US 20110082947A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2011/0082947 A1

(43) Pub. Date: Apr. 7, 2011

Szeto et al.

(54) CONNECTION RATE LIMITING

(75) Inventors: **Ronald W. Szeto**, Pleasanton, CA (US); **David Chun Ying Cheung**, San Jose, CA (US); **Rajkumar Jalan**, Saratoga, CA (US)

(73) Assignee: **Foundry Networks, Inc., a Delaware Corporation**

(21) Appl. No.: **12/723,615**

(22) Filed: **Mar. 12, 2010**

Related U.S. Application Data

(63) Continuation of application No. 10/139,073, filed on May 3, 2002, now Pat. No. 7,707,295.

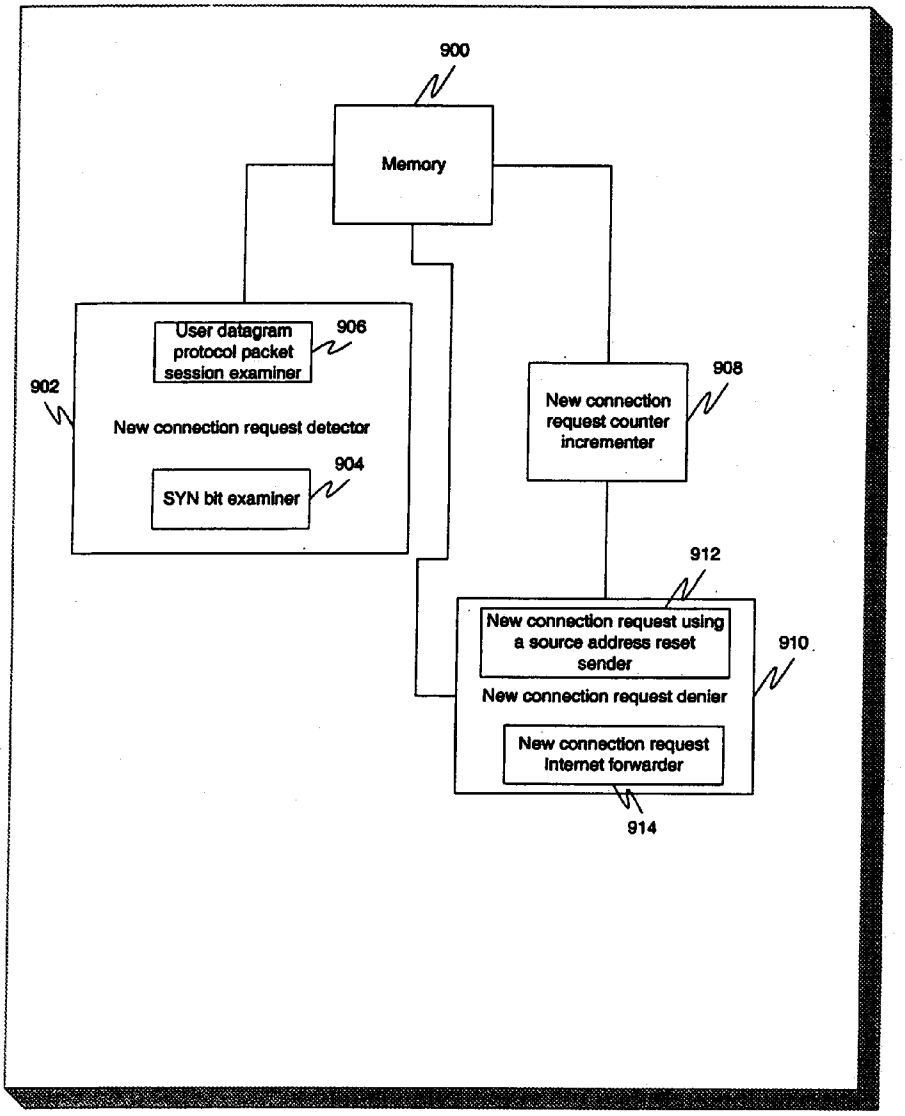
Publication Classification

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/232**

(57) **ABSTRACT**

Each service in a computer network may have a connection rate limit. The number of new connections per time period may be limited by using a series of rules. In a specific embodiment of the present invention, a counter is increased each time a server is selected to handle a connection request. For each service, connections coming in are tracked. Therefore, the source of connection-request packets need not be examined. Only the destination service is important. This saves significant time in the examination of the incoming requests. Each service may have its own set of rules to best handle the new traffic for its particular situation.



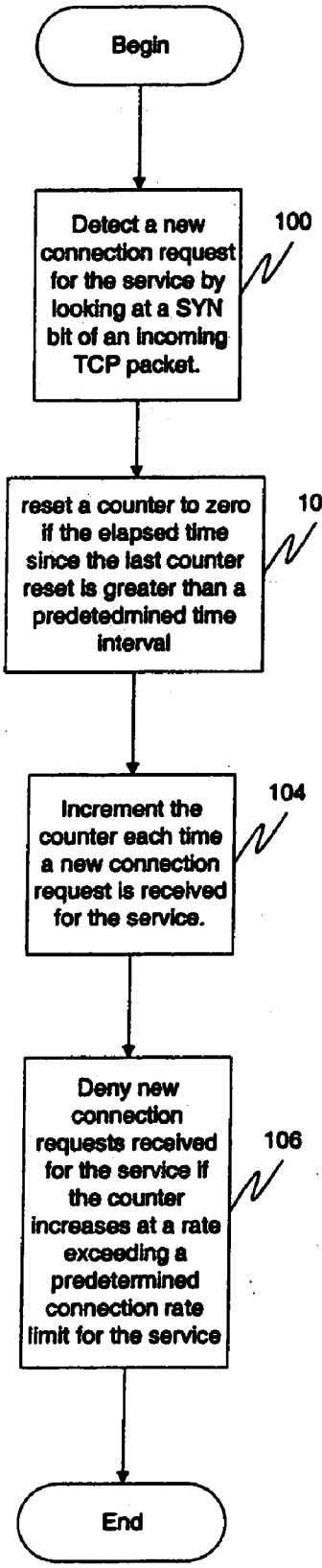

Begin
Detect a new connection request for the service by looking at a SYN bit of an incoming TCP packet.
100
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
102
Increment the counter each time a new connection request is received for the service.
104
Deny new connection requests received for the service if the counter increases at a rate exceeding a predetermined connection rate limit for the service
106
End

FIG. 1

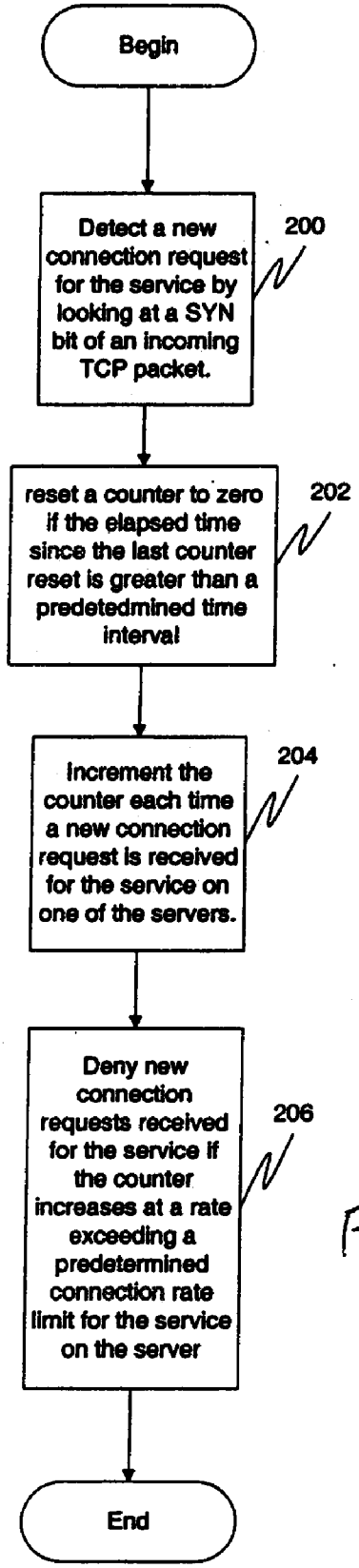

Begin
Detect a new connection request for the service by looking at a SYN bit of an incoming TCP packet.
200
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
202
Increment the counter each time a new connection request is received for the service on one of the servers.
204
Deny new connection requests received for the service if the counter increases at a rate exceeding a predetermined connection rate limit for the service on the server
206
End

FIG. 2

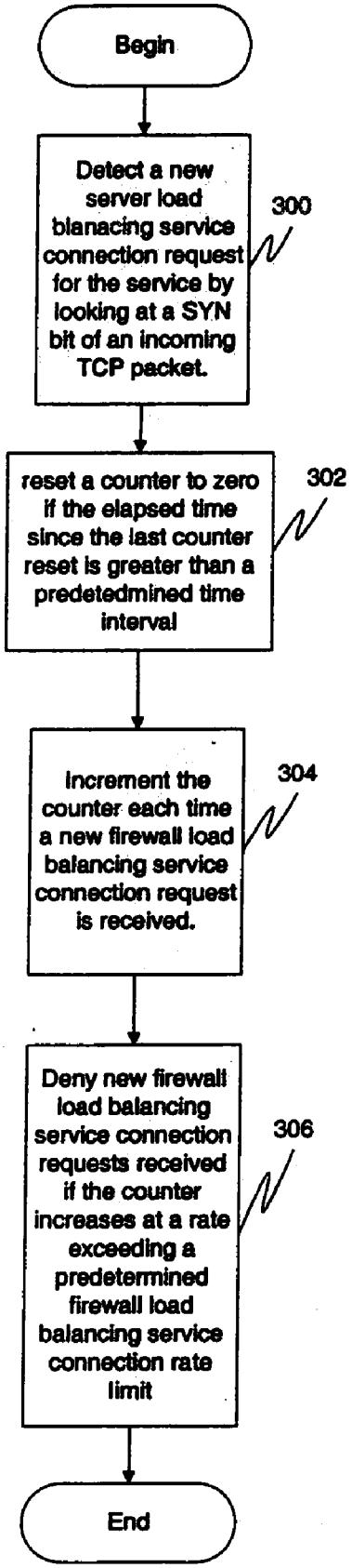
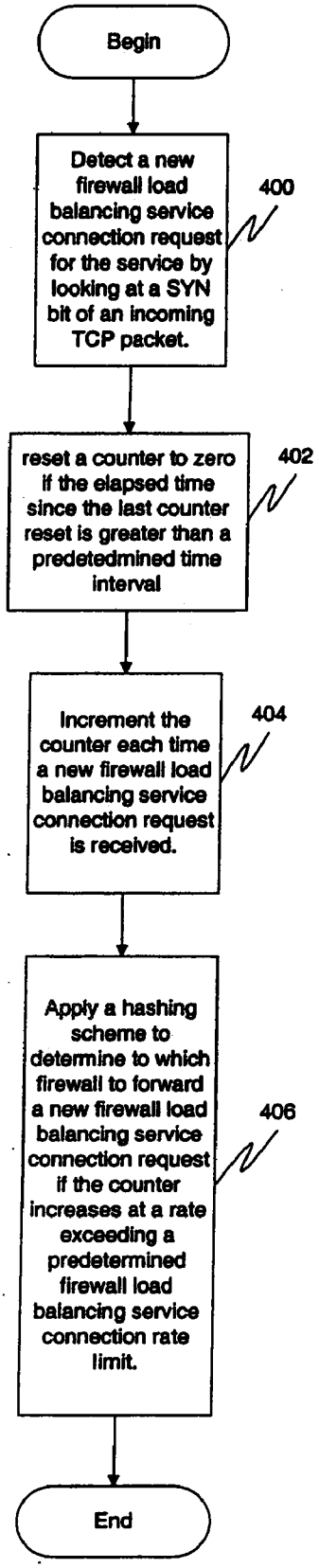
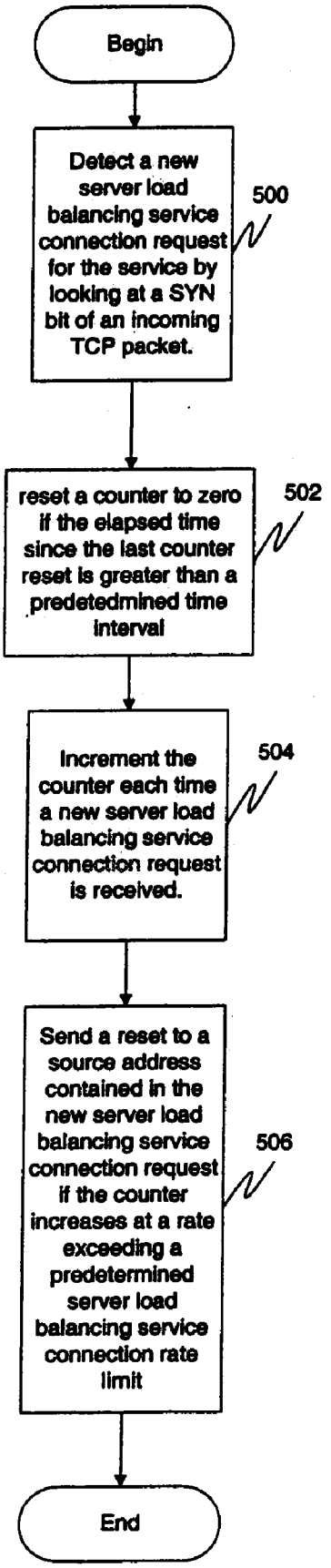

Begin
Detect a new server load blanacing service connection request for the service by looking at a SYN bit of an incoming TCP packet.
300
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
302
Increment the counter each time a new firewall load balancing service connection request is received.
304
Deny new firewall load balancing service connection requests received if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit
306
End

FIG. 3


Begin
Detect a new firewall load balancing service connection request for the service by looking at a SYN bit of an incoming TCP packet.
400
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
402
Increment the counter each time a new firewall load balancing service connection request is received.
404
Apply a hashing scheme to determine to which firewall to forward a new firewall load balancing service connection request if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit.
406
End

F16. 4


Begin
Detect a new server load balancing service connection request for the service by looking at a SYN bit of an incoming TCP packet.
500
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
502
Increment the counter each time a new server load balancing service connection request is received.
504
Send a reset to a source address contained in the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined server load balancing service connection rate limit
506
End

F16.5

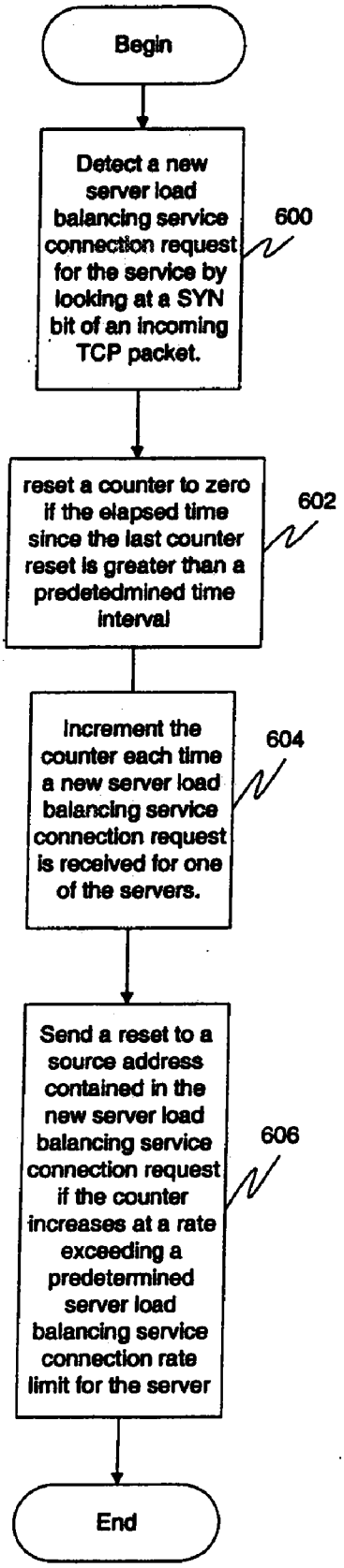

Begin
Detect a new server load balancing service connection request for the service by looking at a SYN bit of an incoming TCP packet.
600
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
602
Increment the counter each time a new server load balancing service connection request is received for one of the servers.
604
Send a reset to a source address contained in the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined server load balancing service connection rate limit for the server
606
End

FIG. 6

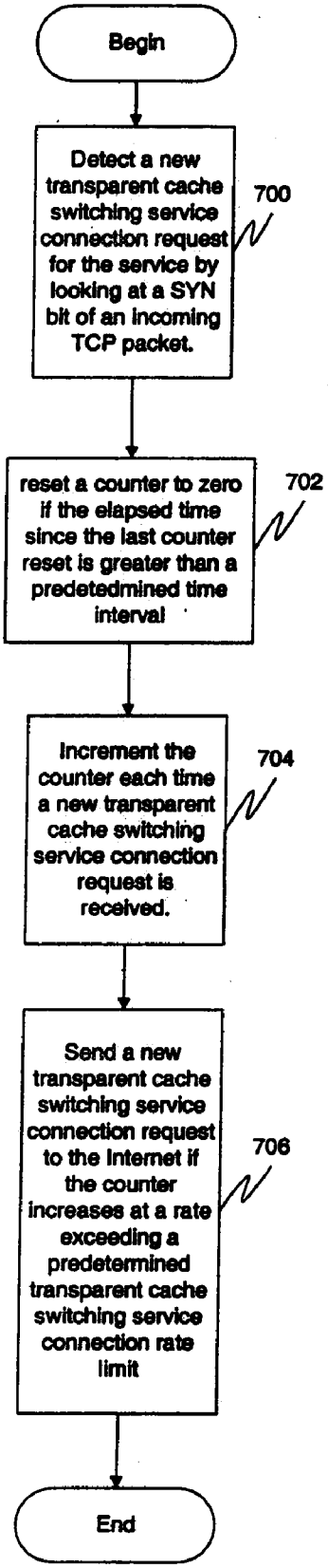
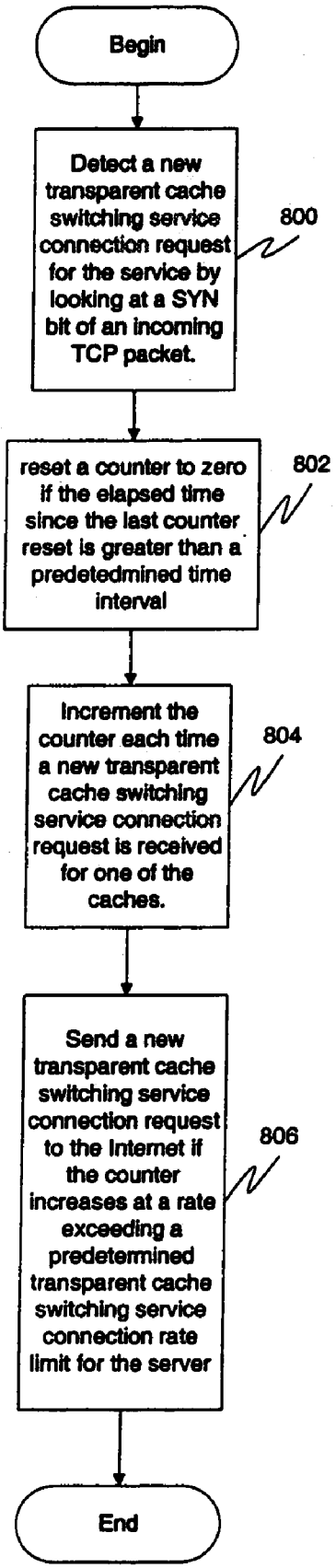
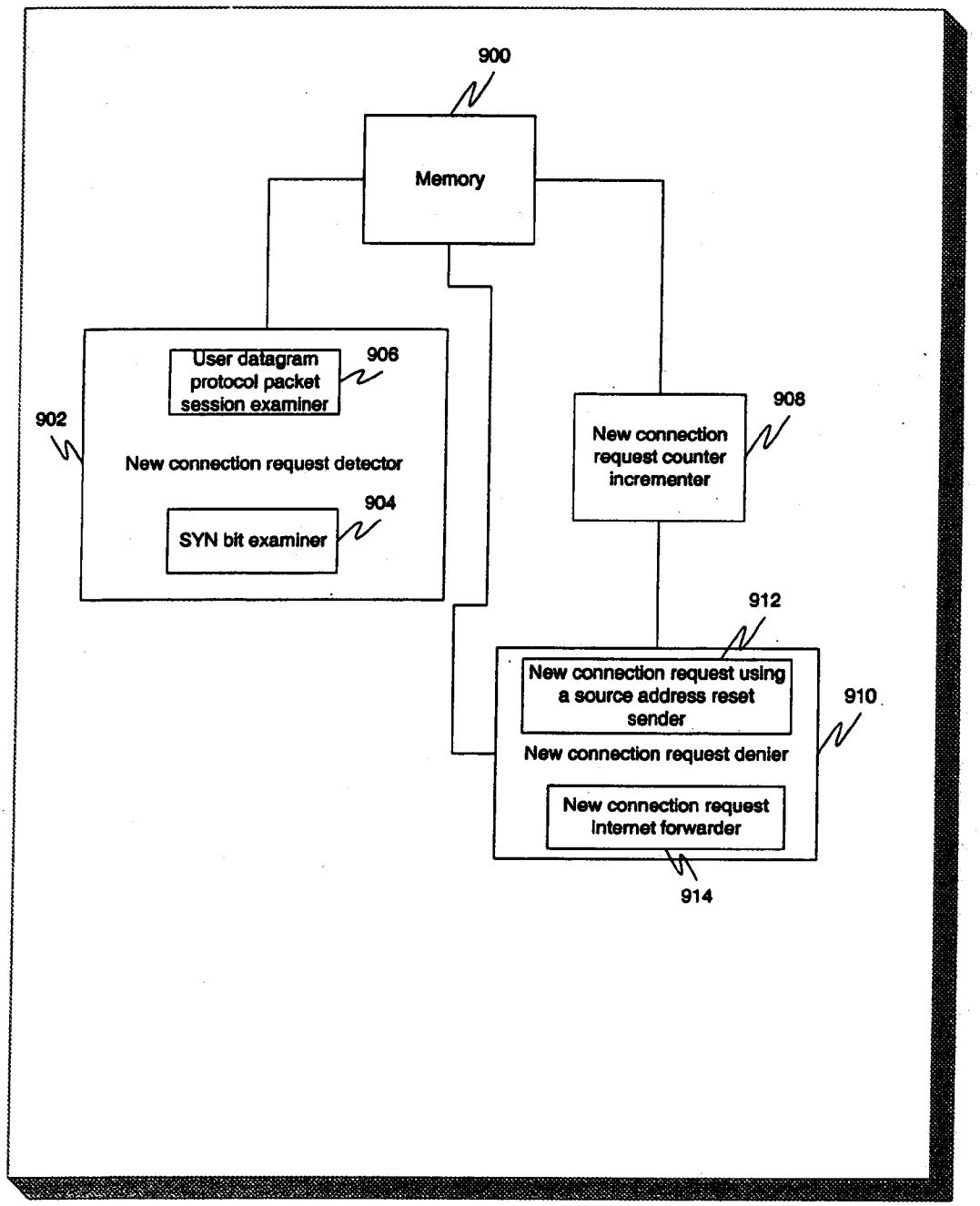

Begin
Detect a new transparent cache switching service connection request for the service by looking at a SYN bit of an incoming TCP packet.
700
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
702
Increment the counter each time a new transparent cache switching service connection request is received.
704
Send a new transparent cache switching service connection request to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit
706
End

Fig. 7


Begin
Detect a new transparent cache switching service connection request for the service by looking at a SYN bit of an incoming TCP packet.
800
reset a counter to zero if the elapsed time since the last counter reset is greater than a predetedmined time interval
802
Increment the counter each time a new transparent cache switching service connection request is received for one of the caches.
804
Send a new transparent cache switching service connection request to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit for the server
806
End

F16. 8


900
Memory
User datagram protocol packet session examiner
906
902
New connection request detector
904
SYN bit examiner
908
New connection request counter incrementer
912
New connection request using a source address reset sender
910
New connection request denier
New connection request Internet forwarder
914

F16. 9

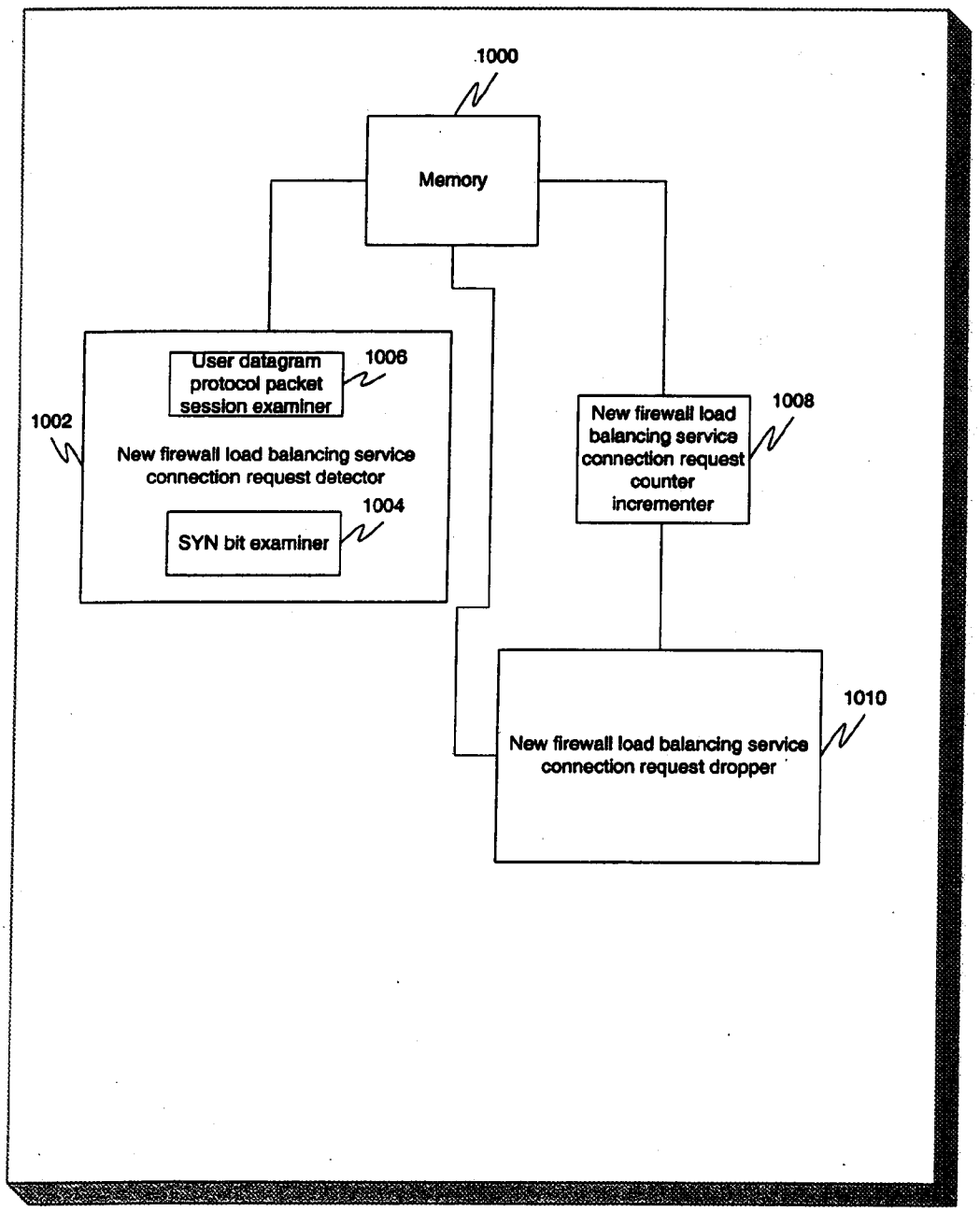

1000
Memory
1002
User datagram protocol packet session examiner
1006
New firewall load balancing service connection request detector
1004
SYN bit examiner
New firewall load balancing service connection request counter incrementer
1008
New firewall load balancing service connection request dropper
1010

FIG. 10

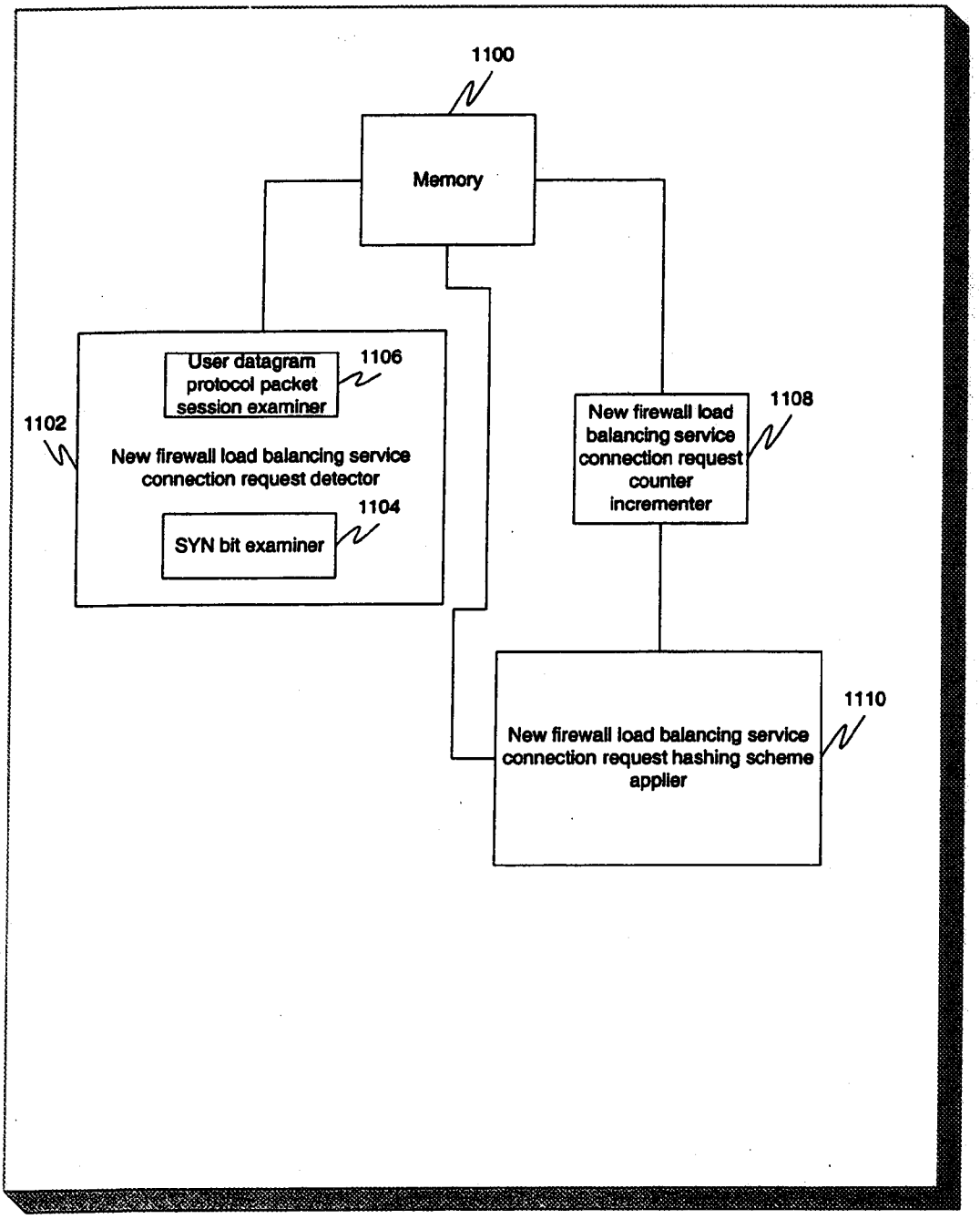

1100
Memory
1102
User datagram protocol packet session examiner
1106
New firewall load balancing service connection request detector
SYN bit examiner
1104
New firewall load balancing service connection request counter incrementer
1108
New firewall load balancing service connection request hashing scheme applier
1110

FIG. 11

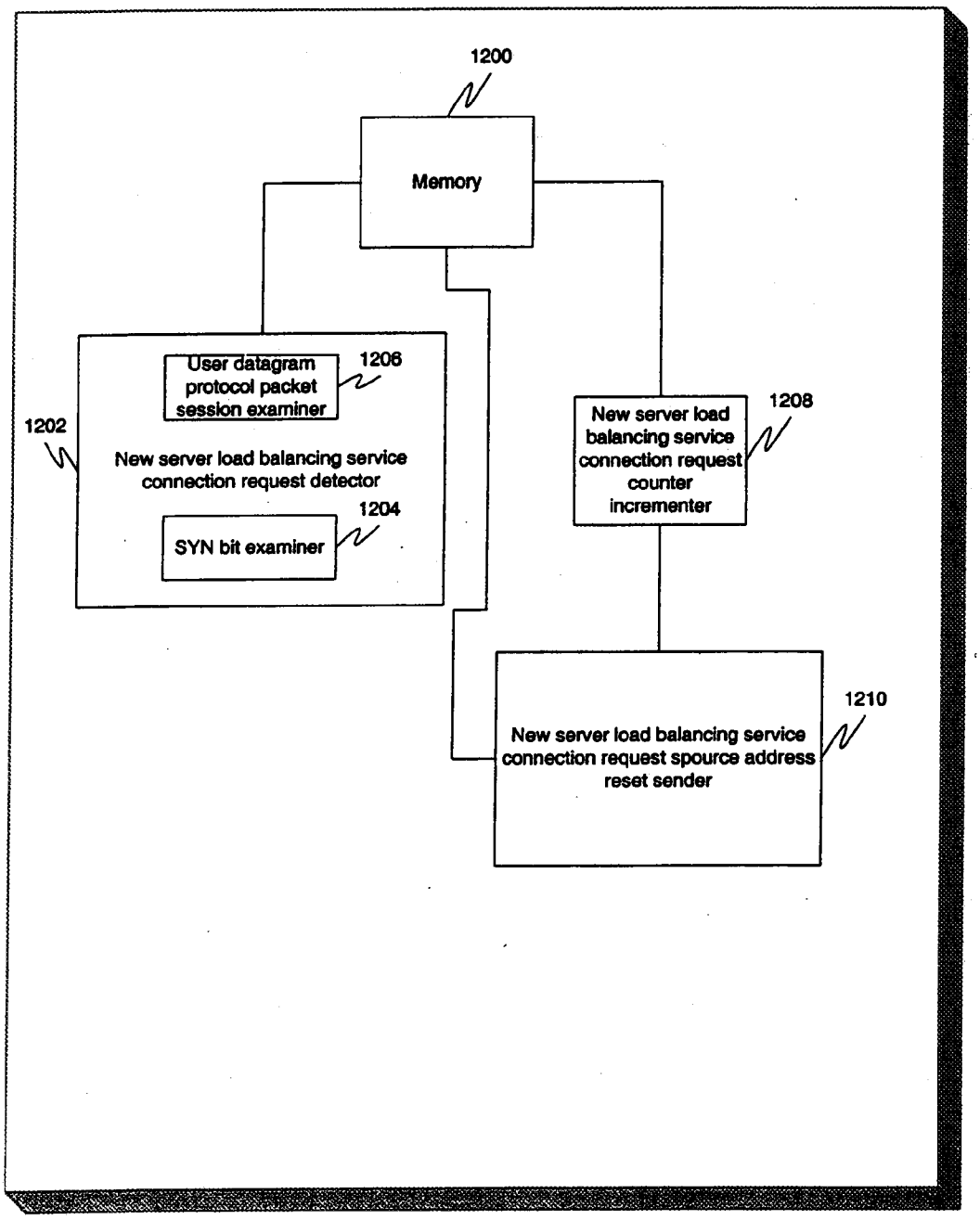
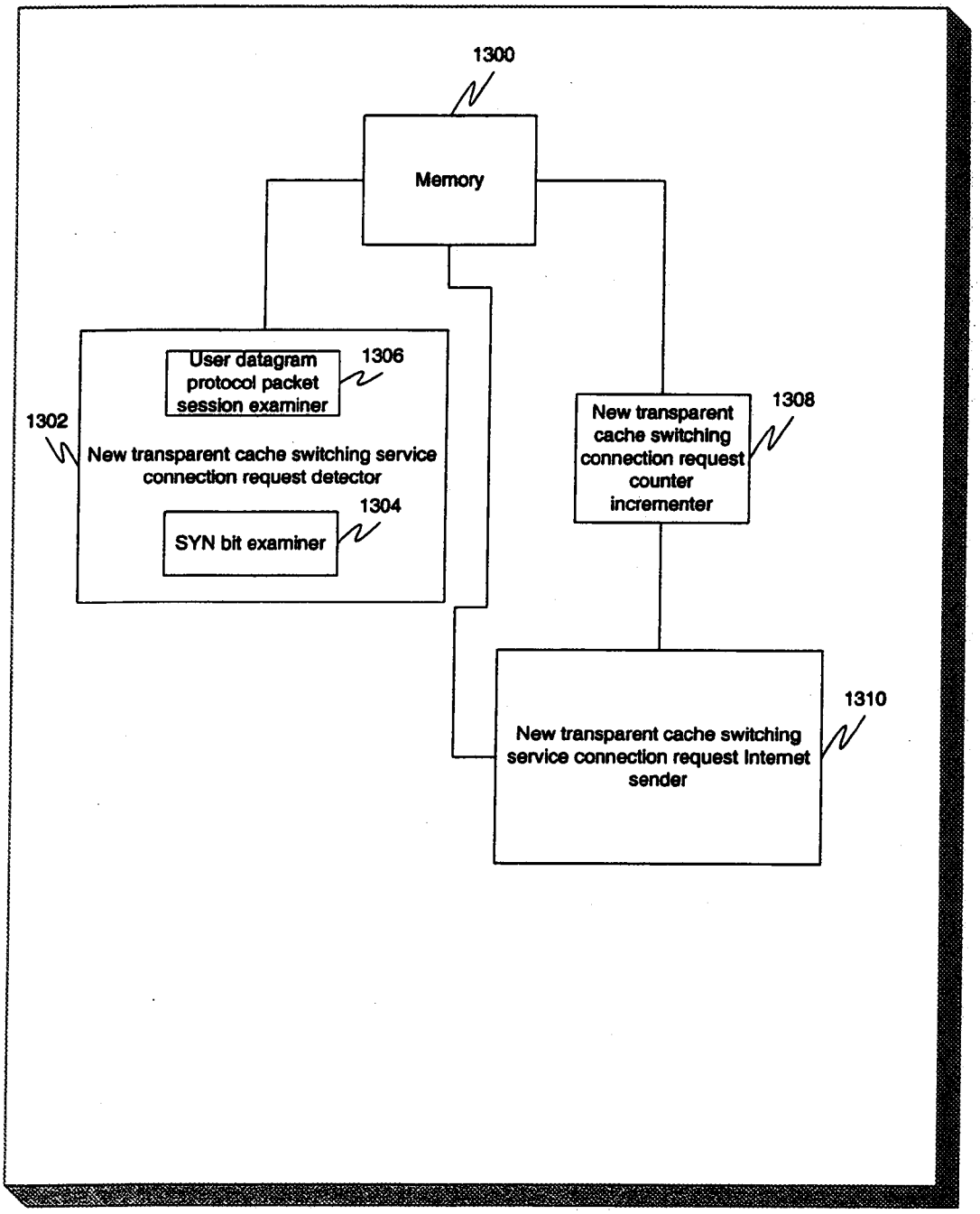

1200
Memory
User datagram protocol packet session examiner
1206
1202
New server load balancing service connection request detector
1204
SYN bit examiner
New server load balancing service connection request counter incrementer
1208
New server load balancing service connection request spource address reset sender
1210

FIG. 12


1300
Memory
1302
User datagram protocol packet session examiner
1306
New transparent cache switching service connection request detector
SYN bit examiner
1304
New transparent cache switching connection request counter incrementer
1308
New transparent cache switching service connection request Internet sender
1310

F16. 13

CONNECTION RATE LIMITING

CROSS-REFERENCE TO RELATED APPLICATION

[0001] The present application is a continuation of Ser. No. 10/139,073, filed May 3, 2002, by Ronald W. Szeto, David Chun Ying Cheung, and Rajkumar Jalan, entitled "CONNECTION RATE LIMITING" and is related to co-pending application Ser. No. 10/139,076, filed May 3, 2002, by Ronald W. Szeto, David Chun Ying Cheung, and Rajkumar Jalan, entitled "CONNECTION RATE LIMITING FOR SERVER LOAD BALANCING AND TRANSPARENT CACHE SWITCHING".

COPYRIGHT NOTICE

FIELD OF THE INVENTION

[0003] The present invention relates to the field of web switches. More particularly, the present invention relates to connection rate limiting to ensure proper functioning of components on a web switch.

BACKGROUND OF THE INVENTION

[0004] Web switches provide traffic management to computer networks. The traffic management extends to packets received both from an outside network, such as the Internet, and from an internal network. A web switch may provide a series of software components to better handle the traffic. These components may include server load balancing (SLB), transparent cache switching (TCS), and firewall load balancing (FWLB). Server load balancing allows IP-based services to be transparently balanced across multiple servers. This distributed design prevents servers from getting overloaded. Transparent cache switching allows for distributed cache servers, and likewise prevents the cache servers from getting overloaded. Firewall load balancing increases the network's overall firewall performance by distributing the Internet traffic load across multiple firewalls.

[0005] Even though these software components are designed to manage traffic, the components themselves may become overwhelmed when traffic is heavy. For example, a server running TCS may become so overloaded with connections that it fails to properly handle packets sent through the connections. Traditional techniques for handling such a situation involve limiting the packet rate. This involves monitoring the number of packets received in short intervals, and dropping or redirecting packets if the number exceeds a threshold value. Unfortunately, for traffic management components, the number of packets received is not a direct predictor of when the components will become overloaded. These traffic management components are more likely to become overloaded when new connections are being established too quickly, as opposed to when new packets are coming in over those connections.

[0006] What is needed is a solution to better handle increased traffic to traffic management components.

BRIEF DESCRIPTION OF THE INVENTION

[0007] Each service in a computer network may have a connection rate limit. The number of new connections per time period may be limited by using a series of rules. In a specific embodiment of the present invention, a counter is increased each time a server is selected to handle a connection request. For each service, connections coming in are tracked. Therefore, the source of connection-request packets need not be examined. Only the destination service is important. This saves significant time in the examination of the incoming requests. Each service may have its own set of rules to best handle the new traffic for its particular situation.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] The accompanying drawings, which are incorporated into and constitute a part of this specification, illustrate one or more embodiments of the present invention and, together with the detailed description, serve to explain the principles and implementations of the invention.

[0009] In the drawings:

[0010] FIG. **1** is a flow diagram illustrating a method for managing a traffic management service in a computer network in accordance with a specific embodiment of the present invention.

[0011] FIG. **2** is a flow diagram illustrating a method for managing a traffic management service distributed over multiple servers in a computer network in accordance with a specific embodiment of the present invention.

[0012] FIG. **3** is a flow diagram illustrating a method for managing a firewall load balancing service in a computer network in accordance with a specific embodiment of the present invention.

[0013] FIG. **4** is a flow diagram illustrating a method for managing a firewall load balancing service distributed over multiple firewalls in a computer network in accordance with a specific embodiment of the present invention.

[0014] FIG. **5** is a flow diagram illustrating a method for managing a server load balancing service in a computer network in accordance with a specific embodiment of the present invention.

[0015] FIG. **6** is a flow diagram illustrating a method for managing a server load balancing service distributed over multiple servers in a computer network in accordance with a specific embodiment of the present invention.

[0016] FIG. **7** is a flow diagram illustrating a method for managing a transparent cache switching service in a computer network in accordance with a specific embodiment of the present invention.

[0017] FIG. **8** is a flow diagram illustrating a method for managing a transparent cache switching service distributed over multiple caches in a computer network in accordance with a specific embodiment of the present invention.

[0018] FIG. **9** is a block diagram illustrating an apparatus for managing a traffic management service in a computer network in accordance with a specific embodiment of the present invention.

[0019] FIG. **10** is a block diagram illustrating an apparatus for managing a firewall load balancing service in a computer network in accordance with a specific embodiment of the present invention.

[0020] FIG. **11** is a block diagram illustrating an apparatus for managing a firewall load balancing service distributed over multiple firewalls in a computer network in accordance with a specific embodiment of the present invention.

[0021] FIG. **12** is a block diagram illustrating an apparatus for managing a server load balancing service in a computer network in accordance with a specific embodiment of the present invention.

DETAILED DESCRIPTION

[0022] Embodiments of the present invention are described herein in the context of a system of computers, servers, and software. Those of ordinary skill in the art will realize that the following detailed description of the present invention is illustrative only and is not intended to be in any way limiting. Other embodiments of the present invention will readily suggest themselves to such skilled persons having the benefit of this disclosure. Reference will now be made in detail to implementations of the present invention as illustrated in the accompanying drawings. The same reference indicators will be used throughout the drawings and the following detailed description to refer to the same or like parts.

[0023] In the interest of clarity, not all of the routine features of the implementations described herein are shown and described. It will, of course, be appreciated that in the development of any such actual implementation, numerous implementation-specific decisions must be made in order to achieve the developer's specific goals, such as compliance with application- and business-related constraints, and that these specific goals will vary from one implementation to another and from one developer to another. Moreover, it will be appreciated that such a development effort might be complex and time-consuming, but would nevertheless be a routine undertaking of engineering for those of ordinary skill in the art having the benefit of this disclosure.

[0024] In accordance with the present invention, the components, process steps, and/or data structures may be implemented using various types of operating systems, computing platforms, computer programs, and/or general purpose machines. In addition, those of ordinary skill in the art will recognize that devices of a less general purpose nature, such as hardwired devices, field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), or the like, may also be used without departing from the scope and spirit of the inventive concepts disclosed herein.

[0025] A traffic management component may be distributed over many different servers. Therefore, for purposes of this application a specific component type (such as TCS) may be referred to as a service. In accordance with a specific embodiment of the present invention, each service has a connection rate limit. The number of new connections per time period may be limited by using a series of rules. In a specific embodiment of the present invention, a counter is increased each time a server is selected to handle a connection request. For each service, connections coming in are tracked. Therefore, the source of connection-request packets need not be examined. Only the destination service is important. This saves significant time in the examination of the incoming requests. Each service may have its own set of rules to best handle the new traffic for its particular situation.

[0026] In accordance with a specific embodiment of the present invention, a new transmission control protocol (TCP) connection request may be detected by looking at the SYN bit of the incoming packet. If it is set to on, then the packet is a new connection request. In accordance with another specific embodiment of the present invention, a new user datagram protocol (UDP) connection request may be detected by looking for any packet that doesn't have a session.

[0027] In accordance with a specific embodiment of the present invention, connection rate limiting is applied to a server load balancing service. Upon receipt of a connection request that would exceed the maximum number of permitted connections per second, a reset is sent to the client (requesting party). Thus, instead of a user's request simply appearing to "hang" indefinitely, feedback is provided to the user to try again.

[0028] In accordance with a specific embodiment of the present invention, connection rate limiting is applied to transparent cache switching. Upon receipt of a connection request that would exceed the maximum number of permitted connections per second, the request is sent to the Internet. Thus, instead of not getting the service at all, the user still has a strong chance of getting the request served. This process is transparent to the user.

[0029] In accordance with a specific embodiment of the present invention, connection rate limiting is applied to firewall load balancing. Upon receipt of a connection request that would exceed the maximum number of permitted connections per second, the request is hashed to send it to a specific firewall. A hashing scheme may be applied to determine to which firewall to send the connection request. Different criteria may be applied in the hash table. For example, the hash table may be defined to direct the request to the firewall with the least connections. Alternatively, a round robin approach may be applied. In another embodiment, a weighted approach may be applied. The "scheme" may alternatively be a lack of a scheme, i.e., packets are simply dropped if the number of permitted connections per second is exceeded.

[0030] In accordance with another embodiment of the present invention, the connection rate limiting may be applied on a per server basis in addition to or instead of a per service basis. For example, the number of connections sent to a particular firewall may be limited, but other firewalls in the system may have no limiting or a different limiting scheme applied.

[0031] FIG. **1** is a flow diagram illustrating a method for managing a traffic management service in a computer network in accordance with a specific embodiment of the present invention. At **100**, a new connection request for the service is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new connection request for the service may be detected by looking for any user datagram protocol (UDP) packets without a session. At **102**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **104**, a counter is incremented each time a new connection request is received for the service. At **106**, new connection requests received for the service are denied if the counter increases at a rate exceeding a predetermined connection rate limit for the service. This denial may comprise sending a reset to a source address contained in a new connection request. Alternatively, it may comprise forwarding the new connection request to the Internet. It may also forward the new connection request in accordance with criteria in a hash table. The connection rate limit may be a number of connections per predetermined time interval.

[0032] FIG. **2** is a flow diagram illustrating a method for managing a traffic management service distributed over mul-

tiple servers in a computer network in accordance with a specific embodiment of the present invention. At **200**, a new connection request for the service is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new connection request for the service may be detected by looking for any user datagram protocol (UDP) packets without a session. At **202**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **204**, a counter is incremented each time a new connection request is received for the service on one of the servers. At **206**, new connection requests received for the service on the one server are denied if the counter increases at a rate exceeding a predetermined connection rate limit for the service on that server. This denying may comprise sending a reset to a source address contained in a new connection request. Alternatively, it may comprise forwarding the new connection request to the Internet. It may also forward the new connection request in accordance with criteria in a hash table. The connection rate limit may be a number of connections per predetermined time interval.

[0033] FIG. **3** is a flow diagram illustrating a method for managing a firewall load balancing service in a computer network in accordance with a specific embodiment of the present invention. At **300**, a new firewall load balancing service connection request is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new firewall load balancing service connection request may be detected by looking for any user datagram protocol (UDP) packets without a session. At **302**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **304**, a counter is incremented each time a new firewall load balancing service connection request is received. At **306**, new firewall load balancing service connection requests are dropped if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit. The connection rate limit may be a number of connections per predetermined time interval.

[0034] FIG. **4** is a flow diagram illustrating a method for managing a firewall load balancing service distributed over multiple firewalls in a computer network in accordance with a specific embodiment of the present invention. At **400**, a new firewall load balancing service connection request for the service is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new firewall load balancing service connection request for the service may be detected by looking for any user datagram protocol (UDP) packets without a session. At **402**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **404**, a counter is incremented each time a new firewall load balancing service connection request is received. At **406**, a hashing scheme is applied to determine to which firewall to forward a new firewall load balancing service connection request if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit. The hashing scheme may be one of several different possibilities. It may comprise directing a new firewall load balancing service connection request to the firewall with the least connections. It may comprise directing a new firewall load balancing service connection request to a firewall according to a round robin approach. It may comprise directing a new firewall load balancing service connection request to a firewall according to a weighted approach. The connection rate limit may be a number of connections per predetermined time interval.

[0035] FIG. **5** is a flow diagram illustrating a method for managing a server load balancing service in a computer network in accordance with a specific embodiment of the present invention. At **500**, a new server load balancing service connection request is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new server load balancing connection request may be detected by looking for any user datagram protocol (UDP) packets without a session. At **502**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **504**, a counter is incremented each time a new server load balancing service connection request is received. At **506**, a reset is sent to a source address contained in the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined server load balancing service connection rate limit. The connection rate limit may be a number of connections per predetermined time interval.

[0036] FIG. **6** is a flow diagram illustrating a method for managing a server load balancing service distributed over multiple servers in a computer network in accordance with a specific embodiment of the present invention. At **600**, a new server load balancing service connection request for the server is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new server load balancing connection request for the server may be detected by looking for any user datagram protocol (UDP) packets without a session. At **602**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **604**, a counter is incremented each time a new server load balancing service connection request for the server is received. At **606**, a reset is sent to a source address contained in the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined server load balancing service connection rate limit for the server. The connection rate limit may be a number of connections per predetermined time interval.

[0037] FIG. **7** is a flow diagram illustrating a method for managing a transparent cache switching service in a computer network in accordance with a specific embodiment of the present invention. At **700**, a new transparent cache switching service connection request is detected by looking at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a new transparent cache switching service connection request may be detected by looking for any user datagram protocol (UDP) packets without a session. At **702**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **704**, a counter is incremented each time a new transparent cache switching service connection request is received. At **706**, the new transparent cache switching service connection request is sent to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit. The connection rate limit may be a number of connections per predetermined time interval.

[0038] FIG. **8** is a flow diagram illustrating a method for managing a transparent cache switching service distributed over multiple caches in a computer network in accordance with a specific embodiment of the present invention. At **800**, a new transparent cache switching service connection request for one of the caches is detected by looking at a SYN bit of an

incoming transmission control protocol (TCP) packet. Alternatively, a new transparent cache switching service connection request for one of the caches may be detected by looking for any user datagram protocol (UDP) packets without a session. At **802**, a counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. At **804**, a counter is incremented each time a new transparent cache switching service connection request for the cache is received. At **806**, the new transparent cache switching service connection request is sent to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit for the cache. The connection rate limit may be a number of connections per predetermined time interval.

[0039] FIG. **9** is a block diagram illustrating an apparatus for managing a traffic management service in a computer network in accordance with a specific embodiment of the present invention. A memory **900** may be used to store a counter. A new connection request detector **902** may detect a new connection request for the service. A SYN bit examiner **904** may be used for this purpose to look at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a user datagram protocol packet session examiner **906** may detect a new connection request for the service by looking for any user datagram protocol (UDP) packets without a session. A counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. A new connection request counter incrementer **908** coupled to the memory **900** and to the new connection request detector **902** increments the counter each time a new connection request is received for the service. If the service is distributed over multiple servers and the request is for one of the servers, the new connection request counter incrementer **908** may increment a counter each time a new connection request is received for the service on the one server. A new connection request denier **910** coupled to the new connection request counter incrementer **908** and to the memory **900** denies new connection requests received for the service if the counter increases at a rate exceeding a predetermined connection rate limit for the service. If the service is distributed over multiple servers and the request is for one of the servers, the new connection request denier **910** may deny new connection requests received for the service on the server if the counter increases at a rate exceeding a predetermined connection rate limit for the service on the server. This denying may comprise sending a reset to a source address contained in a new connection request using a source address reset sender **912**. Alternatively, it may comprise forwarding the new connection request to the Internet using a new connection request Internet forwarder **914**. It may also forward the new connection request as per a hash table using a new connection request hash table forwarder **916**. The connection rate limit may be a number of connections per predetermined time interval.

[0040] FIG. **10** is a block diagram illustrating an apparatus for managing a firewall load balancing service in a computer network in accordance with a specific embodiment of the present invention. A memory **1000** may be used to store a counter. A new firewall load balancing service connection request detector **1002** may detect a new firewall load balancing service connection request. A SYN bit examiner **1004** may be used for this purpose to look at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a user datagram protocol packet session examiner **1006** may detect a new firewall load balancing connection request by looking for any user datagram protocol (UDP) packets without a session. A counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. A new firewall load balancing service connection request counter incrementer **1008** coupled to the memory **1000** and to the new firewall load balancing service connection request detector **1002** increments the counter each time a new firewall load balancing service connection request is received. A new firewall load balancing service connection request dropper **1010** coupled to the new firewall load balancing service connection request counter incrementer **1008** and to the memory **1000** drops new firewall load balancing service connection requests if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit. The connection rate limit may be a number of connections per predetermined time interval.

[0041] FIG. **11** is a block diagram illustrating an apparatus for managing a firewall load balancing service distributed over multiple firewalls in a computer network in accordance with a specific embodiment of the present invention. A memory **1100** may be used to store a counter. A new firewall load balancing service connection request detector **1102** may detect a new firewall load balancing service connection request. A SYN bit examiner **1104** may be used for this purpose to look at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a user datagram protocol packet session examiner **1106** may detect a new firewall load balancing service connection request by looking for any user datagram protocol (UDP) packets without a session. A counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. A new firewall load balancing service connection request counter incrementer **1108** coupled to the memory **1100** and to the new firewall load balancing service connection request detector **1102** increments the counter each time a new firewall load balancing service connection request is received. A new firewall load balancing service connection request hashing scheme applier **1110** coupled to the new firewall load balancing service connection request counter incrementer **1108** and to the memory **1100** applies a hashing scheme to determine to which firewall to forward a new firewall load balancing service connection request if the counter increases at a rate exceeding a predetermined firewall load balancing service connection rate limit. The hashing scheme may be one of several different possibilities. It may comprise directing a new firewall load balancing service connection request to the firewall with the least connections. It may comprise directing a new firewall load balancing service connection request to a firewall according to a round robin approach. It may comprise directing a new firewall load balancing service connection request to a firewall according to a weighted approach. The connection rate limit may be a number of connections per predetermined time interval.

[0042] FIG. **12** is a block diagram illustrating an apparatus for managing a server load balancing service in a computer network in accordance with a specific embodiment of the present invention. A memory **1200** may be used to store a counter. A new server load balancing service connection request detector **1202** may detect a new server load balancing service connection request. A SYN bit examiner **1204** may be used for this purpose to look at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a user datagram protocol packet session examiner **1206** may

detect a new server load balancing service connection request for the service by looking for any user datagram protocol (UDP) packets without a session. A counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. A new server load balancing service connection request counter incrementer **1208** coupled to the memory **1200** and to the new server load balancing service connection request detector **1202** increments a counter each time a new server load balancing connection request is received. If the service is distributed over multiple servers and the request is for one of the servers, the new server load balancing service connection request counter incrementer **1208** may increment the counter each time a new server load balancing service connection request is received for the server. A new server load balancing service connection request source address reset sender **1210** coupled to the new server load balancing service connection request counter incrementer **1208** and to the memory **1200** sends a reset to the source address of the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined server load balancing service connection rate limit. If the service is distributed over multiple servers and the request is for one of the servers, the new server load balancing service connection request source address reset sender **1210** may send a reset to the source address of the new server load balancing service connection request if the counter increases at a rate exceeding a predetermined connection rate limit for the service on the server. The connection rate limit may be a number of connections per predetermined time interval.

[0043] FIG. **13** is a block diagram illustrating an apparatus for managing a transparent cache switching service in a computer network in accordance with a specific embodiment of the present invention. A memory **1300** may be used to store a counter. A new transparent cache switching service connection request detector **1302** may detect a new transparent cache switching service connection request. A SYN bit examiner **1304** may be used for this purpose to look at a SYN bit of an incoming transmission control protocol (TCP) packet. Alternatively, a user datagram protocol packet session examiner **1306** may detect a new transparent cache switching service connection request for the service by looking for any user datagram protocol (UDP) packets without a session. A counter is reset to zero if the elapsed time since the last counter reset is greater than a predetermined time interval. A new transparent cache switching service connection request counter incrementer **1308** coupled to the memory **1300** and to the new transparent cache switching service connection request detector **1302** increments the counter each time a new transparent cache switching connection request is received. If the service is distributed over multiple caches and the request is for one of the caches, the new transparent cache switching service connection request counter incrementer **1308** may increment a counter each time a new transparent cache switching service connection request is received for the cache. A new transparent cache switching service connection request Internet sender **1310** coupled to the new transparent cache switching service connection request counter incrementer **1308** and to the memory **1300** sends the new transparent cache switching service connection request to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit. If the service is distributed over multiple caches and the request is for one of the caches, the new transparent cache switching service connection request Internet sender **1310** may send the new transparent cache switching service connection request to the Internet if the counter increases at a rate exceeding a predetermined transparent cache switching service connection rate limit for the cache. The connection rate limit may be a number of connections per predetermined time interval.

[0044] While embodiments and applications of this invention have been shown and described, it would be apparent to those skilled in the art having the benefit of this disclosure that many more modifications than mentioned above are possible without departing from the inventive concepts herein. The invention, therefore, is not to be restricted except in the spirit of the appended claims.

What is claimed is:

1. A computer implemented method for firewall load balancing connection rate limiting, the method comprising:

incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

directing, by the computing platform, the new destination firewall load balancing service connection request to a particular one of the plurality of firewalls of the destination firewall load balancing service, if the counter has not increased at a rate exceeding the predetermined connection rate limit.

2. The method of claim **1**, further comprising resetting the counter to zero if the elapsed time since the last counter reset is greater than a predetermined time interval.

3. The method of claim **2** wherein the predetermined connection rate limit is a number of transactions per predetermined time interval.

4. The method of claim **1**, further comprising detecting a new firewall load balancing service connection request by looking at a SYN bit of an incoming transmission control protocol (TCP) packet.

5. The method of claim **1** wherein the incrementing is based at least in part on the identification of the destination firewall load balancing service.

6. A computer implemented method for firewall load balancing connection rate limiting, the method comprising:

incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

dropping, by the computing platform, the new connection requests for the firewall load balancing service if the counter increases at a rate exceeding a predetermined connection rate limit for the firewall load balancing service.

7. The method of claim 6, further comprising resetting the counter to zero if the elapsed time since the last counter reset is greater than a predetermined time interval.

8. The method of claim 7 wherein the predetermined connection rate limit is a number of transactions per predetermined time interval.

9. The method of claim 6, further comprising detecting a new firewall load balancing service connection request by looking at a SYN bit of an incoming transmission control protocol (TCP) packet.

10. The method of claim 6 wherein the incrementing is based at least in part on the identification of the destination firewall load balancing service.

11. An apparatus for firewall load balancing connection rate limiting, the apparatus comprising:

a memory; and

a computing platform of a network switch, the computing platform configured to:

increment a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

direct the new destination firewall load balancing service connection request to a particular one of the plurality of firewalls of the destination firewall load balancing service, if the counter has not increased at a rate exceeding the predetermined connection rate limit.

12. The apparatus of claim 11 wherein the computing platform is further configured to reset the counter to zero if the elapsed time since the last counter reset is greater than a predetermined time interval.

13. The apparatus of claim 12 wherein the predetermined connection rate limit is a number of transactions per predetermined time interval.

14. The apparatus of claim 11 wherein the computing platform is further configured to detect a new firewall load balancing service connection request by looking at a SYN bit of an incoming transmission control protocol (TCP) packet.

15. The method of claim 11 wherein the incrementing is based at least in part on the identification of the destination firewall load balancing service.

16. An apparatus for firewall load balancing connection rate limiting, the apparatus comprising:

a memory; and

a computing platform of a network switch, the computing platform configured to:

increment a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

drop the new connection requests for the firewall load balancing service if the counter increases at a rate exceeding a predetermined connection rate limit for the firewall load balancing service.

17. The apparatus of claim 16 wherein the computing platform is further configured to reset the counter to zero if the elapsed time since the last counter reset is greater than a predetermined time interval.

18. The apparatus of claim 17 wherein the predetermined connection rate limit is a number of transactions per predetermined time interval.

19. The apparatus of claim 16 wherein the computing platform is further configured to detect a new firewall load balancing service connection request by looking at a SYN bit of an incoming transmission control protocol (TCP) packet.

20. The apparatus of claim 16 wherein the incrementing is based at least in part on the identification of the destination firewall load balancing service.

21. An apparatus for firewall load balancing connection rate limiting, the apparatus comprising:

a memory;

means for incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

means for directing, by the computing platform, the new destination firewall load balancing service connection request to a particular one of the plurality of firewalls of the destination firewall load balancing service, if the counter has not increased at a rate exceeding the predetermined connection rate limit.

22. An apparatus for firewall load balancing connection rate limiting, the apparatus comprising:

a memory;

means for incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

means for dropping, by the computing platform, the new connection requests for the firewall load balancing service if the counter increases at a rate exceeding a predetermined connection rate limit for the firewall load balancing service.

23. A program storage device readable by a machine, embodying a program of instructions executable by the machine to perform a method for firewall load balancing connection rate limiting, the method comprising:

incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received

and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

directing, by the computing platform, the new destination firewall load balancing service connection request to a particular one of the plurality of firewalls of the destination firewall load balancing service, if the counter has not increased at a rate exceeding the predetermined connection rate limit.

24. A program storage device readable by a machine, embodying a program of instructions executable by the machine to perform a method for firewall load balancing connection rate limiting, the method comprising:

incrementing, by a computing platform of a network switch, a counter each time a new connection request for a destination firewall load balancing service is received and a firewall is selected from a plurality of firewalls of the firewall load balancing service to handle the new connection request, the request identifying the destination firewall load balancing service, the counter indicating a total number of times the destination firewall load balancing service has been requested within a predetermined time interval by examining a destination address of the request; and

dropping, by the computing platform, the new connection requests for the firewall load balancing service if the counter increases at a rate exceeding a predetermined connection rate limit for the firewall load balancing service.

* * * * *