US009270686B1

(12) **United States Patent**
**Canion et al.**

(10) **Patent No.: US 9,270,686 B1**
(45) **Date of Patent: Feb. 23, 2016**

(54) **ZERO COPY PACKET BUFFERING USING SHADOW SENDS**

(75) Inventors: **Rodney S. Canion**, West Lake Hills, TX (US); **Alexander I. Tomlinson**, Austin, TX (US)

(73) Assignee: **Hewlett Packard Enterprise Development LP**, Houston, TX (US)

( * ) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 2337 days.

(21) Appl. No.: **12/183,642**

(22) Filed: **Jul. 31, 2008**

**Related U.S. Application Data**

(60) Provisional application No. 60/953,802, filed on Aug. 3, 2007.

(51) **Int. Cl.**
***H04L 29/06*** (2006.01)

(52) **U.S. Cl.**
CPC ..... ***H04L 63/1408*** (2013.01); *H04L 63/0263* (2013.01)

(58) **Field of Classification Search**
None
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

| | | | | |
|---|---|---|---|---|
| 7,203,960 | B1 * | 4/2007 | Painter | 726/22 |
| 7,844,700 | B2 * | 11/2010 | Marinescu et al. | 709/224 |
| 7,992,206 | B1 * | 8/2011 | Painter et al. | 726/23 |
| 8,042,184 | B1 * | 10/2011 | Batenin | 726/24 |
| 2003/0023786 | A1 * | 1/2003 | Craddock et al. | 710/52 |
| 2006/0195904 | A1 * | 8/2006 | Williams | 726/24 |

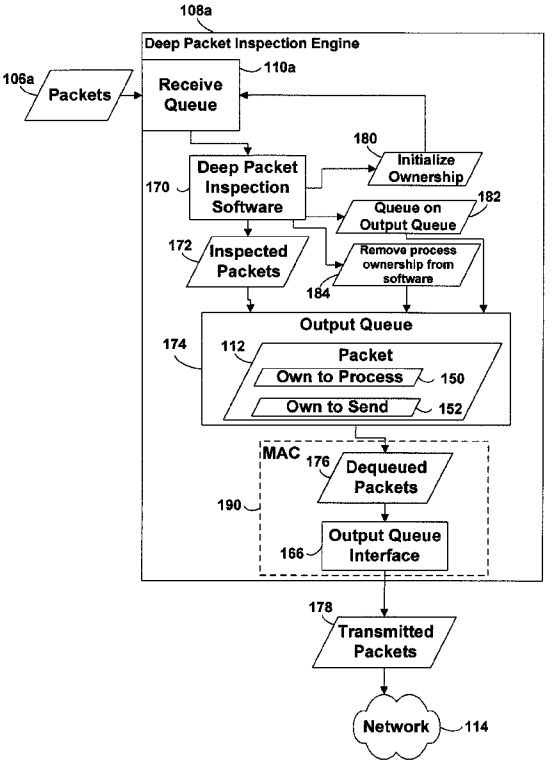
* cited by examiner

*Primary Examiner* — Kaveh Abrishamkar
(74) *Attorney, Agent, or Firm* — Hewlett Packard Enterprise Patent Department

(57) **ABSTRACT**

Packets in an intrusion prevention system are inspected by a deep packet inspection engine. A packet may be queued for transmission onto an output queue and transmitted over a network while deep packet inspection is still being performed on the packet. Such simultaneous inspection processing and transmission may be implemented using two ownership bits for the packet, one to indicate "ownership to process" and one to indicate "ownership to send," instead of the single ownership bit that is used in conventional systems. Furthermore, the packet may be inspected, queued onto the output queue, and transmitted without making a copy of the packet within the deep packet inspection engine. These techniques enable the inspection latency, and therefore the overall transmission latency, of packets to decrease, thereby improving the overall performance of the intrusion prevent system.

**10 Claims, 5 Drawing Sheets**



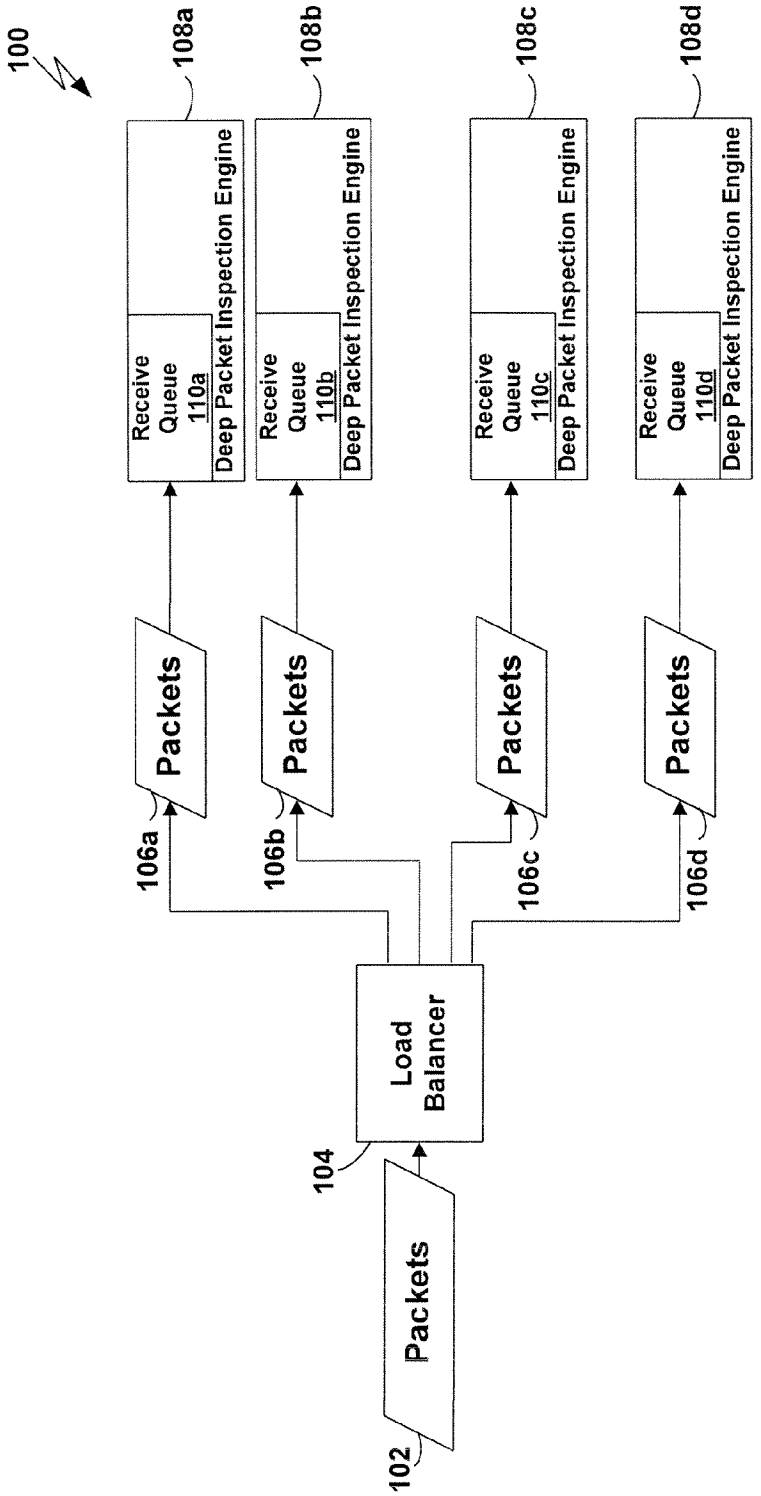

100
108a
108b
108c
108d
Receive Queue 110a
Deep Packet Inspection Engine
Receive Queue 110b
Deep Packet Inspection Engine
Receive Queue 110c
Deep Packet Inspection Engine
Receive Queue 110d
Deep Packet Inspection Engine
Packets
106a
Packets
106b
Packets
106c
Packets
106d
Load Balancer
104
Packets
102

FIG. 1A

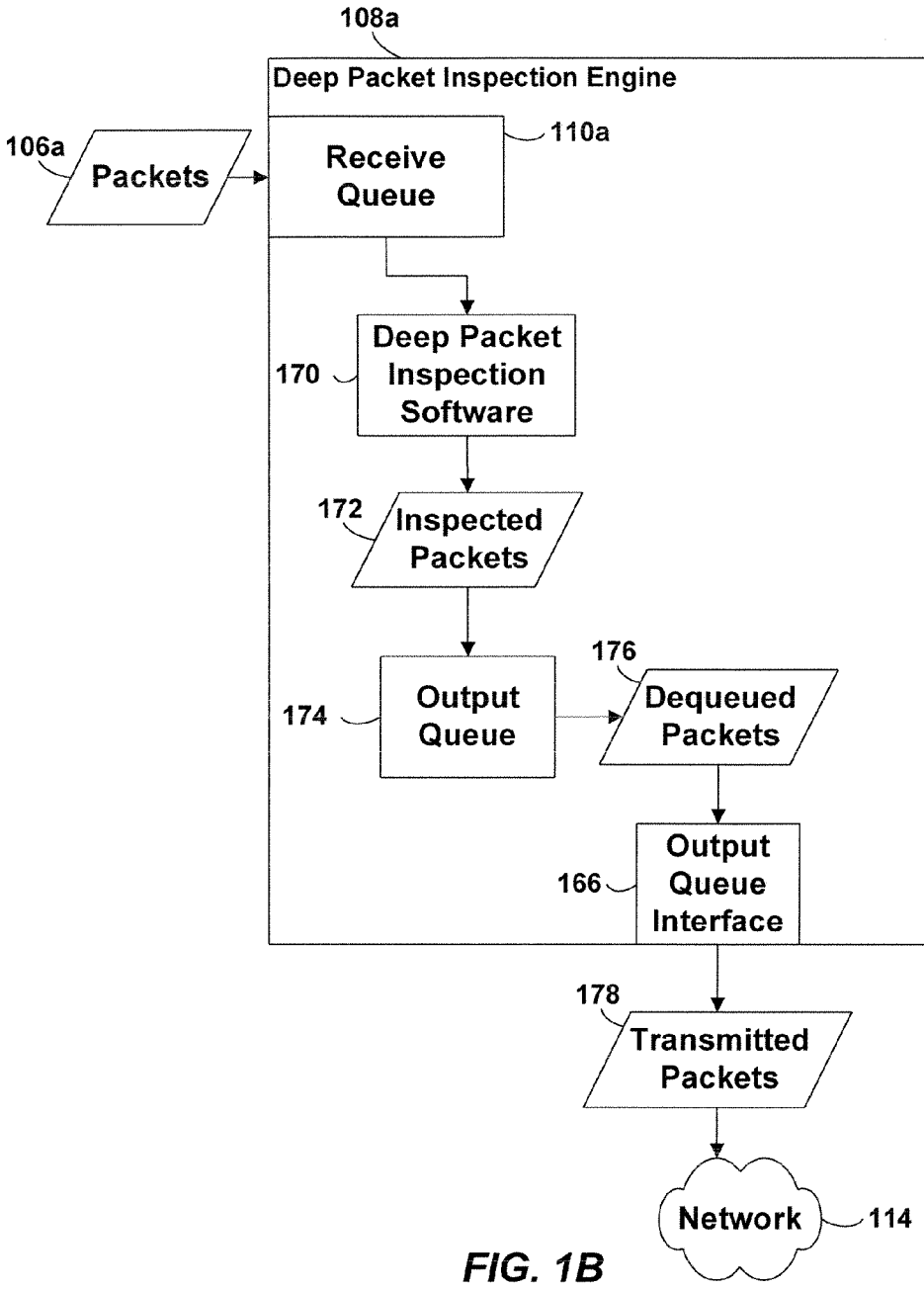

108a
Deep Packet Inspection Engine
106a
Packets
Receive Queue
110a
Deep Packet Inspection Software
170
172
Inspected Packets
Output Queue
174
176
Dequeued Packets
Output Queue Interface
166
178
Transmitted Packets
Network
114

FIG. 1B

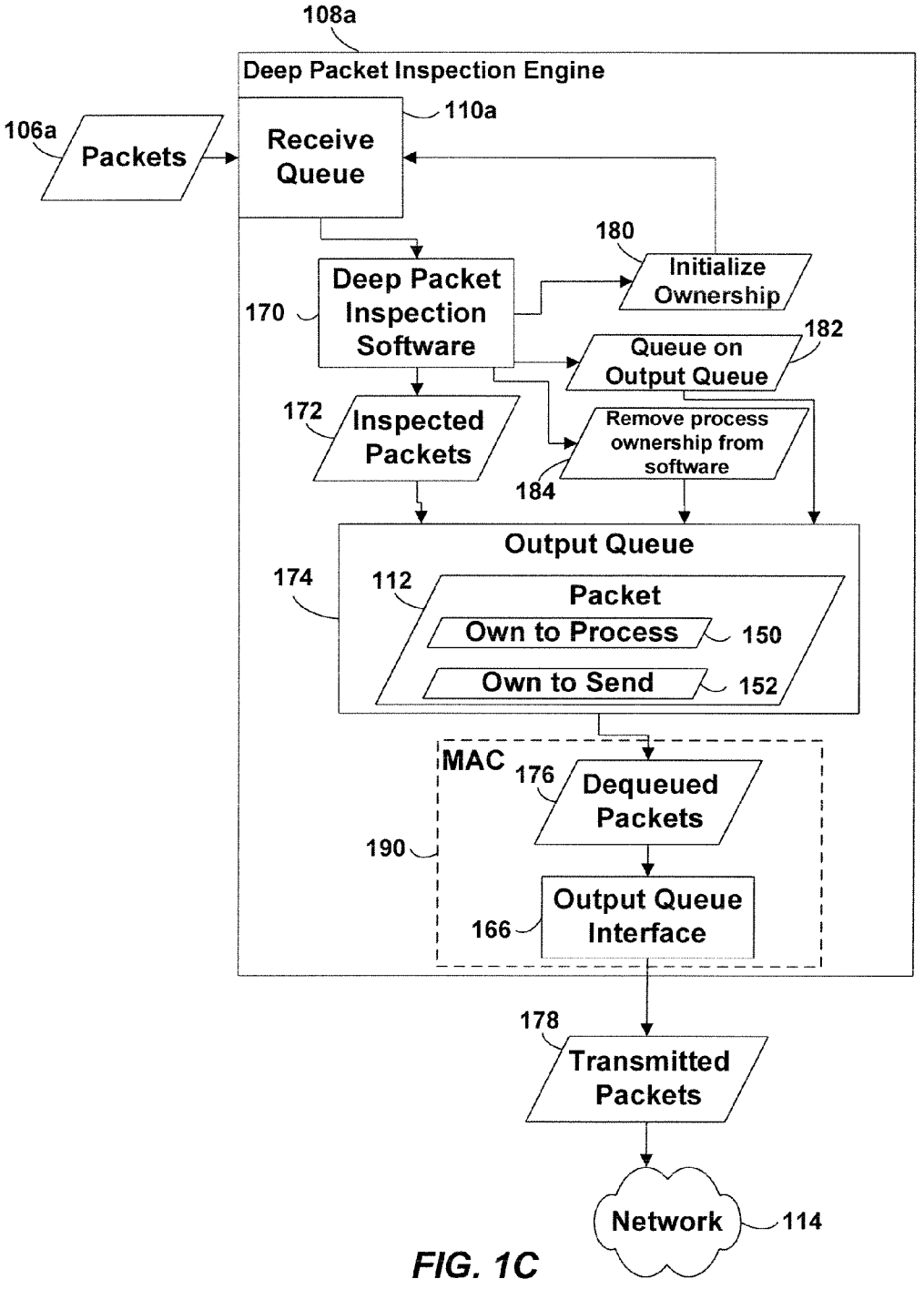

108a
Deep Packet Inspection Engine
106a
Packets
Receive Queue
110a
180
Initialize Ownership
Deep Packet Inspection Software
170
182
Queue on Output Queue
172
Inspected Packets
Remove process ownership from software
184
Output Queue
174
112
Packet
Own to Process
150
Own to Send
152
MAC
176
Dequeued Packets
190
Output Queue Interface
166
178
Transmitted Packets
Network
114

FIG. 1C

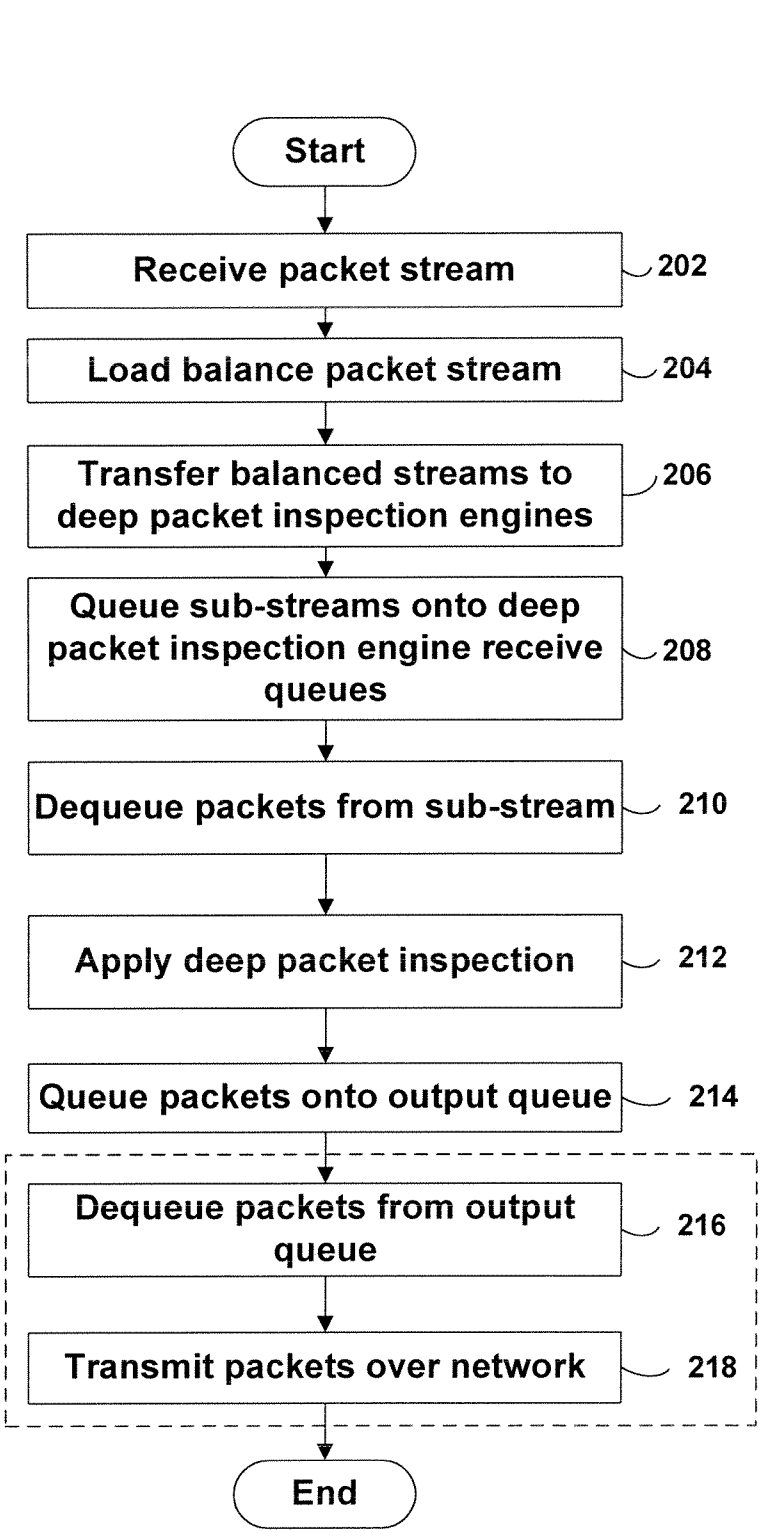

200
Start
Receive packet stream
202
Load balance packet stream
204
Transfer balanced streams to deep packet inspection engines
206
Queue sub-streams onto deep packet inspection engine receive queues
208
Dequeue packets from sub-stream
210
Apply deep packet inspection
212
Queue packets onto output queue
214
Dequeue packets from output queue
216
Transmit packets over network
218
End

*FIG. 2A*

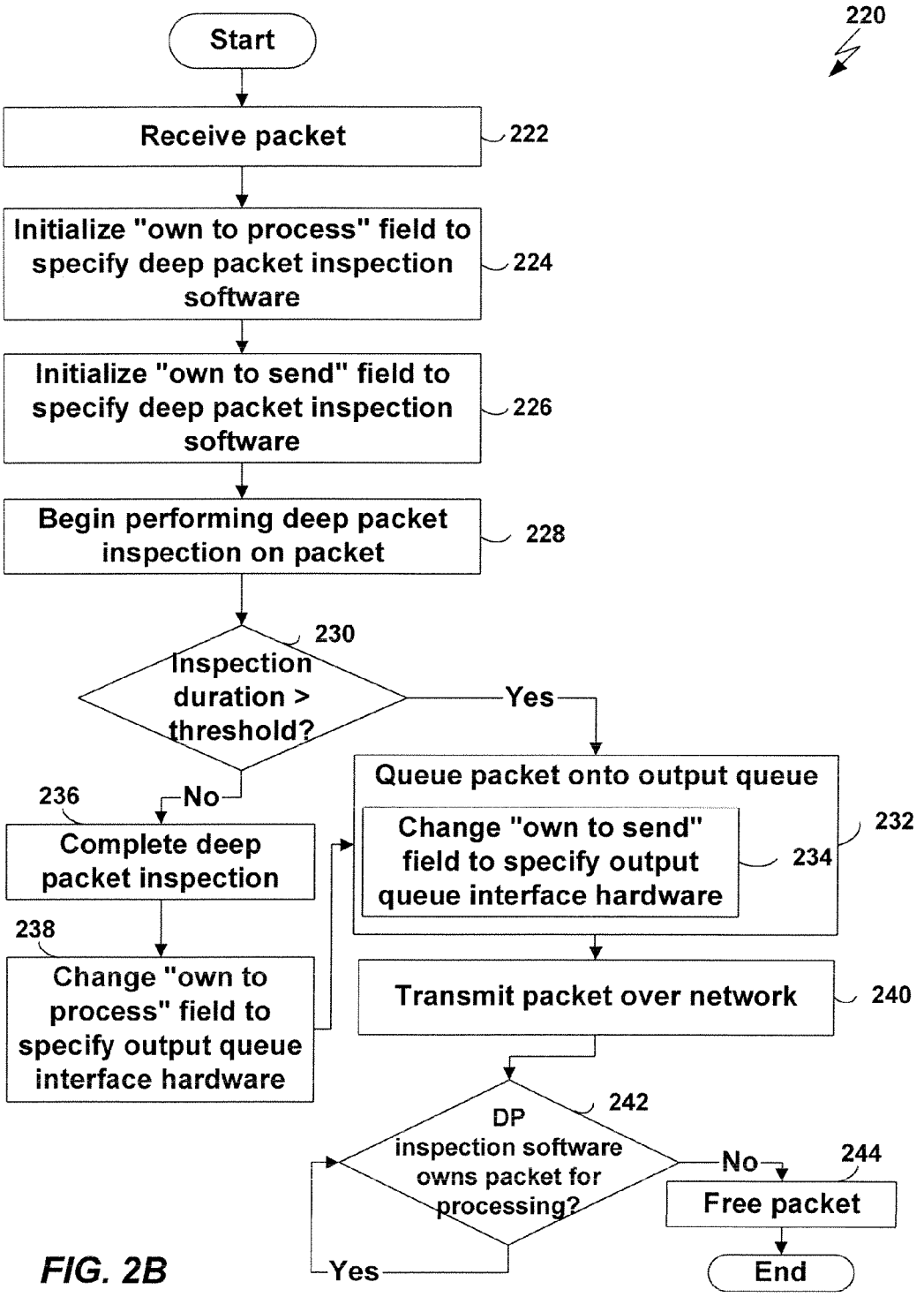

220
Start
Receive packet
222
Initialize "own to process" field to specify deep packet inspection software
224
Initialize "own to send" field to specify deep packet inspection software
226
Begin performing deep packet inspection on packet
228
230
Inspection duration > threshold?
Yes
No
236
Complete deep packet inspection
238
Change "own to process" field to specify output queue interface hardware
Queue packet onto output queue
Change "own to send" field to specify output queue interface hardware
234
232
Transmit packet over network
240
242
DP inspection software owns packet for processing?
No
244
Free packet
Yes
End

FIG. 2B

# ZERO COPY PACKET BUFFERING USING SHADOW SENDS

## RELATED APPLICATION

This application claims the benefit of U.S. Provisional Application No. 60/953,802 entitled "Zero Copy Packet Buffering Using Shadow Sends" filed Aug. 3, 2007, by Canion, et al.

## BACKGROUND

As the use of digital electronic communication networks has grown in recent years, the sophistication of internal and external network attacks in the form of viruses, Trojan horses, worms, and malware of various kinds has increased dramatically. Just as dramatic is the accelerated increase of network speeds and a corresponding drop in network costs, thereby driving the rapid adoption of networks. These and other factors have necessitated the development of innovative and more advanced network security measures.

For example, Intrusion Detection Systems (IDS) can often detect network attacks, but as passive systems they generally offer little more than after-the-fact notification. In contrast, Intrusion Prevention Systems (IPS) have been developed to complement traditional security products, such as firewalls, by proactively analyzing network traffic flows and active connections while scanning incoming and outgoing requests. As network traffic passes through the IPS, it is examined for malicious packets. Such examination may be performed by one or more "deep packet inspection engines" which perform "deep packet inspection" on some or all of the packets in the network traffic. Traffic is blocked if the IPS identifies it as posing a potential threat or as being associated with an unwanted application, while legitimate traffic is allowed to pass through the system unimpeded.

Properly implemented, an IPS can be an effective network security safeguard. There are, however, needs for improved IPS capabilities. For example, an IPS may include multiple deep packet inspection engines for performing deep packet inspection on traffic flows passing through the IPS because a single deep packet inspection engine, typically implemented as a microprocessor executing a suitable operating system and software, may not be capable of processing the flows at a sufficiently high throughput. Techniques for balancing network traffic load among multiple deep packet inspection engines in an IPS to increase the aggregate performance of such engines and thereby the overall performance of the IPS are disclosed in U.S. patent application Ser. No. 11/443,490, filed by Brian C. Smith, Alexander Sarin, and Hazem M. Kadaba on May 30, 2006, entitled "Intrusion Prevention System Edge Controller"; and U.S. patent application Ser. No. 11/782,840, filed by Gerald S. Stellenberg, Brian C. Smith, and James M. Rollette on Jul. 25, 2007, entitled "System and Method for Traffic Load Balancing to Manage Multiple Processors".

Furthermore, the amount of time required to perform deep packet inspection on a single packet may vary widely from packet to packet. This amount of processing time, referred to as "inspection latency," is affected, for example, by packet length and packet type. If the type of packet inspection applied to a particular type of packet requires that a complex regular expression ("regex") pattern be matched against the packet, the inspection latency for that packet may be many orders of magnitude greater than the packet transmission speed. For example, the transmission time of a maximum-size Ethernet packet over a gigabit Ethernet link is 12.304 microseconds. Applying deep packet inspection to a packet using a recursive regex pattern may take 10 milliseconds or longer, i.e., approximately 1,000 times longer than the transmission speed.



Conventional packet processing techniques require that processing of a packet be completed by packet inspection software before the packet can be forwarded into a hardware buffer for transmission over the network. This can introduce delays into packet transmission, particularly for packets to which regex pattern matching is applied and which have high inspection latency.

Furthermore, traditional packet processing typically requires repeatedly copying each packet in the course of processing it. For example, the typical life cycle of a packet includes copying the packet into a buffer, inspecting the packet, and copying the packet out of the buffer in order to transmit the packet. Such repeated copying of each packet requires additional hardware resources and further increases the inspection latency of each packet.

What is needed, therefore, are techniques for decreasing the transmission latency of packets, particularly those having high inspection latency, in Intrusion Prevention Systems.

## SUMMARY

Packets in an intrusion prevention system are inspected by a deep packet inspection engine. A packet may be queued for transmission onto an output queue and transmitted over a network while deep packet inspection is still being performed on the packet. Such simultaneous inspection processing and transmission may be implemented using two ownership bits for the packet, one to indicate "ownership to process" and one to indicate "ownership to send," instead of the single ownership bit that is used in conventional systems. Furthermore, the packet may be inspected, queued onto the output queue, and transmitted without making a copy of the packet within the deep packet inspection engine. These techniques enable the inspection latency, and therefore the overall transmission latency, of packets to decrease, thereby improving the overall performance of the intrusion prevent system.

For example, one embodiment of the present invention is directed to a method including: (A) receiving a packet over a network; (B) performing deep packet inspection on the packet; and (C) queuing the packet for transmission over a network while deep packet inspection is being performed on the packet. The method may further include: (D) transmitting the packet over the network while deep packet inspection is being performed on the packet.

Alternatively, for example, the method may further include: (D) determining whether a predetermined criterion associated with the packet is satisfied; and (E) performing (C) if the predetermined criterion is satisfied. The predetermined criterion may, for example, include whether deep packet inspection has been performed on the packet for longer than a predetermined amount of time.

Furthermore, the packet may be queued for transmission over the network without copying the packet.

The method may further include assigning processing ownership of the packet to deep packet inspection means and assigning transmission ownership of the packet to output means. Inspection ownership may be assigned by modifying a first value of a first field in the packet, and transmission ownership of the packet may be assigned by modifying a second value of a second field in the packet. The first and second fields may be individual bits.

Other features and advantages of various aspects and embodiments of the present invention will become apparent from the following description and from the claims.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1**A is a block diagram of a network Intrusion Prevent System (IPS) according to one embodiment of the present invention;

FIG. **1**B is a diagram of one of the deep packet inspection engines of FIG. **1**A in more detail according to one embodiment of the present invention;

FIG. **1**C is a diagram of one of the deep packet inspection engines of FIG. **1**A in more detail according to another embodiment of the present invention;

FIG. **2**A is a flowchart of a method performed by the system of FIG. **1**A according to one embodiment of the present invention; and

FIG. **2**B is a flowchart of a method performed by a deep packet inspection engine to process a packet according to one embodiment of the present invention.

## DETAILED DESCRIPTION

Referring to FIG. **1**A, a block diagram is shown of a network Intrusion Prevention System (IPS) **100** according to one embodiment of the present invention. Referring to FIG. **2**A, a flowchart is shown of a method **200** performed by the system **100** of FIG. **1**A according to one embodiment of the present invention.

A stream of packets **102** enters a load balancer **104** (step **202**), which divides the stream of packets **102** into four sub-streams **106***a-d* (step **204**), which the load balancer **104** transfers to deep packet inspection engines **108***a-d*, respectively (step **206**). Examples of techniques that may be used to implement the load balancer **104** may be found in U.S. patent application Ser. Nos. 11/443,490 and 11/782,840. Although four sub-streams **106***a-d* are shown in FIG. **1**A for purposes of example, the load balancer **104** may balance the incoming packet stream **102** onto any number of sub-streams.

Deep packet inspection engines **108***a-d* include receive queues **110***a-d*, respectively. The deep packet inspection engines **108***a-d* receive the packet sub-streams **106***a-d* and queue the packets in the sub-streams **106***a-d* onto the receive queues **110***a-d*, respectively (step **208**). The receive queues **110***a-d* are examples of "packet inspection queues," as that term is used herein. A packet inspection queue is a queue onto which packets are queued for a determination of whether they require deep packet inspection, and for deep packet inspection if required. The receive queues **110***a* may, for example, be hardware-implemented first-in first-out (FIFO) queues, according to which the packets in the sub-streams **106***a-d* are processed by the corresponding deep packet inspection engines **108***a-d* in the order in which the packets in the sub-streams **106***a-d* are received by the deep packet inspection engines **108***a-d*.

The deep packet inspection engines **108***a-d* may process their receive queues **110***a-d* in any manner. For example, referring to FIG. **1**B, deep packet inspection engine **108***a* may include deep packet inspection software **170** which dequeues packets sequentially from the receive queue **110***a* (step **210**) and applies deep packet inspection to them, thereby producing an internal stream of inspected packets **172** (step **212**). In a conventional system, this process typically involves making copies of the packets from the receive queue **110***a* for purposes of applying deep packet inspection to the packets.

Note that the inspected packets **172** may not include all of the packets **106***a* received by the deep packet inspection engine **108***a* since, for example, the deep packet inspection software **170** may drop packets which fail deep packet inspection.

The deep packet inspection software **170** queues the inspected packets **172** onto a hardware-implemented output queue **174** (step **214**). In a conventional system, queueing the packets **172** onto the output queue **174** typically involves copying the packets from a software-implemented queue to the hardware implemented queue **174**.

The deep packet inspection engine **108***a* also includes a hardware-implemented output queue interface **166** which dequeues packets **176** from the output queue **174** (step **216**) and transmits the dequeued packets **176** over the network **114** as transmitted packets **178** (step **218**). Steps **216** and **218** may, for example, be implemented by an output MAC **190** in the deep packet inspection engine **108***a*.

As mentioned above, it may be time-consuming to apply deep packet inspection to the packets **106***a*. Conventional packet processing techniques require that processing of a packet be completed by packet inspection software (such as the deep packet inspection software **170**) before the packet can be forwarded into the hardware output queue **174** for transmission over the network **114**. This can introduce delays into packet transmission, particularly for packets to which regex pattern matching is applied and which have high inspection latency.

In general, embodiments of the present invention address this problem by queuing packets onto the output queue **174** without copying the packets and without waiting for deep packet inspection of the packets to finish. In other words, in embodiments of the present invention, a packet may be queued onto the output queue **174** while the deep packet inspection software **170** is still performing deep packet inspection on the packet. The deep packet inspection engine **108***a* may even transmit the packet over the network **114** before deep packet inspection of the packet is complete. Performing deep packet inspection on packets without copying them, and enabling packets to be queued for output and even transmitted over the network **114** before deep packet inspection of the packets is complete, enables the inspection latency, and therefore the overall transmission latency, of packets to decrease, thereby improving the overall performance of the IPS **100**.

For example, referring to FIG. **1**C, the deep packet inspection engine **108***a* is shown in more detail according to one embodiment of the present invention. FIG. **1**C focuses on a single packet **112** in the output queue **174** for purposes of example. The packet **112** includes an "own to process" field **150** and an "own to send" field **152**. The own to process field **150** indicates whether the deep packet inspection software **170** is currently performing deep packet inspection on the packet **112**. For example, the own to process field **150** may be implemented as a single bit, where a value of zero indicates that the deep packet inspection software **170** is performing deep packet inspection on the packet **112**, and where a value of one indicates that the deep packet inspection software **170** is not performing deep packet inspection on the packet **112**, such as because the deep packet inspection software **170** has finished performing deep packet inspection on the packet **112** or because the deep packet inspection software **170** has aborted deep packet inspection of the packet **112**.

Similarly, the own to send field **152** indicates whether the output queue interface **166** (e.g., MAC) has authority to transmit the packet **112** over the network **114**. For example, the own to send field **152** may be implemented as a single bit,

where a value of zero indicates that the output queue interface **166** is not authorized to transmit the packet **112** over the network **114** (because, for example, the deep packet inspection software **170** has not yet queued the packet **112** onto the output queue **174**), and where a value of one indicates that the output queue interface **166** is authorized to transmit the packet **112** over the network **114** (because, for example, the deep packet inspection software **170** has queued the packet **112** onto the output queue **174**). The packet **112** may be queued onto the output queue **174** by changing a pointer in a descriptor in the output buffer **174** to point to the packet **112** and setting the value of the "own to send" bit **152** to indicate that the output queue interface **166** has ownership of the packet **112** for purposes of transmitting it over the network **114**. The packet **112** may be queued onto the output queue **174**, in other words, without copying the packet **112**.

As a result, the deep packet inspection software **170** may own the packet **112** for purposes of performing deep packet inspection on the packet **112** at the same time as the output queue interface **166** owns the packet **112** for purposes of transmitting the packet **112** over the network **114**. As described in more detail below, this enables the output queue interface **166** to transmit the packet **112** over the network **114** even while the deep packet inspection software **170** continues to perform deep packet inspection on the packet **112**.

Referring to FIG. **2**B, a flowchart is shown of a method **220** that is performed by the deep packet inspection engine **108***a* of FIG. **1**C according to one embodiment of the present invention. Although the method **220** focuses on the single packet **112**, those having ordinary skill in the art will appreciate how to integrate the method **220** shown in FIG. **2**B with the method **200** shown in FIG. **2**A.

The deep packet inspection engine **108***a* receives the packet **112** into the receive queue **110***a* (step **222**) and initializes **180** both the own to process field **150** (step **224**) and the own to send field **152** (step **226**) to specify that the packet **112** is owned by the deep packet inspection software **170**. The initialization may, for example, be performed by the deep packet inspection software **170**. Although the packet **112** is only shown in the output queue **174** in FIG. **1**B for ease of illustration, the packet **112** may initially be in the receive queue **110***a*.

The deep packet inspection software **170** begins to perform deep packet inspection on the packet **112** (step **228**). Note that the deep packet inspection software **170** may perform deep packet inspection on the packet **112** directly in the receive queue **110***a*, in other words, without making a copy of the packet **112**.

Deep packet inspection may not be performed on all packets in the receive queue **110***a*. However, for purposes of example, assume that the deep packet inspection software **170** performs deep packet inspection on the packet **112**.

If deep packet inspection of the packet **112** lasts for longer than a predetermined time threshold (step **230**), the deep packet inspection software **170** may queue **182** the packet **112** on the output queue **174** (step **232**). Queueing the packet **112** on the output queue **174** may include changing the value of the own to send field **152** to specify that the output queue interface **166** owns the packet **112** (step **234**), while leaving the value of the own to process field **150** unchanged. In fact, the packet **112** may be queued onto the output queue **174** solely by changing the value of the own to send field **152** to specify that the output queue interface **166** owns the packet **112**. As a result, the deep packet inspection software **170** retains ownership of the packet **112** for purposes of performing deep packet inspection on the packet **112**, while the output queue interface **166** gains ownership of the packet **112** for purposes of preparing the packet **112** for transmission over the network **114** and for performing such transmission.

Therefore, the deep packet inspection software **170** may continue to perform deep packet inspection on the packet **112** even after the packet is queued onto the output queue **174** in step **232**.

Although in the embodiment illustrated in FIG. **2**B, the packet **112** is only queued onto the output queue **174** before deep packet inspection of the packet **112** is completed if deep packet inspection of the packet **112** lasts longer than the predetermined threshold, this is not a limitation of the present invention. For example, all packets may be queued onto the output queue **174** while deep packet inspection is being performed on them, even if no deep packet inspection threshold is exceeded.

The packet **112** may be queued onto the output queue **174** without copying the packet **112**. Therefore, although the receive queue **110***a* and the output queue **174** are shown as separate components in FIG. **1**C, this is not a requirement of the present invention. Rather, for example, the packet **112** may be queued onto the output queue **174** by changing a pointer to the packet **112** or by changing the value of an additional field (not shown) in the packet **112**, but without copying the packet **112**.

Returning to the embodiment illustrated in FIG. **2**B, if deep packet inspection of the packet **112** completes before the inspection time threshold is exceeded (step **236**), the deep packet inspection software **170** changes **184** the value of the own to process field **150** of the packet **112** to specify the output queue interface **166** (step **238**), and queues the packet **112** onto the output queue **174** (step **232**).

The output queue interface **166** may perform additional processing on the packet **112** before transmitting the packet **112** over the network **114** (step **240**). The output queue interface **166** determines whether the deep packet inspection software **170** is still performing deep packet inspection on the packet **112** by, for example, determining whether the own to process **150** field of the packet **112** still specifies the deep packet inspection software **170** (step **242**). If the output queue interface **166** determines that the deep packet inspection software **170** is still performing deep packet inspection on the packet **112**, then the output queue interface **166** waits until the deep packet inspection software **170** is no longer processing the packet **112**.

One deep packet inspection software **170** is no longer performing deep packet inspection on the packet **112**, the output queue interface **166** frees the position of the packet **112** in the output queue **174** for use by subsequent packets (step **244**). The position of the packet **112** in the output queue **174** may be freed, for example, by changing the value of the packet's "own to process" field **150** to indicate that the deep packet inspection software **170** no longer owns the packet **112** for processing.

Since deep packet inspection of the packet **112** may be ongoing even after the deep packet inspection engine **108***a* has transmitted the packet **112** over the network **114**, it is possible that deep packet inspection of the packet **112** will fail after the packet **112** has been transmitted over the network. In the case of such a failure, the state of the flow (session) of which the packet **112** is a part may be changed to "discard." As a result, subsequent packets in the same flow will be marked as "discard," thereby causing the recipient of the flow to discard the flow's packets. Furthermore, the packet's flow may be marked as a high priority flow for purposes of deep packet inspection, thereby causing the deep packet inspection engine **108***a* to take up subsequent packets in that flow ahead of packets with lower priority. The deep packet inspection

engine **108***a* will process such high priority packets quickly because, as part of a flow whose status is "discard," deep packet inspection need not be applied to them.

One advantage of embodiments of the present invention is that they enable packets to be queued for output and even transmitted over a network while deep packet inspection continues to be performed on such packets. As a result, deep packet inspection does not delay packet transmission, even in cases where deep packet inspection lasts for a particularly long time. The techniques disclosed herein therefore reduce total inspection latency, thereby reducing total transmission latency.

Embodiments of the present invention provide such improved performance without sacrificing security, because even if a packet fails deep packet inspection after it has been transmitted, the packet may still effectively be terminated by marking its flow as "discard" or taking other steps to ensure that the packet's recipient is not harmed by it. For example, the techniques disclosed above with respect to FIG. **2**C may be used to enable the message containing a packet to be terminated if deep packet inspection of the packet fails after it has been transmitted.

Furthermore, embodiments of the present invention enable deep packet inspection and other processing of packets to be performed without making multiple copies of each packet. This further decreases both inspection latency and transmission latency, reduces memory consumption, and enables devices such as the deep packet inspection engine to be more compact and manufactured at lower cost.

It is to be understood that although the invention has been described above in terms of particular embodiments, the foregoing embodiments are provided as illustrative only, and do not limit or define the scope of the invention. Various other embodiments, including but not limited to the following, are also within the scope of the claims. For example, elements and components described herein may be further divided into additional components or joined together to form fewer components for performing the same functions.

Although in certain embodiments disclosed herein there are multiple deep packet inspection engines **108***a-d*, this is not a requirement of the present invention. Embodiments of the present invention may be applied, for example, to systems including only a single deep packet inspection engine.

Although in certain embodiments disclosed herein, the deep packet inspection engine **108***a* both reduces or eliminates internal copying of the packet **112** and enables the packet **112** to be inspected and transmitted simultaneously, all of these features need not be implemented together. For example, reduction or elimination of packet copying may be implemented without simultaneous packet inspection and transmission. Conversely, simultaneous packet inspection and transmission may be implemented without reduction or elimination of packet copying.

In the embodiment illustrated in FIG. **1**C, the ownership states of the packet **112** are implemented using two ownership fields **150** and **152**. The ownership states of the packet **112** may, however, be implemented in other ways, such as by descriptors, packet headers, or some internal state which is linked to the output queue **174**. Furthermore, although each of the ownership fields **150** and **152** is described herein as a one-bit field having two possible values, this is not a limitation of the present invention. Rather, the ownership states of the packet **112** may be represented in any manner.

Although certain components, such as the deep packet inspection software **170**, are described herein as being implemented in software, and certain components, such as the output queue interface **166**, are described herein as being implemented in hardware, these are not limitations of the present invention. More generally, the techniques described above may be implemented, for example, in hardware, software, firmware, or any combination thereof. The techniques described above may be implemented in one or more computer programs executing on a programmable computer including a processor, a storage medium readable by the processor (including, for example, volatile and non-volatile memory and/or storage elements), at least one input device, and at least one output device. Program code may be applied to input entered using the input device to perform the functions described and to generate output. The output may be provided to one or more output devices.

Although in certain embodiments described herein, packets are simultaneously transmitted and inspected if deep packet inspection of such packets lasts longer than a predetermined threshold, this is not a requirement of the present invention. For example, "dual ownership" of packets by the deep packet inspection software **170** and the output queue interface **166** may be applied to all packets without discrimination.

Furthermore, criteria other than or in addition to the predetermined time threshold may be used at step **230** in FIG. **2**B to determine whether to allow transmission of a packet while it is still being inspected. For example, the decision whether to allow simultaneous ownership of a packet by both the deep packet inspection software **170** and the output queue interface **166** may be made based on characteristics of the flow of which the packet is a part. For example, dual ownership of the packet **112** may be allowed if the packet **112** is part of a flow for which termination of subsequent packets (such as by marking their status as "discard") will cause the recipient to discard packet **112**. If the packet **112** is not part of a flow having this characteristic, for example, then dual ownership of the packet **112** may not be allowed.

As another example, if an entire flow is contained in the packet **112**, then dual ownership of the packet **112** may be disallowed because it may not be possible in such a case to prevent harmful effects of the packet **112** if deep packet inspection of the packet **112** fails after it is transmitted.

Each computer program within the scope of the claims below may be implemented in any programming language, such as assembly language, machine language, a high-level procedural programming language, or an object-oriented programming language. The programming language may, for example, be a compiled or interpreted programming language.

Each such computer program may be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a computer processor. Method steps of the invention may be performed by a computer processor executing a program tangibly embodied on a computer-readable medium to perform functions of the invention by operating on input and generating output. Suitable processors include, by way of example, both general and special purpose microprocessors. Generally, the processor receives instructions and data from a read-only memory and/or a random access memory. Storage devices suitable for tangibly embodying computer program instructions include, for example, all forms of non-volatile memory, such as semiconductor memory devices, including EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROMs. Any of the foregoing may be supplemented by, or incorporated in, specially-designed ASICs (application-specific integrated circuits) or FPGAs (Field-Programmable Gate Arrays). A computer can generally also receive pro-

grams and data from a storage medium such as an internal disk (not shown) or a removable disk. These elements will also be found in a conventional desktop or workstation computer as well as other computers suitable for executing computer programs implementing the methods described herein, which may be used in conjunction with any digital print engine or marking engine, display monitor, or other raster output device capable of producing color or gray scale pixels on paper, film, display screen, or other output medium.

What is claimed is:

**1**. A method comprising:

(A) receiving a packet over a network;

(B) performing deep packet inspection on the packet;

(C) queuing the packet for transmission over a network while deep packet inspection is being performed on the packet;

before (B), assigning processing ownership of the packet to deep packet inspection means;

before (C), assigning transmission ownership of the packet to output means,

wherein (B) comprises performing deep packet inspection on the packet using the deep packet inspection means, and

wherein assigning processing ownership of the packet comprises modifying a first value of a first field in the packet and wherein assigning transmission ownership of the packet comprises modifying a second value of a second field in the packet.

**2**. The method of claim **1**, further comprising:

(D) transmitting the packet over the network while deep packet inspection is being performed on the packet.

**3**. The method of claim **1**, further comprising:

(D) determining whether a predetermined criterion associated with the packet is satisfied; and

(E) performing (C) if the predetermined criterion is satisfied.

**4**. The method of claim **3**, wherein (D) comprises determining whether deep packet inspection has been performed on the packet for longer than a predetermined amount of time.

**5**. The method of claim **1**, wherein (C) comprises queuing the packet for transmission over the network without copying the packet.

**6**. The method of claim **1**, wherein the first field comprises a first bit and wherein the second field comprises a second bit.

**7**. The method of claim **1**, wherein queuing the packet for transmission over a network comprises queuing the packet in an output queue for transmission of the packet from the output queue over the network.

**8**. The method of claim **7**, wherein the output queue is a hardware output queue.

**9**. A non-transitory computer readable storage medium on which is embedded machine readable instructions, said machine readable instructions, when executed, implementing a method of deep packet inspection, said machine readable instructions comprising computer readable code to:

receive a packet over a network;

perform deep packet inspection on the packet;

queue the packet for transmission over a network while deep packet inspection is being performed on the packet;

prior to perform the deep packet inspection on the packet, modify a first value of a first field in the packet to indicate that the deep packet inspection is performed on the packet; and

prior to queue the packet for transmission, modify a second value of a second field in the packet to indicate that the packet is queued in an output queue for transmission.

**10**. A network device comprising:

a hardware output queue; and

a deep packet inspection engine to:

perform deep packet inspection on a packet received over a network; and

queue the packet in the output queue for transmission over the network while deep packet inspection is being performed on the packet,

wherein prior to performing the deep packet inspection on the packet, the deep packet inspection engine modifies a first value of a first field in the packet to indicate that the deep packet inspection is performed on the packet, and

wherein the prior to queuing the packet for transmission, the deep packet inspection engine modifies a second value of a second field in the packet to indicate that the packet is queued in the output queue for transmission.

* * * * *