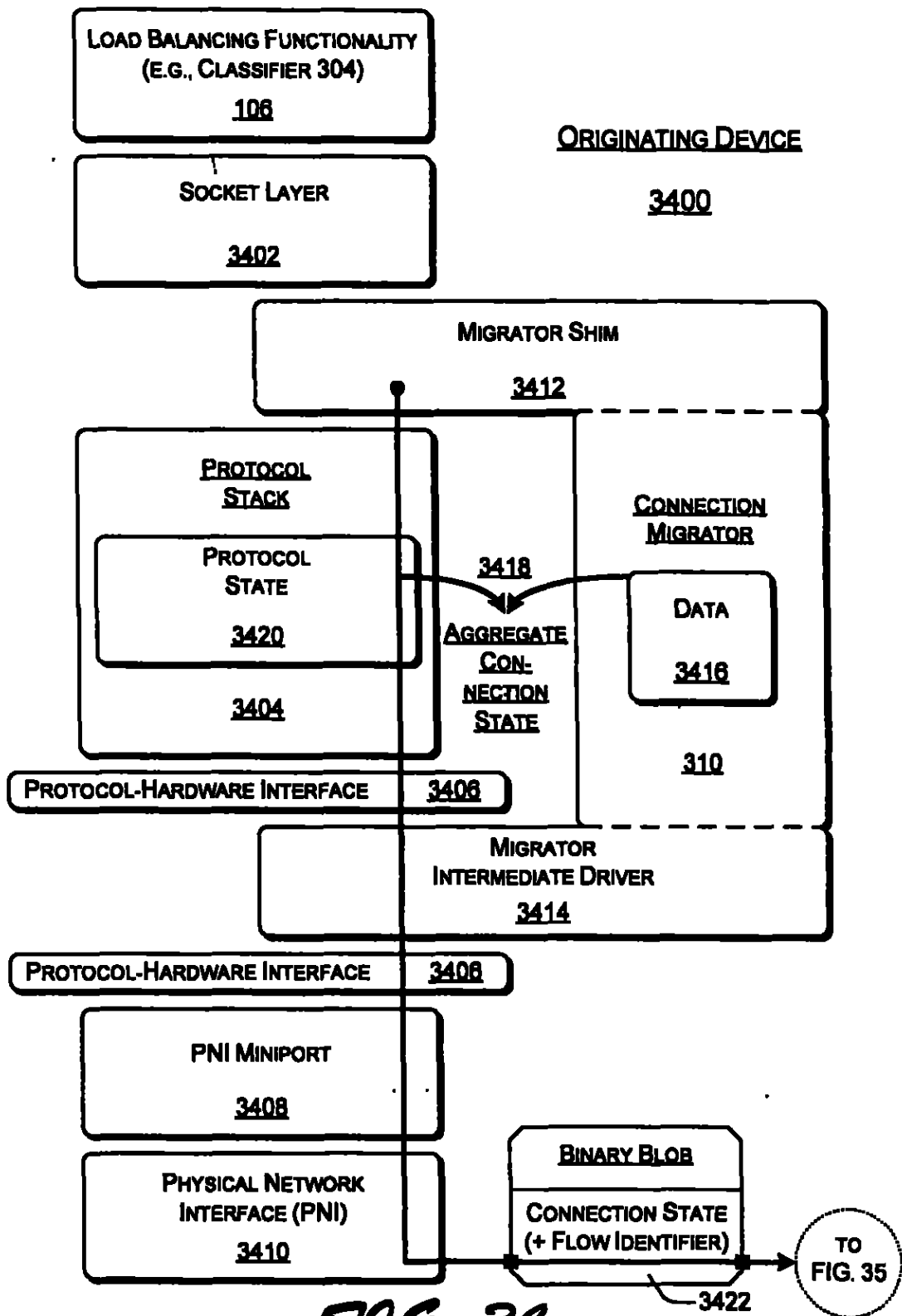


FIG. 34

345

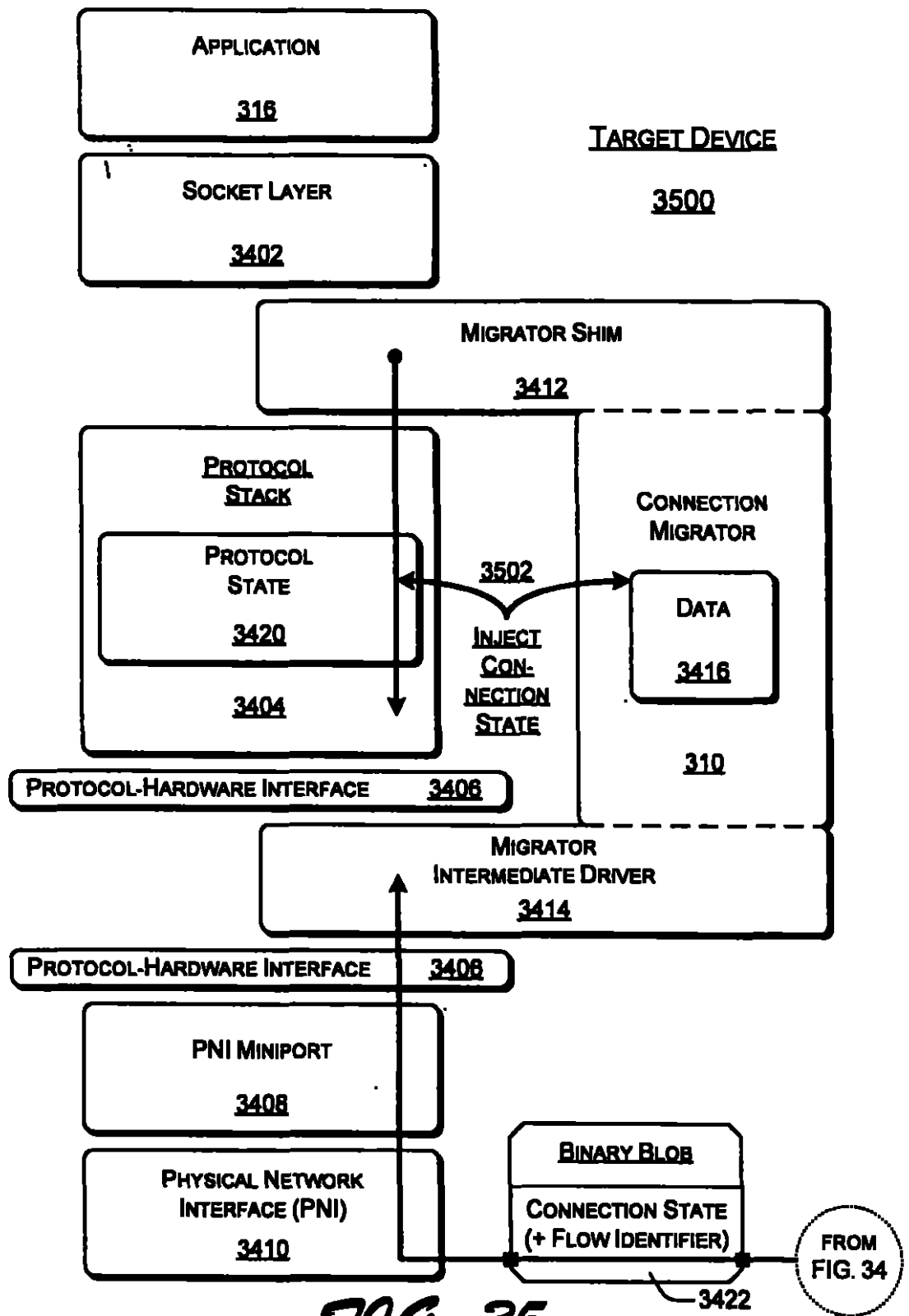


FIG. 35

346

559
559

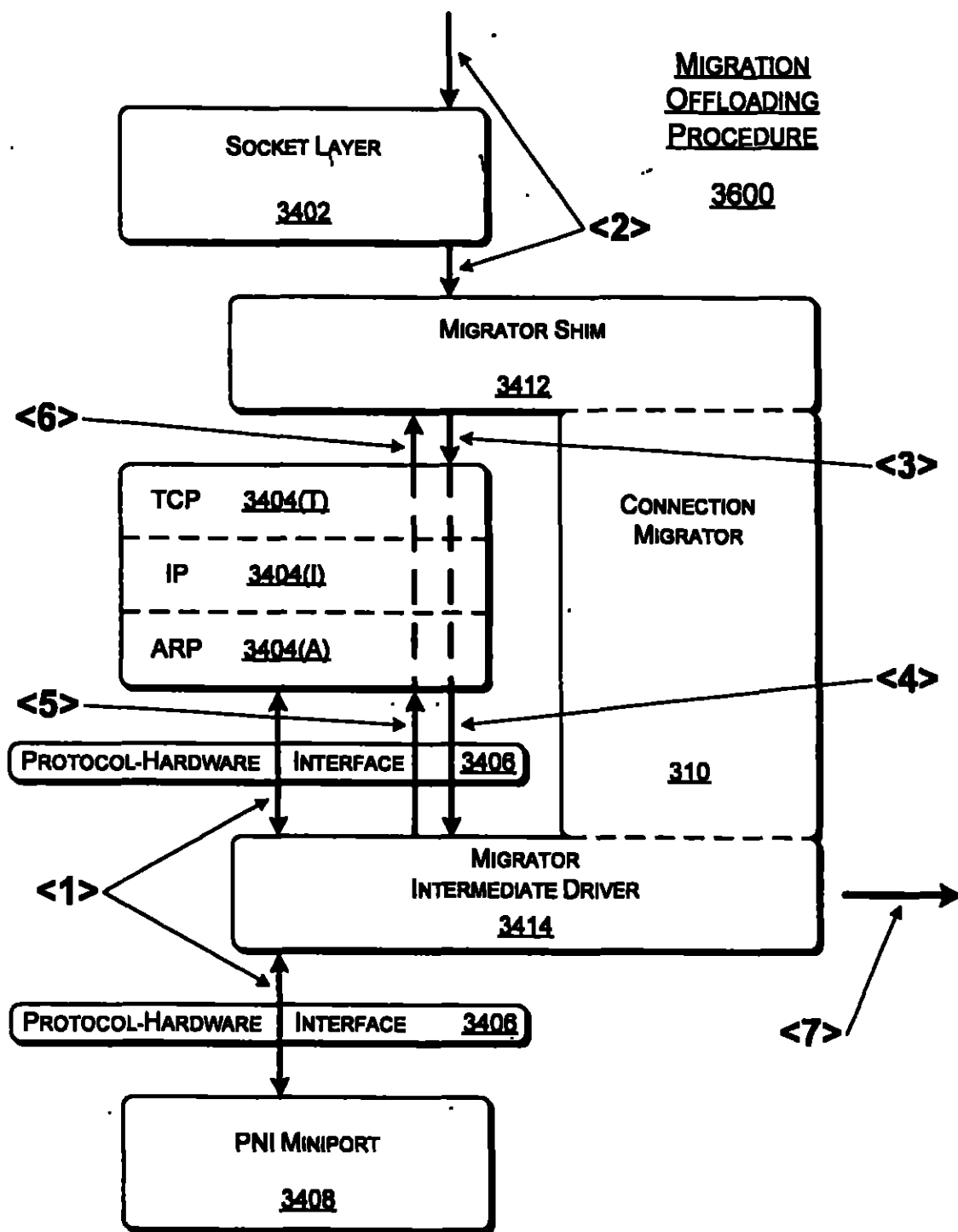


Fig. 36

547

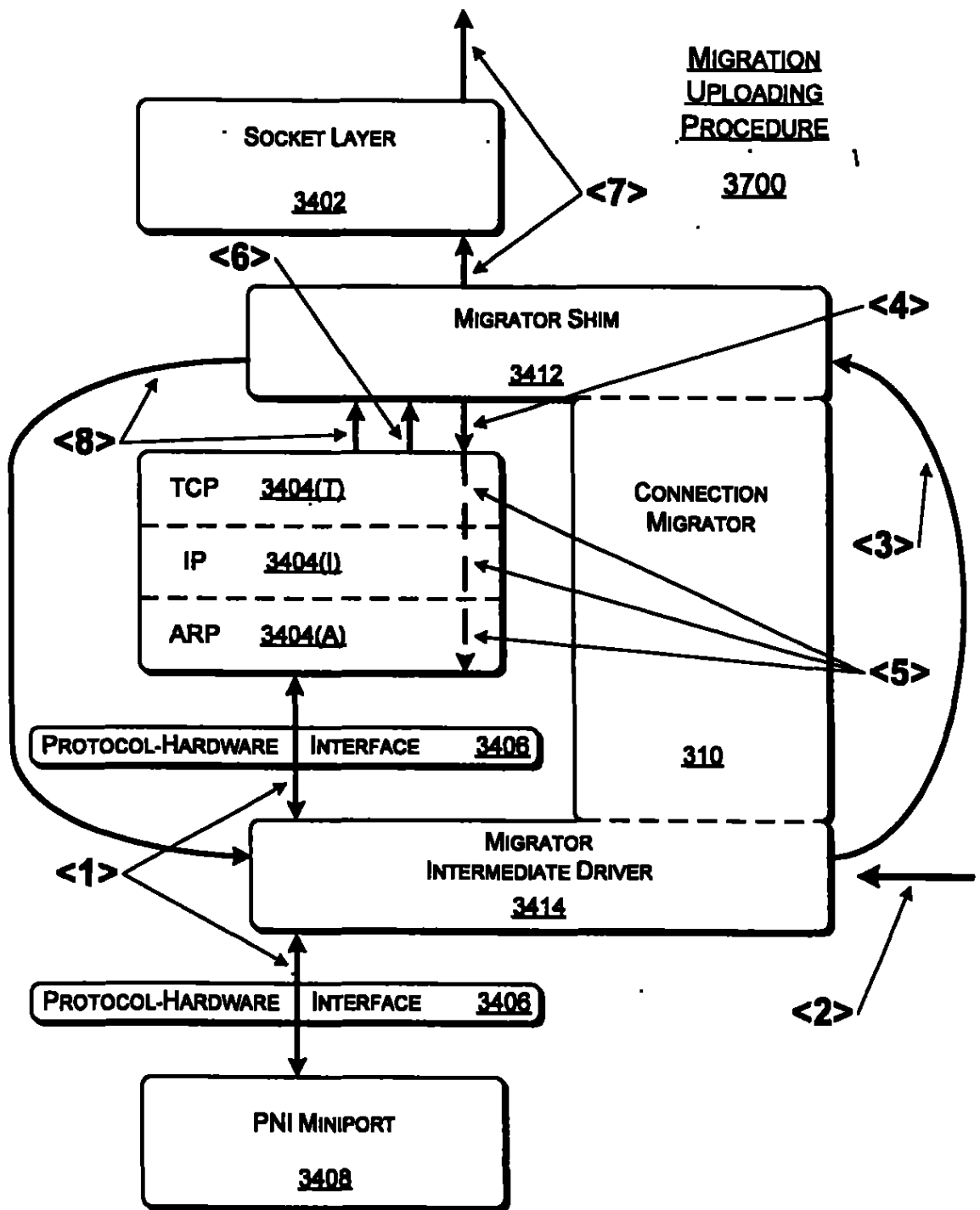


Fig. 37

548

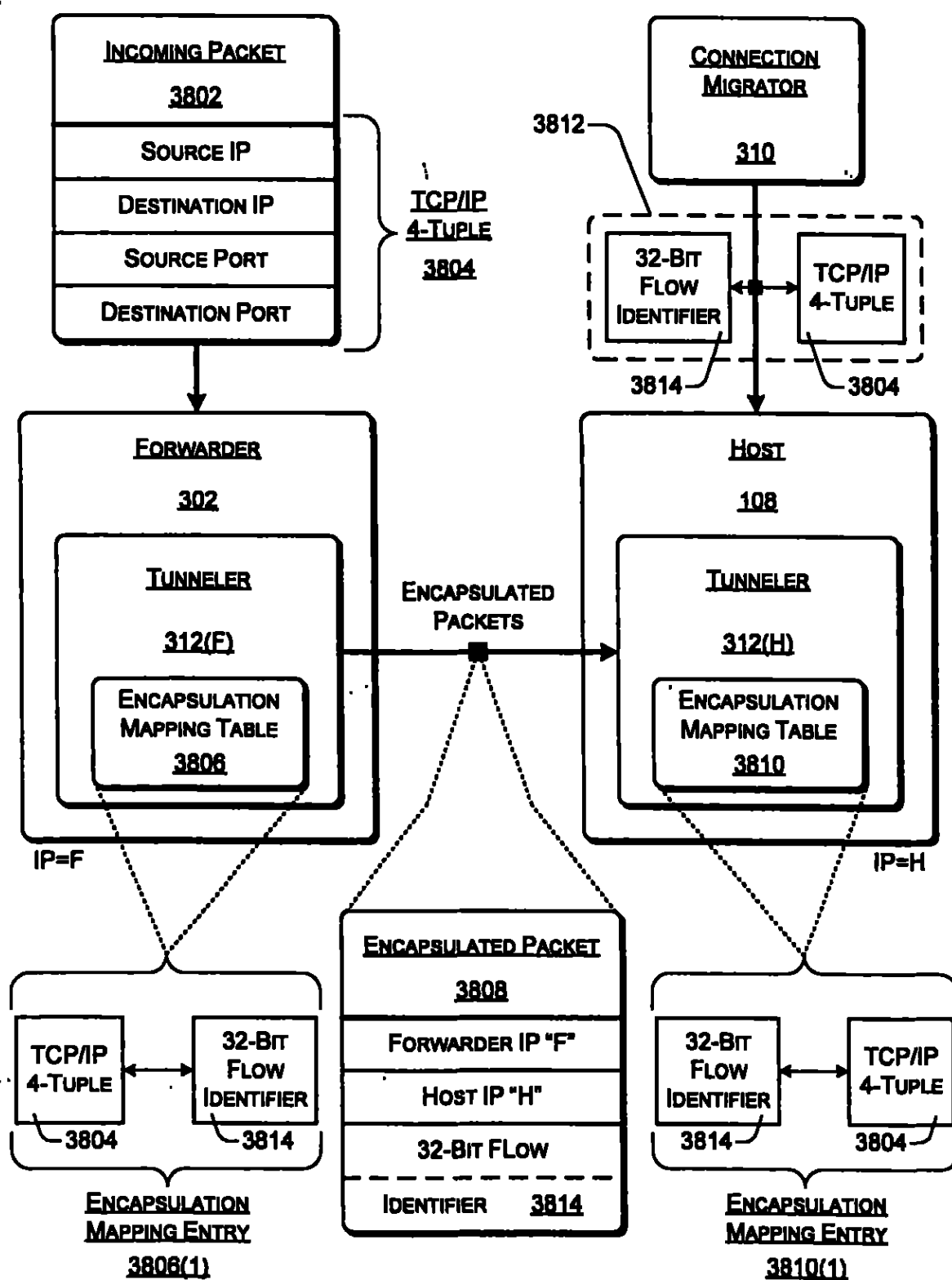


FIG. 38

+

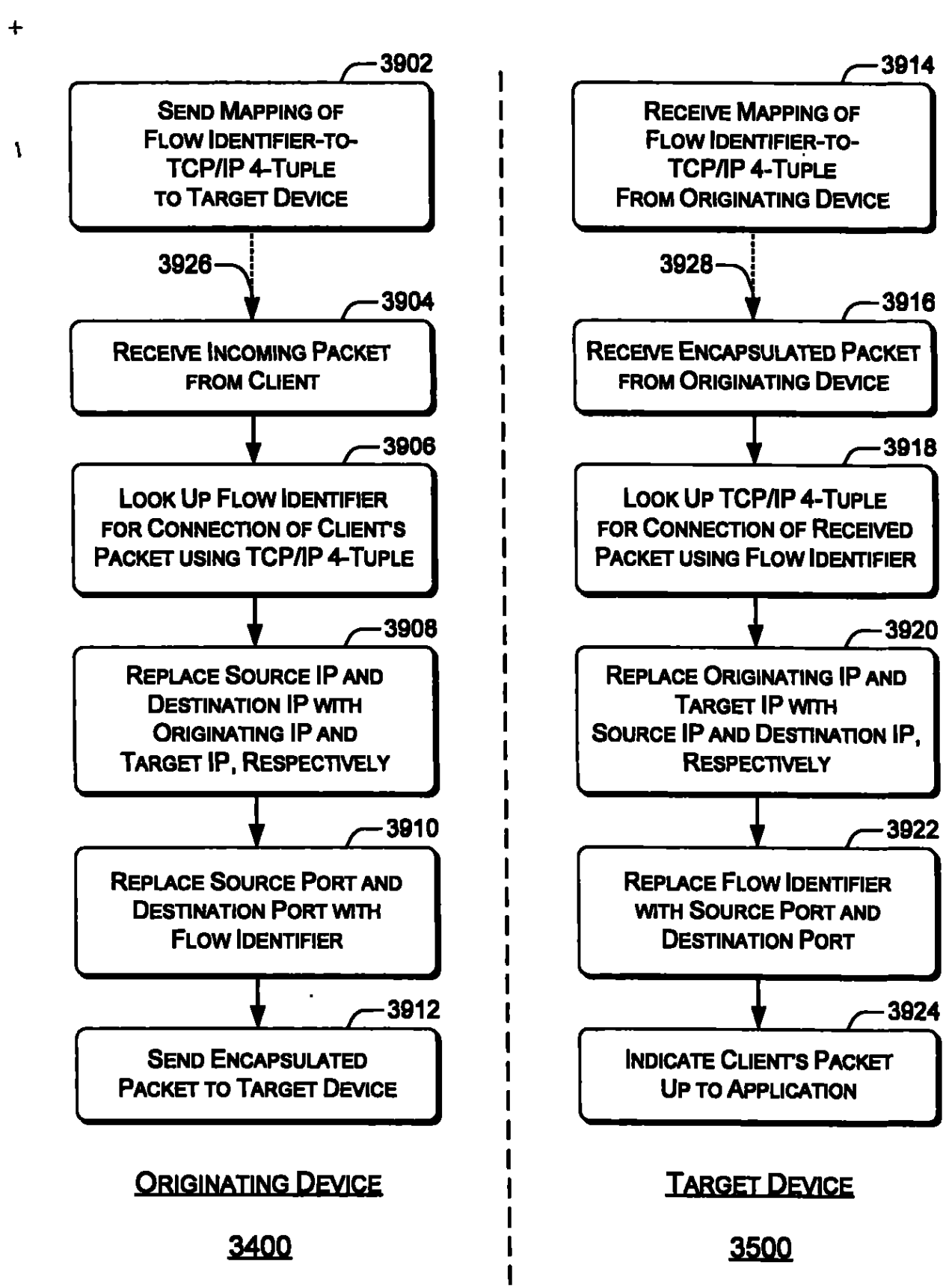


FIG. 39

3900

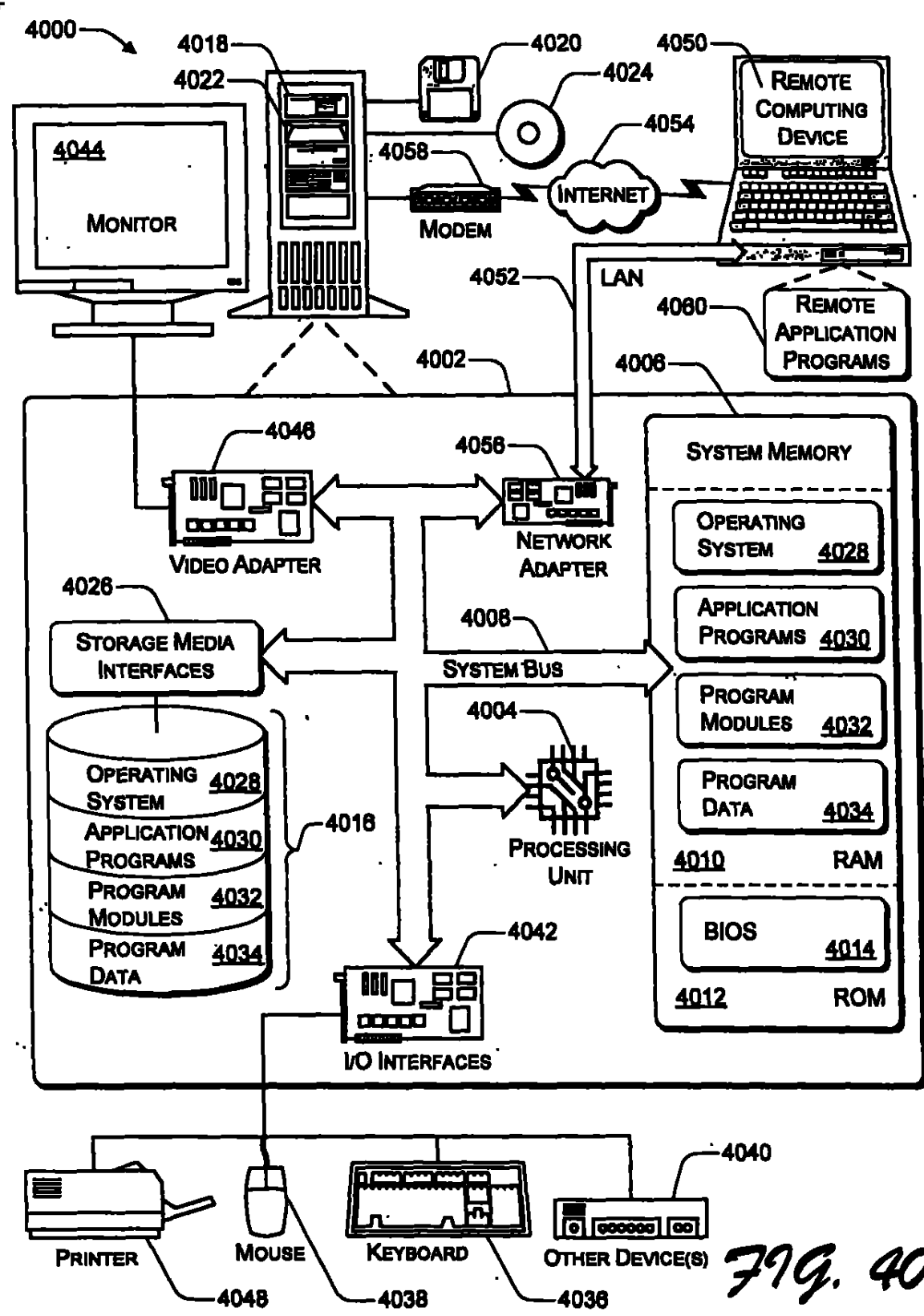


Fig. 40

51

REQUISITOS NECESARIOS PARA PRESENTACION Y AGILIZACION DEL TRAMITE

	USUARIO	RADICADOR
PATENTE DE INVENCION	()	()
MODELO DE UTILIDAD	()	()
- Indicación de que se solicita la concesión de una patente.	()	()
- Datos de identificación del solicitante o de la persona que se presenta la solicitud.	()	()
- Descripción de la invención.	()	()
- Dibujos de ser estos pertinentes.	()	()
- Comprobante de Pago de las tasas establecidas.	()	()

COMPLETA () INCOMPLETA ()

DISEÑO INDUSTRIAL ()

- Indicación de que se solicita el registro de Diseño Industrial	()	()
- Datos de identificación del solicitante o de la persona que presenta la solicitud.	()	()
- Representación gráfica o fotográfica del Diseño Industrial o muestra del Material que incorpora el diseño.	()	()
- Comprobante de pago de las tasas establecidas	()	()

COMPLETA () INCOMPLETA ()

ESQUEMA DE TRAZADO ()

- Indicación de que solicita el registro de un Esquema de trazado.	()	()
- Datos de identificación del solicitante o de la persona que presenta La solicitud.	()	()
- Representación gráfica del esquema de trazado	()	()
- Comprobante de pago de las tasas establecidas.	()	()

COMPLETA () INCOMPLETA ()

Firma Responsable.

Fecha:

11/13



566

Industria y Comercio
SUPERINTENDENCIAFolio 554**2020**
Bogotá DC**Doctor(a)**
DARIO CARDENAS NAVAS
Apoderado(a)

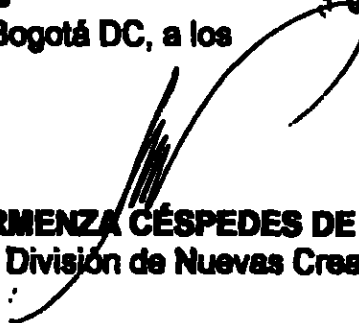
REFERENCIA:	Radicación No.	04-61217
	Trámite	002
	Evento	001
	Actuación	416
	Oficio No.	1240
	Folios	001

Teniendo en cuenta que la solicitud de privilegio de patente que se tramita bajo el expediente indicado en la referencia, reúne los requisitos de forma establecidos en los artículos 26 y 27 de la Decisión 486 de la Comisión de la Comunidad Andina, y en las demás disposiciones vigentes sobre la materia, se envía el extracto de la solicitud a la Oficina de Comunicaciones para efecto de su publicación en la Gaceta de Propiedad Industrial.

De acuerdo con lo previsto por el artículo 40 de la Decisión 486/00, la publicación en este caso se hará a partir del día 28 de diciembre de 2005.

Cúmplase**Dado en Bogotá DC, a los**

30 JUL 2004


ALIX CARMENZA CÉSPEDES DE VERGEL
Jefe de la División de Nuevas Creaciones

Carrera 13 No. 27-00 Pisos 5, 7 y 10
Conmutador: 382 0840
Fax: 350 5220
E-mail: Info@sic.gov.co
www.sic.gov.co
Bogotá, D.C., Colombia