US 20090031136A1

(19) United States
(12) Patent Application Publication
Milliken et al.

(10) Pub. No.: US 2009/0031136 A1
(43) Pub. Date: Jan. 29, 2009

(54) HASH-BASED SYSTEMS AND METHODS FOR DETECTING AND PREVENTING TRANSMISSION OF UNWANTED E-MAIL

(76) Inventors: **Walter Clark Milliken**, Dover, NH (US); **William Timothy Strayer**, West Newton, MA (US); **Stephen Douglas Milligan**, Stow, MA (US)

Correspondence Address:
Zilka-Kotab, PC
P.O. BOX 721120
SAN JOSE, CA 95172-1120 (US)

(21) Appl. No.: **12/243,778**

(22) Filed: **Oct. 1, 2008**

Related U.S. Application Data

(60) Division of application No. 10/654,771, filed on Sep. 4, 2003, which is a continuation-in-part of application No. 10/251,403, filed on Sep. 20, 2002, now Pat. No. 7,328,349, said application No. 10/654,771 is a continuation-in-part of application No. 09/881,145, filed on Jun. 14, 2001, now abandoned, said application No. 10/654,771 is a continuation-in-part of application No. 09/881,074, filed on Jun. 14, 2001, now Pat. No. 6,981, 158.

(60) Provisional application No. 60/407,975, filed on Sep. 5, 2002, provisional application No. 60/341,462, filed on Dec. 14, 2001, provisional application No. 60/212, 425, filed on Jun. 19, 2000, provisional application No. 60/212,425, filed on Jun. 19, 2000.

Publication Classification

(51) **Int. Cl.**
G06F 21/00 (2006.01)

(52) **U.S. Cl.** **713/176**

(57) **ABSTRACT**

A system (**120**) detects transmission of potentially unwanted e-mail messages. The system (**120**) may receive e-mail messages and generate hash values based on one or more portions of the e-mail messages. The system (**120**) may then determine whether the generated hash values match hash values associated with prior e-mail messages. The system (**120**) may determine that one of the e-mail messages is a potentially unwanted e-mail message when one or more of the generated hash values associated with the e-mail message match one or more of the hash values associated with the prior e-mail messages.

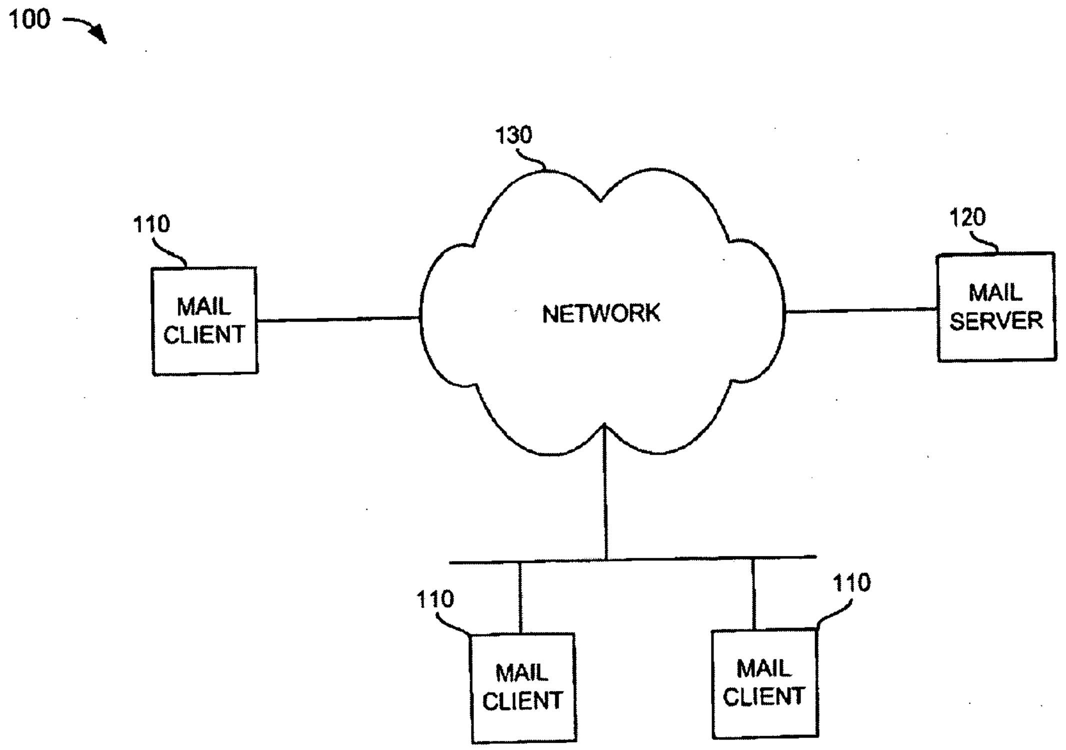
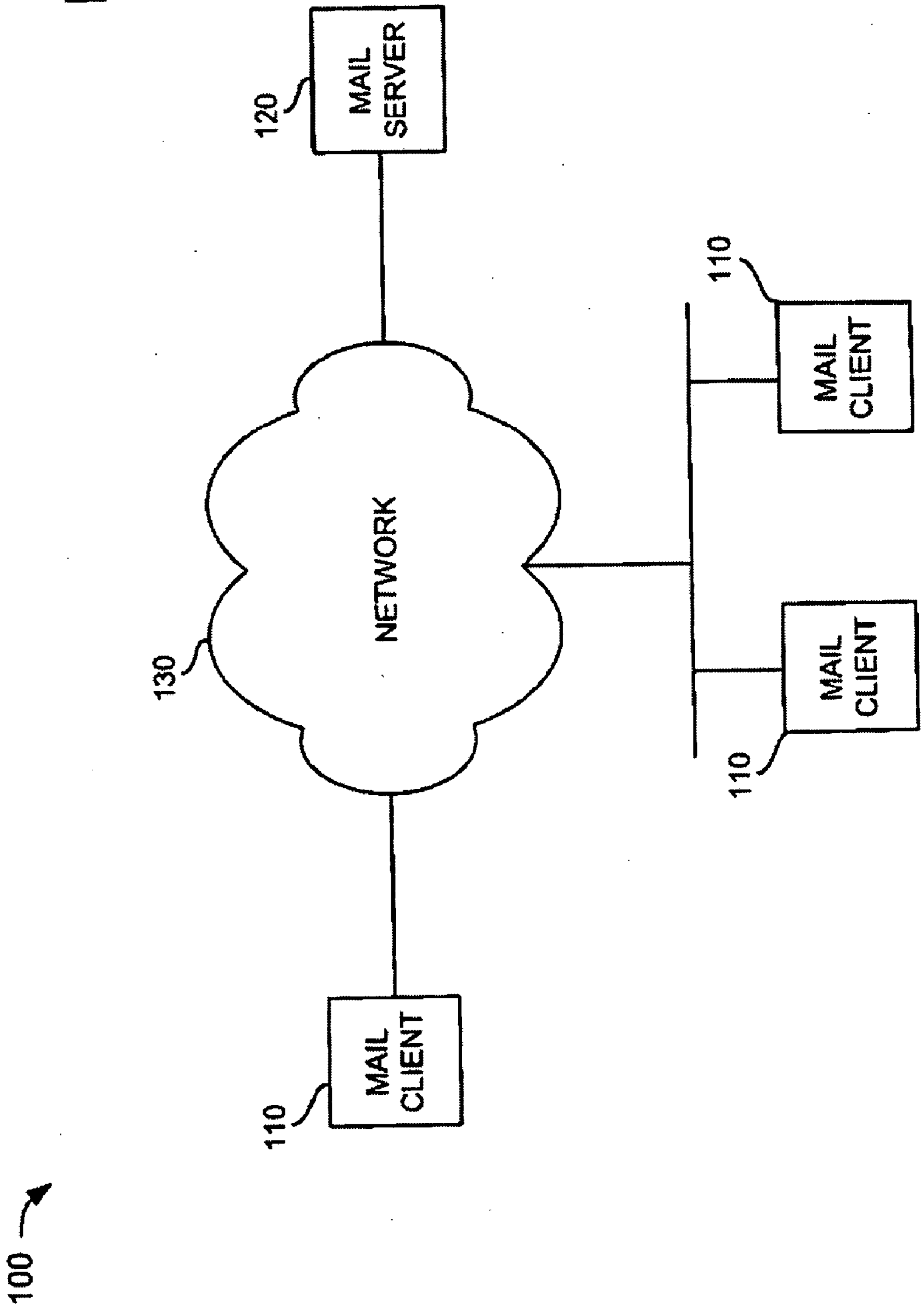


FIG. 1


100
110
MAIL CLIENT
130
NETWORK
120
MAIL SERVER
110
MAIL CLIENT
110
MAIL CLIENT

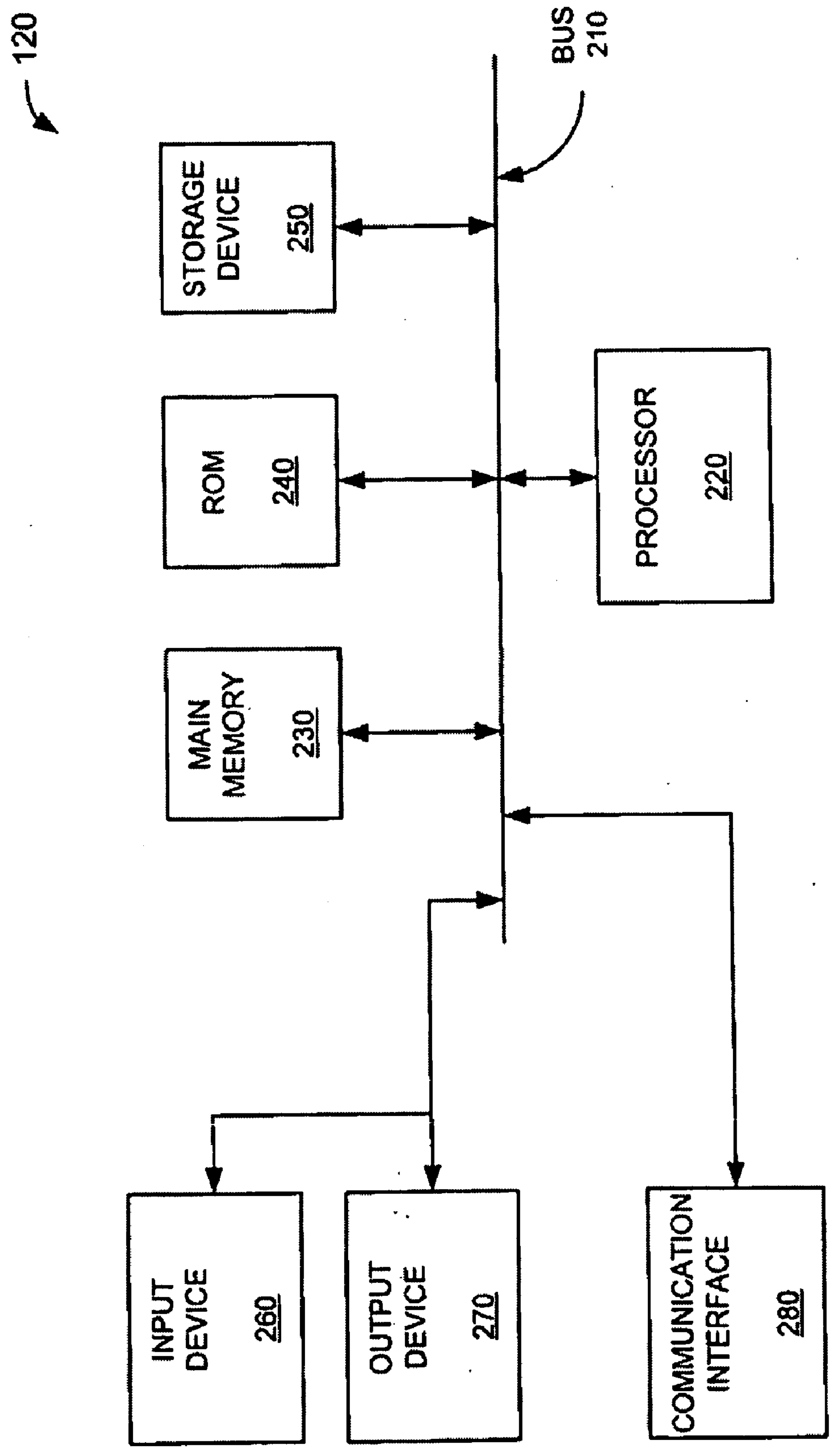

120
BUS 210
STORAGE DEVICE 250
ROM 240
MAIN MEMORY 230
PROCESSOR 220
INPUT DEVICE 260
OUTPUT DEVICE 270
COMMUNICATION INTERFACE 280

FIG. 2

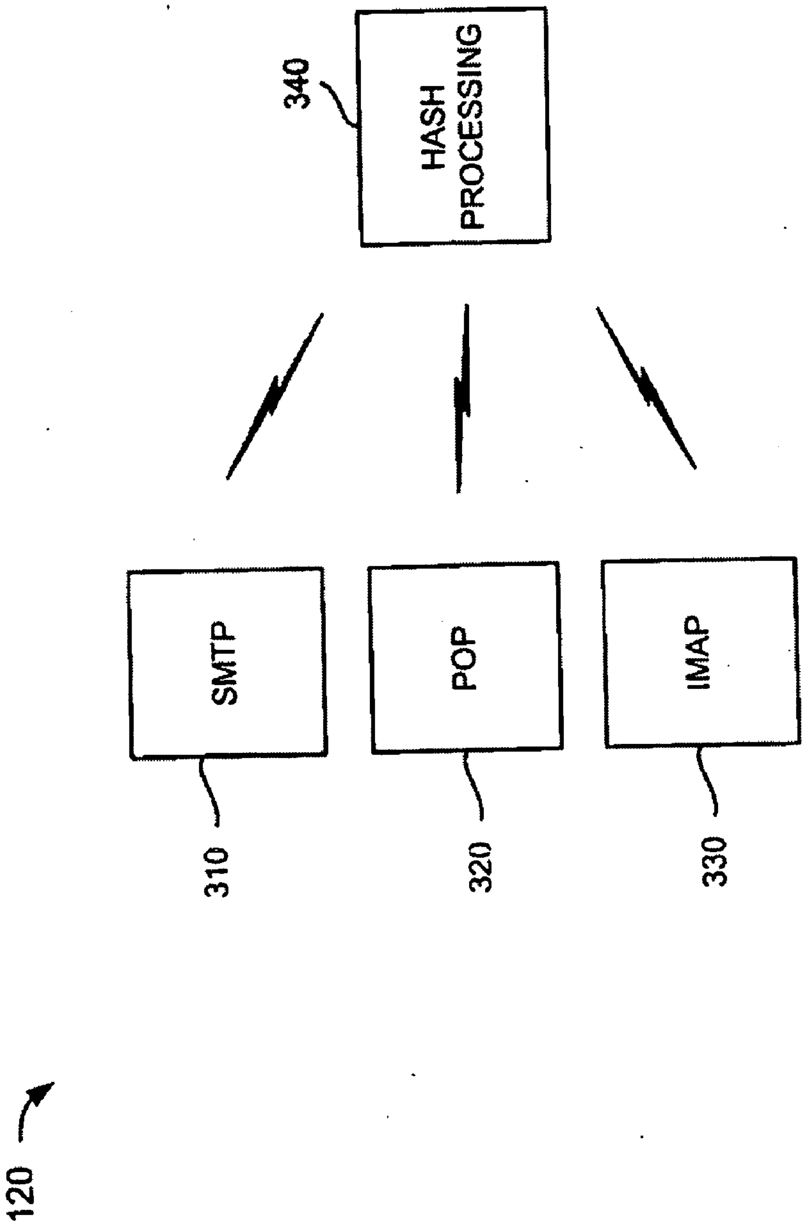


FIG. 3

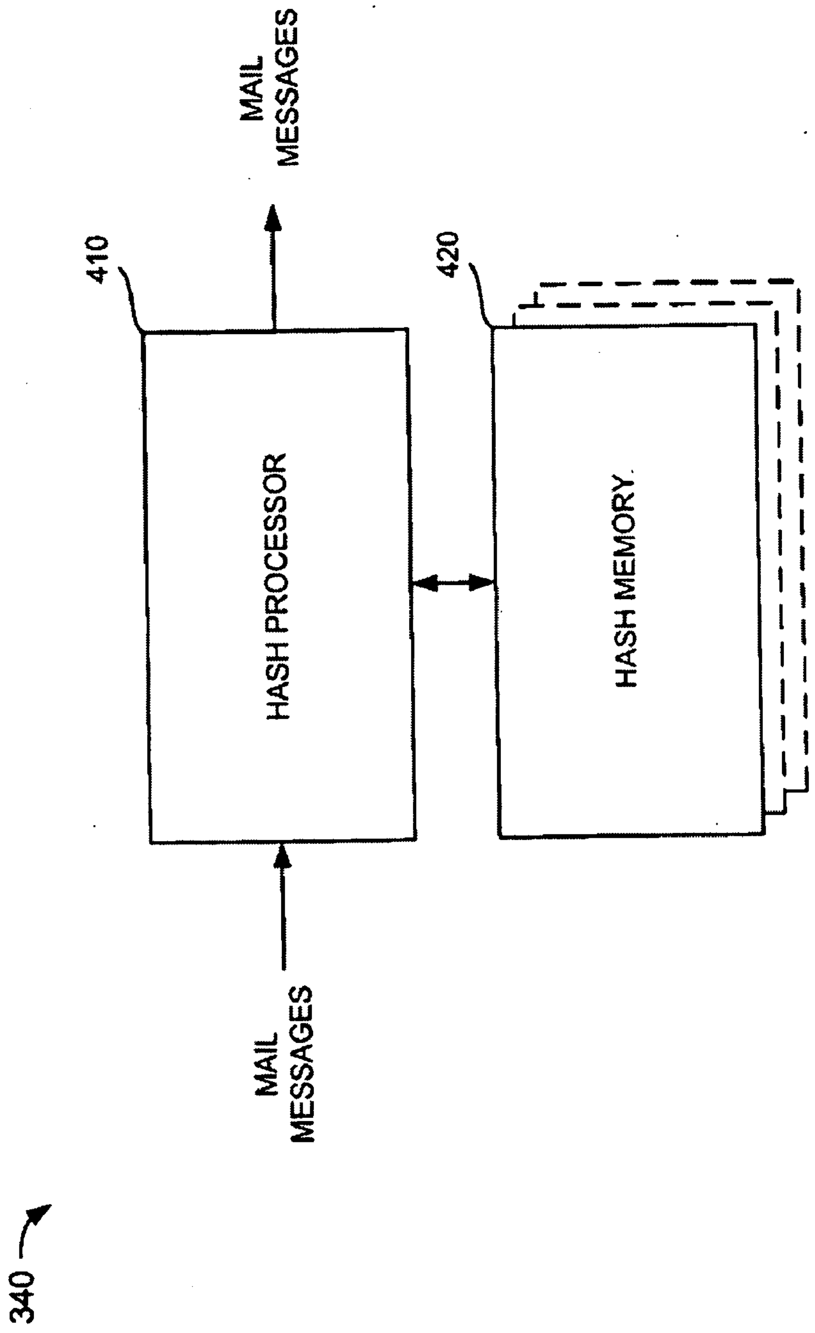

340
MAIL MESSAGES
410
HASH PROCESSOR
MAIL MESSAGES
420
HASH MEMORY

FIG. 4

FIG. 5A

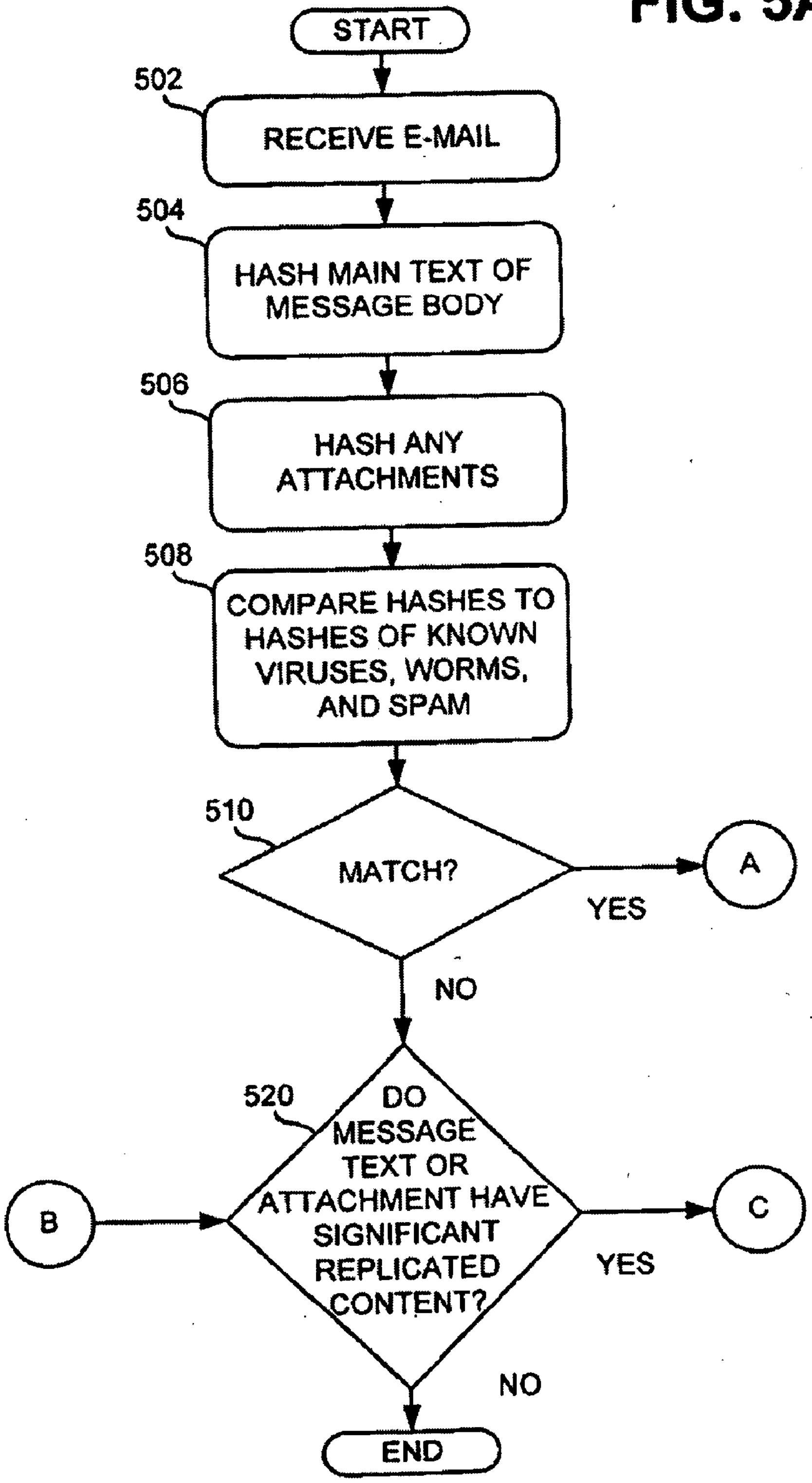

START
502
RECEIVE E-MAIL
504
HASH MAIN TEXT OF MESSAGE BODY
506
HASH ANY ATTACHMENTS
508
COMPARE HASHES TO HASHES OF KNOWN VIRUSES, WORMS, AND SPAM
510
MATCH?
YES
A
NO
520
DO MESSAGE TEXT OR ATTACHMENT HAVE SIGNIFICANT REPLICATED CONTENT?
B
YES
C
NO
END

FIG. 5B

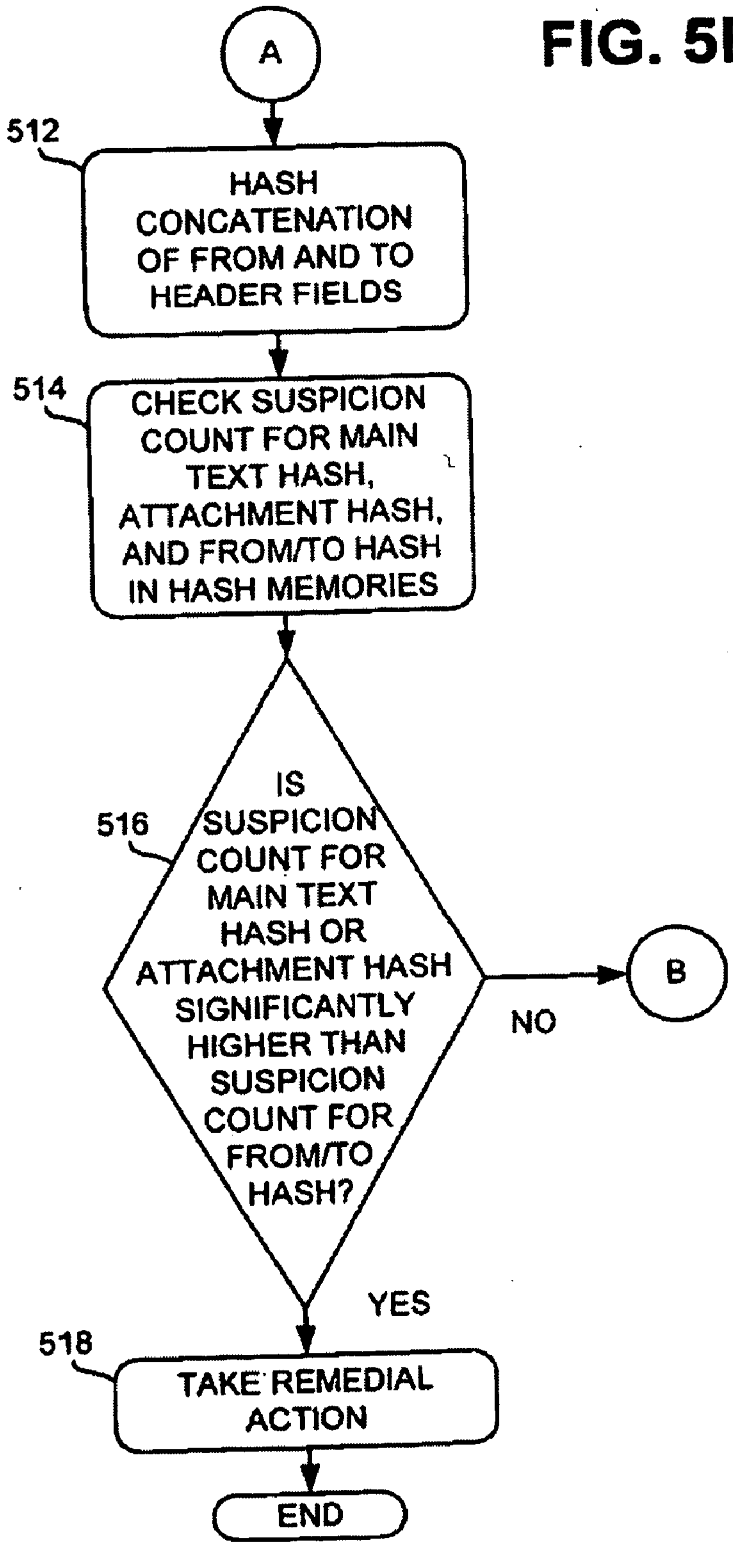

A
512
HASH CONCATENATION OF FROM AND TO HEADER FIELDS
514
CHECK SUSPICION COUNT FOR MAIN TEXT HASH, ATTACHMENT HASH, AND FROM/TO HASH IN HASH MEMORIES
516
IS SUSPICION COUNT FOR MAIN TEXT HASH OR ATTACHMENT HASH SIGNIFICANTLY HIGHER THAN SUSPICION COUNT FOR FROM/TO HASH?
NO
B
YES
518
TAKE REMEDIAL ACTION
END

FIG. 5C

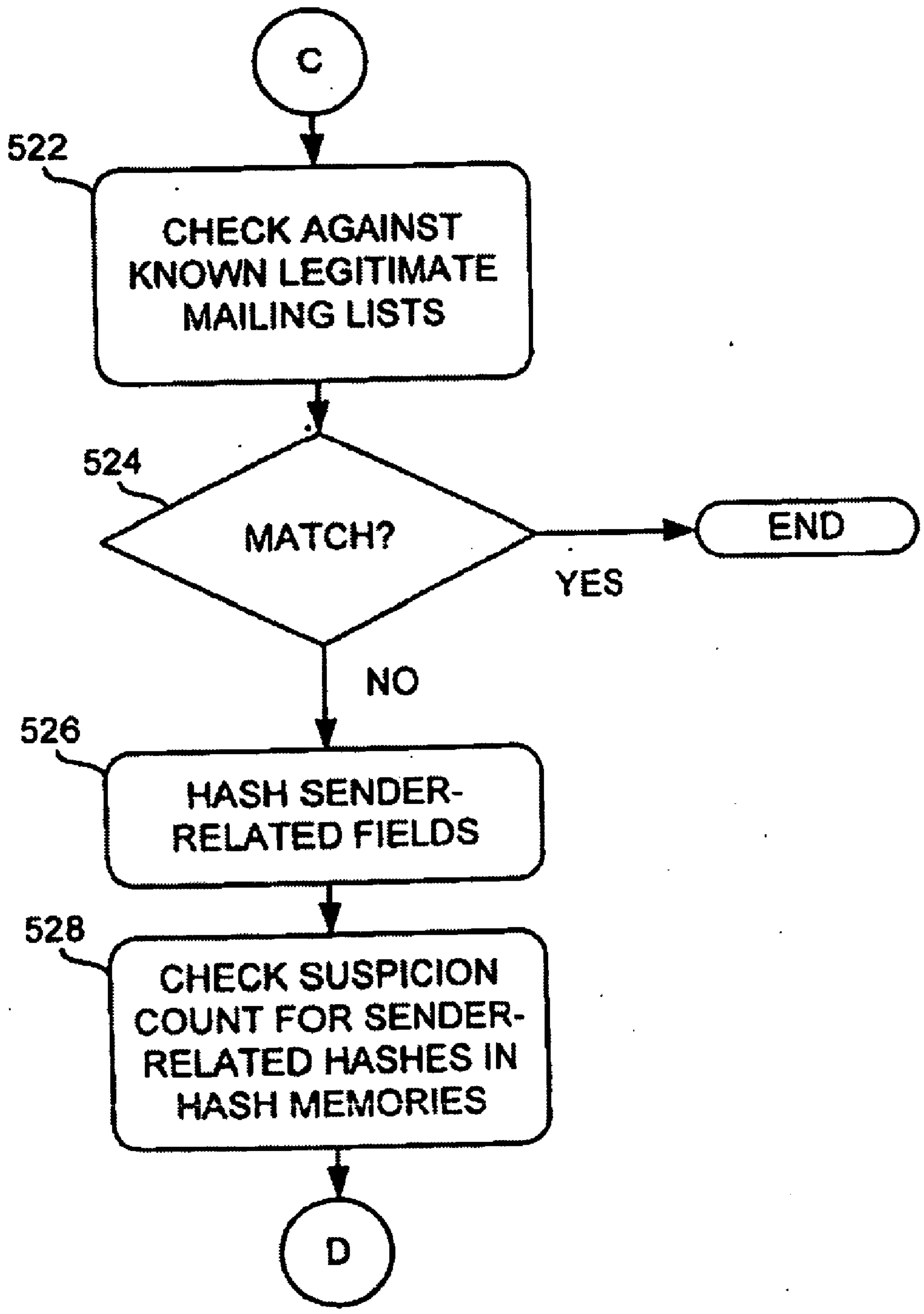

C
522
CHECK AGAINST KNOWN LEGITIMATE MAILING LISTS
524
MATCH?
YES
END
NO
526
HASH SENDER-RELATED FIELDS
528
CHECK SUSPICION COUNT FOR SENDER-RELATED HASHES IN HASH MEMORIES
D

FIG. 5D

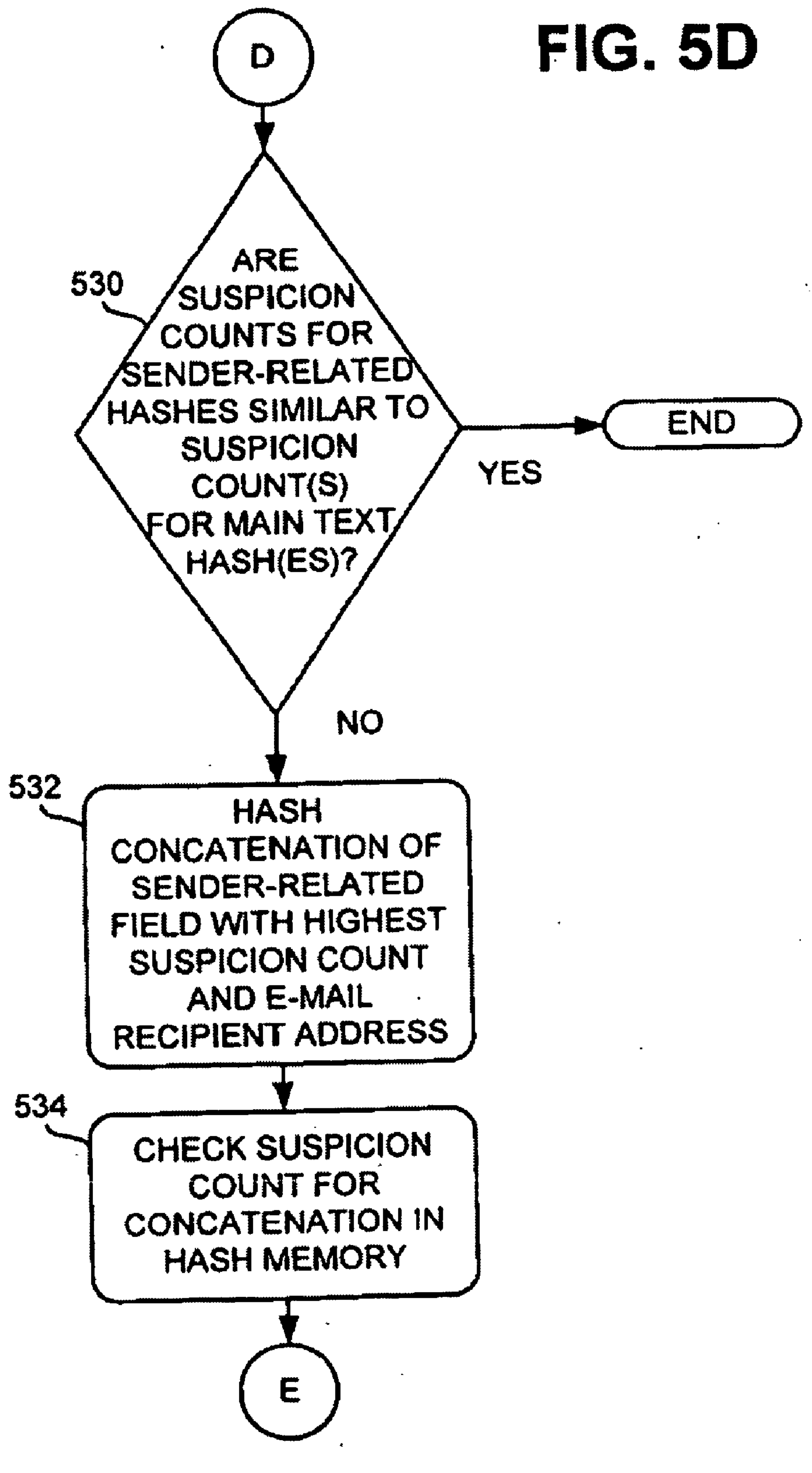
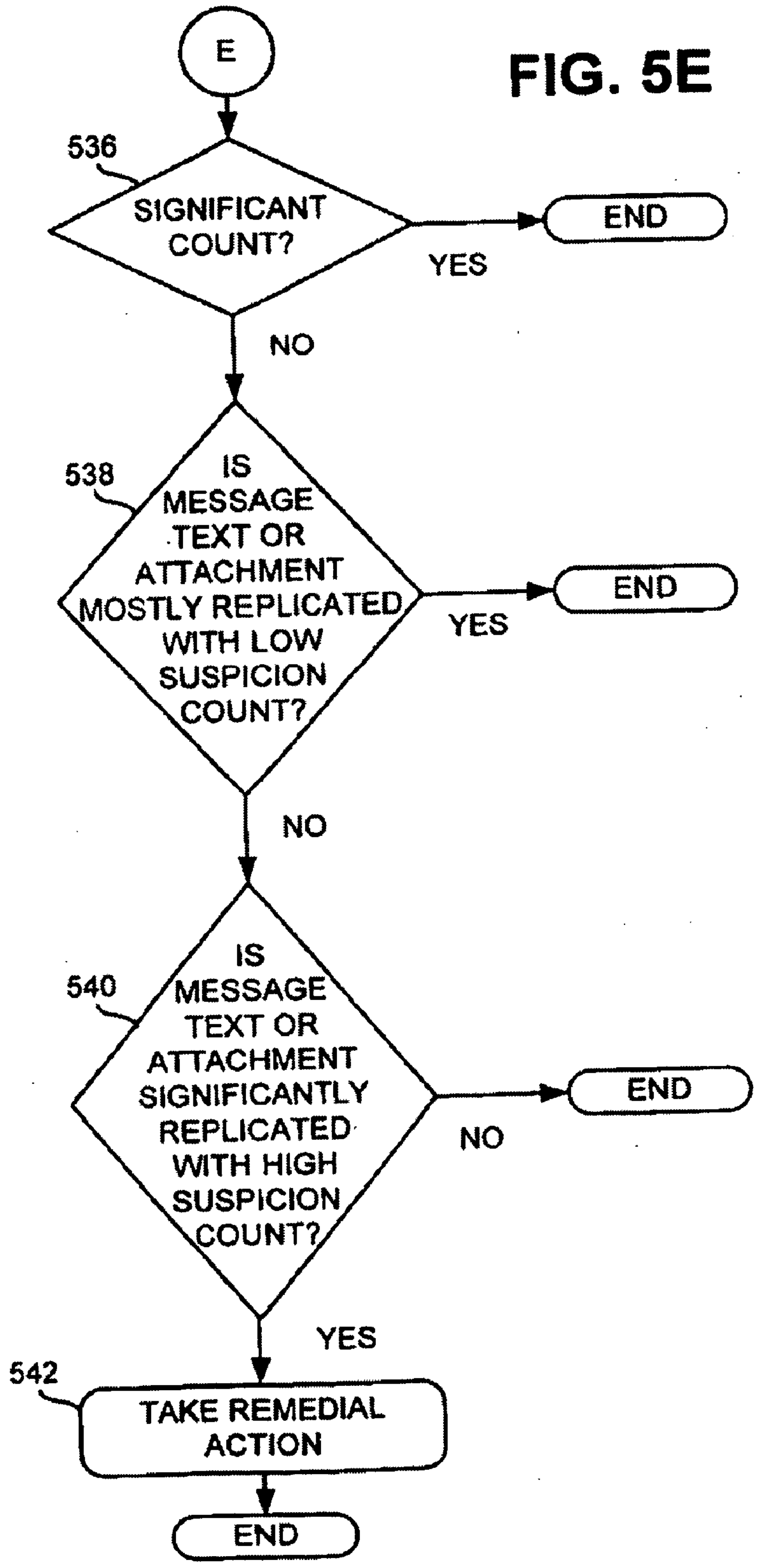

D
530
ARE SUSPICION COUNTS FOR SENDER-RELATED HASHES SIMILAR TO SUSPICION COUNT(S) FOR MAIN TEXT HASH(ES)?
YES
END
NO
532
HASH CONCATENATION OF SENDER-RELATED FIELD WITH HIGHEST SUSPICION COUNT AND E-MAIL RECIPIENT ADDRESS
534
CHECK SUSPICION COUNT FOR CONCATENATION IN HASH MEMORY
E

FIG. 5E


E
536
SIGNIFICANT COUNT?
YES
END
NO
538
IS MESSAGE TEXT OR ATTACHMENT MOSTLY REPLICATED WITH LOW SUSPICION COUNT?
YES
END
NO
540
IS MESSAGE TEXT OR ATTACHMENT SIGNIFICANTLY REPLICATED WITH HIGH SUSPICION COUNT?
NO
END
YES
542
TAKE REMEDIAL ACTION
END

HASH-BASED SYSTEMS AND METHODS FOR DETECTING AND PREVENTING TRANSMISSION OF UNWANTED E-MAIL

RELATED APPLICATIONS

[0001] This application is a divisional of U.S. patent application Ser. No. 10/654,771, filed Sep. 4, 2003, which, in turn, claims priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/407,975, filed Sep. 5, 2002, both of which are incorporated herein by reference. U.S. patent application Ser. No. 10/654,771 is also a continuation-in-part of U.S. patent application Ser. No. 10/251,403, filed Sep. 20, 2002, which claims priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/341,462, filed Dec. 14, 2001, both of which are incorporated herein by reference. U.S. patent application Ser. No. 10/654,771 is also a continuation-in-part of U.S. patent application Ser. No. 09/881,145, and U.S. patent application Ser. No. 09/881,074, both of which were filed on Jun. 14, 2001, and both of which claim priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/212,425, filed Jun. 19, 2000, all of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

[0002] 1. Field of the Invention

[0003] The present invention relates generally to network security and, more particularly, to systems and methods for detecting and/or preventing the transmission of unwanted e-mails, such as e-mails containing worms and viruses, including polymorphic worms and viruses, and unsolicited commercial e-mails.

[0004] 2. Description of Related Art

[0005] Availability of low cost computers, high speed networking products, and readily available network connections has helped fuel the proliferation of the Internet. This proliferation has caused the Internet to become an essential tool for both the business community and private individuals. Dependence on the Internet arises, in part, because the Internet makes it possible for multitudes of users to access vast amounts of information and perform remote transactions expeditiously and efficiently. Along with the rapid growth of the Internet have come problems arising from attacks from within the network and the shear volume of commercial e-mail. As the size of the Internet continues to grow, so does the threat posed to users of the Internet.

[0006] Many of the problems take the form of e-mail. Viruses and worms often masquerade within e-mail messages for execution by unsuspecting e-mail recipients. Unsolicited commercial e-mail, or "spam," is another burdensome type of e-mail because it wastes both the time and resources of the e-mail recipient.

[0007] Existing techniques for detecting viruses, worms, and spam examine each e-mail message individually. In the case of viruses and worms, this typically means examining attachments for byte-strings found in known viruses and worms (possibly after uncompressing or de-archiving attached files), or simulating execution of the attachment in a "safe" compartment and examining its behaviors. Similarly, existing spam filters usually examine a single e-mail message looking for heuristic traits commonly found in unsolicited commercial e-mail, such as an abundance of Uniform Resource Locators (URLs), heavy use of all-capital-letter words, use of colored text or large fonts, and the like, and then "score" the message based on the number and types of such traits found. Both the anti-virus and the anti-spam techniques can demand significant processing of each message, adding to the resource burden imposed by unwanted e-mail. Neither technique makes use of information collected from other recent messages.

[0008] Thus, there is need for an efficient technique that can quickly detect viruses, worms, and spam in e-mail messages arriving at e-mail servers, possibly by using information contained in multiple recent messages to detect unwanted mail more quickly and efficiently.

SUMMARY OF THE INVENTION

[0009] Systems and methods consistent with the present invention address this and other needs by providing a new defense that detects and prevents the transmission of unwanted (and potentially unwanted) e-mail, such as e-mails containing viruses, worms, and spam.

[0010] In accordance with an aspect of the invention as embodied and broadly described herein, a method for detecting transmission of potentially unwanted e-mail messages is provided. The method includes receiving e-mail messages and generating hash values based on one or more portions of the e-mail messages. The method further includes determining whether the generated hash values match hash values associated with prior e-mail messages. The method may also include determining that one of the e-mail messages is a potentially unwanted e-mail message when one or more of the generated hash values associated with the e-mail message match one or more of the hash values associated with the prior e-mail messages.

[0011] In accordance with another aspect of the invention, a mail server includes one or more hash memories and a hash processor. The one or more hash memories is/are configured to store count values associated with hash values. The hash processor is configured to receive an e-mail message, hash one or more portions of the e-mail message to generate hash values, and increment the count values corresponding to the generated hash values. The hash processor is further configured to determine whether the e-mail message is a potentially unwanted e-mail message based on the incremented count values.

[0012] In accordance with yet another aspect of the invention, a method for detecting transmission of unwanted e-mail messages is provided. The method includes receiving e-mail messages and detecting unwanted e-mail messages of the received e-mail messages based on hashes of previously received e-mail messages, where multiple hashes are performed on each of the e-mail messages.

[0013] In accordance with a further aspect of the invention, a method for detecting transmission of potentially unwanted e-mail messages is provided. The method includes receiving an e-mail message; generating hash values over blocks of the e-mail message, where the blocks include at least two of a main text portion, an attachment portion, and a header portion of the e-mail message; determining whether the generated hash values match hash values associated with prior e-mail messages; and determining that the e-mail message is a potentially unwanted e-mail message when one or more of the generated hash values associated with the e-mail message match one or more of the hash values associated with the prior e-mail messages.

[0014] In accordance with another aspect of the invention, a mail server in a network of cooperating mail servers is provided. The mail server includes one or more hash memories and a hash processor. The one or more hash memories is/are configured to store information relating to hash values corresponding to previously-observed e-mails. The hash processor is configured to receive at least some of the hash values from another one or more of the cooperating mail servers and store information relating to the at least some of the hash values in at least one of the one or more hash memories. The hash processor is further configured to receive an e-mail message, hash one or more portions of the received e-mail message to generate hash values, determine whether the generated hash values match the hash values corresponding to previously-observed e-mails, and identify the received e-mail message as a potentially unwanted e-mail message when one or more of the generated hash values associated with the received e-mail message match one or more of the hash values corresponding to previously-observed e-mails.

[0015] In accordance with yet another aspect of the invention, a mail server is provided. The mail server includes one or more hash memories and a hash processor. The one or more hash memories is/are configured to store count values associated with hash values. The hash processor is configured to receive e-mail messages, hash one or more portions of the received e-mail messages to generate hash values, increment the count values corresponding to the generated hash values, as incremented count values, and generate suspicion scores for the received e-mail messages based on the incremented count values.

[0016] In accordance with a further aspect of the invention, a method for preventing transmission of unwanted e-mail messages is provided. The method includes receiving an e-mail message; generating hash values over portions of the e-mail message as the e-mail message is being received; and incrementally determining whether the generated hash values match hash values associated with prior e-mail messages. The method further includes generating a suspicion score for the e-mail message based on the incremental determining; and rejecting the e-mail message when the suspicion score of the e-mail message is above a threshold.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate the invention and, together with the description, explain the invention. In the drawings,

[0018] FIG. **1** is a diagram of a system in which systems and methods consistent with the present invention may be implemented;

[0019] FIG. **2** is an exemplary diagram of the e-mail server of FIG. **1** according to an implementation consistent with the principles of the invention;

[0020] FIG. **3** is an exemplary functional block diagram of the e-mail server of FIG. **2** according to an implementation consistent with the principles of the invention;

[0021] FIG. **4** is an exemplary diagram of the hash processing block of FIG. **3** according to an implementation consistent with the principles of the invention; and

[0022] FIGS. **5**A-**5**E are flowcharts of exemplary processing for detecting and/or preventing transmission of an unwanted e-mail message, such as an e-mail containing a virus or worm, including a polymorphic virus or worm, or an unsolicited commercial e-mail, according to an implementation consistent with the principles of the invention.

DETAILED DESCRIPTION

[0023] The following detailed description of the invention refers to the accompanying drawings. The same reference numbers in different drawings may identify the same or similar elements. Also, the following detailed description does not limit the invention. Instead, the scope of the invention is defined by the appended claims and equivalents.

[0024] Systems and methods consistent with the present invention provide virus, worm, and unsolicited e-mail detection and/or prevention in e-mail servers. Placing these features in e-mail servers provides a number of new advantages, including the ability to align hash blocks to crucial boundaries found in e-mail messages and eliminate certain countermeasures by the attacker, such as using small Internet Protocol (IP) fragments to limit the detectable content in each packet. It also allows these features to relate e-mail header fields with the potentially-harmful segment of the message (usually an "attachment"), and decode common file-packing and encoding formats that might otherwise make a virus or worn undetectable by the packet-based technique (e.g., ".zip files").

[0025] By placing these features within an e-mail server, the ability to detect replicated content in the network at points where large quantities of traffic are present is obtained. By relating many otherwise-independent messages and finding common factors, the e-mail server may detect unknown, as well as known, viruses and worms. These features may also be applied to detect potential unsolicited commercial e-mail ("spam").

[0026] E-mail servers for major Internet Service Providers (ISPs) may process a million e-mail messages a day, or more, in a single server. When viruses and worms are active in the network, a substantial fraction of this e-mail may actually be traffic generated by the virus or worm. Thus, an e-mail server may have dozens to thousands of examples of a single e-mail-borne virus pass through it in a day, offering an excellent opportunity to determine the relationships between e-mail messages and detect replicated content (a feature that is indicative of virus/worm propagation) and spam, among other, more legitimate traffic (such as traffic from legitimate mailing lists).

[0027] Systems and methods consistent with the principles of the invention provide mechanisms to detect and stop e-mail-borne viruses and worms before the addressed user receives them, in an environment where the virus is still inert. Current e-mail servers do not normally execute any code in the e-mail being transported, so they are not usually subject to virus/worm infections from the content of the e-mails they process—though, they may be subject to infection via other forms of attack.

[0028] Besides e-mail-borne viruses and worms, another common problem found in e-mail is mass-e-mailing of unsolicited commercial e-mail, colloquially referred to as "spam." It is estimated that perhaps 25%-50% of all e-mail messages now received for delivery by major ISP e-mail servers is spam.

[0029] Users of network e-mail services are desirous of mechanisms to block e-mail containing viruses or worms from reaching their machines (where the virus or worm may easily do harm before the user realizes its presence). Users are also desirous of mechanisms to block unsolicited commercial e-mail that consumes their time and resources.

[0030] Many commercial e-mail services put a limit on each user's e-mail accumulating at the server, and not yet downloaded to the customer's machine. If too much e-mail arrives between times when the user reads his e-mail, additional e-mail is either "bounced" (i.e., returned to the sender's e-mail server) or even simply discarded, both of which events can seriously inconvenience the user. Because the user has no control over arriving e-mail due to e-mail-borne viruses/worms, or spam, it is a relatively common occurrence that the user's e-mail quota overflows due to unwanted and potentially harmful messages. Similarly, the authors of e-mail-borne viruses, as well as senders of spam, have no reason to limit the size of their messages. As a result, these messages are often much larger than legitimate e-mail messages, thereby increasing the risks of such denial of service to the user by overflowing the per-user e-mail quota.

[0031] Users are not the only group inconvenienced by spam and e-mail-borne viruses and worms. Because these types of unwanted e-mail can form a substantial fraction, even a majority, of e-mail traffic in the Internet, for extended periods of time, ISPs typically must add extra resources to handle a peak e-mail load that would otherwise be about half as large. This ratio of unwanted-to-legitimate e-mail traffic appears to be growing daily. Systems and methods consistent with the principles of the invention provide mechanisms to detect and discard unwanted e-mail in network e-mail servers.

Exemplary System Configuration

[0032] FIG. **1** is a diagram of an exemplary system **100** in which systems and methods consistent with the present invention may be implemented. System **100** includes mail clients **110** connected to a mail server **120** via a network **130**. Connections made in system **100** may be via wired, wireless, and/or optical communication paths. While FIG. **1** shows three mail clients **110** and a single mail server **120**, there can be more or fewer clients and servers in other implementations consistent with the principles of the invention.

[0033] Network **130** may facilitate communication between mail clients **110** and mail server **120**. Typically, network **130** may include a collection of network devices, such as routers or switches, that transfer data between mail clients **110** and mail server **120**. In an implementation consistent with the present invention, network **130** may take the form of a wide area network, a local area network, an intranet, the Internet, a public telephone network, a different type of network, or a combination of networks.

[0034] Mail clients **110** may include personal computers, laptops, personal digital assistants, or other types of wired or wireless devices that are capable of interacting with mail server **120** to receive e-mails. In another implementation, clients **110** may include software operating upon one of these devices. Client **110** may present e-mails to a user via a graphical user interface.

[0035] Mail server **120** may include a computer or another device that is capable of providing e-mail services for mail clients **110**. In another implementation, server **120** may include software operating upon one of these devices.

[0036] FIG. **2** is an exemplary diagram of mail server **120** according to an implementation consistent with the principles of the invention. Server **120** may include bus **210**, processor **220**, main memory **230**, read only memory (ROM) **240**, storage device **250**, input device **260**, output device **270**, and communication interface **280**. Bus **210** permits communication among the components of server **120**.

[0037] Processor **220** may include any type of conventional processor or microprocessor that interprets and executes instructions. Main memory **230** may include a random access memory (RAM) or another type of dynamic storage device that stores information and instructions for execution by processor **220**. ROM **240** may include a conventional ROM device or another type of static storage device that stores static information and instructions for use by processor **220**. Storage device **250** may include a magnetic and/or optical recording medium and its corresponding drive.

[0038] Input device **260** may include one or more conventional mechanisms that permit an operator to input information to server **120**, such as a keyboard, a mouse, a pen, voice recognition and/or biometric mechanisms, etc. Output device **270** may include one or more conventional mechanisms that output information to the operator, such as a display, a printer, a pair of speakers, etc. Communication interface **280** may include any transceiver-like mechanism that enables server **120** to communicate with other devices and/or systems. For example, communication interface **280** may include mechanisms for communicating with another device or system via a network, such as network **130**.

[0039] As will be described in detail below, server **120**, consistent with the present invention, provides e-mail services to clients **110**, while detecting unwanted e-mails and/or preventing unwanted e-mails from reaching clients **110**. Server **120** may perform these tasks in response to processor **220** executing sequences of instructions contained in, for example, memory **230**. These instructions may be read into memory **230** from another computer-readable medium, such as storage device **250** or a carrier wave, or from another device via communication interface **280**.

[0040] Execution of the sequences of instructions contained in memory **230** may cause processor **220** to perform processes that will be described later. Alternatively, hardwired circuitry may be used in place of or in combination with software instructions to implement processes consistent with the present invention. Thus, processes performed by server **120** are not limited to any specific combination of hardware circuitry and software.

[0041] FIG. **3** is an exemplary functional block diagram of mail server **120** according to an implementation consistent with the principles of the invention. Server **120** may include a Simple Mail Transfer Protocol (SMTP) block **310**, a Post Office Protocol (POP) block **320**, an Internet Message Access Protocol (IMAP) block **330**, and a hash processing block **340**.

[0042] SMTP block **310** may permit mail server **120** to communicate with other mail servers connected to network **130** or another network. SMTP is designed to efficiently and reliably transfer e-mail across networks. SMTP defines the interaction between mail servers to facilitate the transfer of e-mail even when the mail servers are implemented on different types of computers or running different operating systems.

[0043] POP block **320** may permit mail clients **110** to retrieve e-mail from mail server **120**. POP block **320** may be designed to always receive incoming e-mail. POP block **320** may then hold e-mail for mail clients **110** until mail clients **110** connect to download them.

[0044] IMAP block **330** may provide another mechanism by which mail clients **110** can retrieve e-mail from mail server

120. IMAP block **330** may permit mail clients **110** to access remote e-mail as if the e-mail was local to mail clients **110**.

[0045] Hash processing block **340** may interact with SMTP block **310**, POP block **320**, and/or IMAP block **330** to detect and prevent transmission of unwanted e-mail, such as e-mails containing viruses or worms and unsolicited commercial e-mail (span).

[0046] FIG. **4** is an exemplary diagram of hash processing block **340** according to an implementation consistent with the principles of the invention. Hash processing block **340** may include hash processor **410** and one or more hash memories **420**. Hash processor **410** may include a conventional processor, an application specific integrated circuit (ASIC), a field-programmable gate array (MPGA), or some other type of device that generates one or more representations for each received e-mail and records the e-mail representations in hash memory **420**.

[0047] An e-mail representation will likely not be a copy of the entire e-mail, but rather it may include a portion of the e-mail or some unique value representative of the e-mail. For example, a fixed width number may be computed across portions of the e-mail in a manner that allows the entire e-mail to be identified.

[0048] To further illustrate the use of representations, a 32-bit hash value, or digest, may be computed across portions of each e-mail. Then, the hash value may be stored in hash memory **420** or may be used as an index, or address, into hash memory **420**. Using the hash value, or an index derived therefrom, results in efficient use of hash memory **420** while still allowing the content of each e-mail passing through mail server **120** to be identified.

[0049] Systems and methods consistent with the present invention may use any storage scheme that records information about one or more portions of each e-mail in a space-efficient fashion, that can definitively determine if a portion of an e-mail has not been observed, and that can respond positively (i.e., in a predictable way) when a portion of an e-mail has been observed. Although systems and methods consistent with the present invention can use virtually any technique for deriving representations of portions of e-mails the remaining discussion will use hash values as exemplary representations of portions of e-mails received by mail server **120**.

[0050] In implementations consistent with the principles of the invention, hash processor **410** may hash one or more portions of a received e-mail to produce a hash value used to facilitate hash-based detection. For example, hash processor **410** may hash one or more of the main text within the message body, any attachments, and one or more header fields, such as sender-related fields (e.g., "From:," "Sender:," "Reply-To:," "Return-Path:," and "Error-To:"). Hash processor **410** may perform one or more hashes on each of the e-mail portions using the same or different hash functions.

[0051] As described in more detail below, hash processor **410** may use the hash results of the hash operation to recognize duplicate occurrences of e-mails and raise a warning if the duplicate e-mail occurrences arrive within a short period of time and raise their level of suspicion above some threshold. It may also be possible to use the hash results for tracing the path of an unwanted e-mail through the network.

[0052] Each hash value may be determined by taking an input block of data and processing it to obtain a numerical value that represents the given input data. Suitable hash functions are readily known in the art and will not be discussed in detail herein. Examples of hash functions include the Cyclic Redundancy Check (CRC) and Message Digest 5 (MD5). The resulting hash value, also referred to as a message digest or hash digest, may include a fixed length value. The hash value may serve as a signature for the data over which it was computed.

[0053] The hash value essentially acts as a fingerprint identifying the input block of data over which it was computed. Unlike fingerprints, however, there is a chance that two very different pieces of data will hash to the same value, resulting in a hash collision. An acceptable hash function should provide a good distribution of values over a variety of data inputs in order to prevent these collisions. Because collisions occur when different input blocks result in the same hash value, an ambiguity may arise when attempting to associate a result with a particular input.

[0054] Hash processor **410** may store a representation of each e-mail it observes in hash memory **420**. Hash processor **410** may store the actual hash values as the e-mail representations or it may use other techniques for minimizing storage requirements associated with retaining hash values and other information associated therewith. A technique for minimizing storage requirements may use one or more arrays or Bloom filters.

[0055] Rather than storing the actual hash value, which can typically be on the order of 32 bits or more in length, hash processor **410** may use the hash value as an index for addressing an array within hash memory **420**. In other words, when hash processor **410** generates a hash value for a portion of an e-mail, the hash value serves as the address location into the array. At the address corresponding to the hash value, a count value may be incremented at the respective storage location, thus, indicating that a particular hash value, and hence a particular e-mail portion, has been seen by hash processor **410**. In one implementation, the count value is associated with an 8-bit counter with a maximum value that sticks at **255**. While counter arrays are described by way of example, it will be appreciated by those skilled in the relevant art, that other storage techniques may be employed without departing from the spirit of the invention.

[0056] Hash memory **420** may store a suspicion count that is used to determine the overall suspiciousness of an e-mail message. For example, the count value (described above) may be compared to a threshold, and the suspicion count for the e-mail may be incremented if the threshold is exceeded. Hence, there may be a direct relationship between the count value and the suspicion count, and it may be possible for the two values to be the same. The larger the suspicion count, the more important the hit should be considered in determining the overall suspiciousness of the packet. Alternatively, the suspicion count can be combined in a "scoring function" with values from this or other hash blocks in the same message in order to determine whether the message should be considered suspicious.

[0057] It is not enough, however, for hash memory **420** to simply identify that an e-mail contains content that has been seen recently. There are many legitimate sources (e.g., e-mail list servers) that produce multiple copies of the same message, addressed to multiple recipients. Similarly, individual users often e-mail messages to a group of people and, thus, multiple copies might be seen if several recipients happen to receive their mail from the same server. Also, people often forward copies of received messages to friends or co-workers.

[0058] In addition, virus/worm authors typically try to minimize the replicated content in each copy of the virus/

worm, in order to not be detected by existing virus and worm detection technology that depends on detecting fixed sequences of bytes in a known virus or worm. These mutable viruses/worms are usually known as polymorphic, and the attacker's goal is to minimize the recognizability of the virus or worm by scrambling each copy in a different way. For the virus or worm to remain viable, however, a small part of it can be mutable in only a relatively small number of ways, because some of its code must be immediately-executable by the victim's computer, and that limits the mutation and obscurement possibilities for the critical initial code part.

[0059] In order to accomplish the proper classification of various types of legitimate and unwanted e-mail messages, multiple hash memories **420** can be employed, with separate hash memories **420** being used for specific sub-parts of a standard e-mail message. The outputs of different ones of hash memories **420** can then be combined in an overall "scoring" or classification function to determine whether the message is undesirable or legitimate, and possibly estimate the probability that it belongs to a particular class of traffic, such as a virus/worm message, spam, e-mail list message, normal user-to-user message.

[0060] For e-mail following the Internet mail standard RFC 822 (and its various extensions), hashing of certain individual e-mail header fields into field-specific hash memories **420** may be useful. Among the header fields for which this may be helpful are: (1) various sender-related fields, such as "From:", "Sender:", "Reply-To:". "Return-Path:" and "Error-To:"; (2) the "To:" field (often a fixed value for a mailing list, frequently missing or idiosyncratic in spam messages); and (3) the last few "Received:" headers (i.e., the earliest ones, since they are normally added at the top of the message), excluding any obvious timestamp data. It may also be useful to hash a combination of the "From:" field and the e-mail address of the recipient (transferred as part of the SMTP mail-transfer protocol, and not necessarily found in the message itself).

[0061] Any or all of hash memories **420** may be pre-loaded with knowledge of known good or bad traffic. For example, known viruses and spam content (e.g., the infamous "Craig Shergold letter" or many pyramid swindle letters) can be pre-hashed into the relevant hash memories **420**, and/or periodically refreshed in the memory as part of a periodic "cleaning" process described below. Also, known legitimate mailing lists, such as mailing lists from legitimate e-mail list servers, can be added to a "From:" hash memory **420** that passes traffic without further examination.

[0062] Over time, hash memories **420** may fill up and the possibility of overflowing an existing count value increases. The risk of overflowing a count value may be reduced if the counter arrays are periodically flushed to other storage media, such as a magnetic disk drive, optical media, solid state drive, or the like. Alternatively, the counter arrays may be slowly and incrementally erased. To facilitate this, a time-table may be established for flushing/erasing the counter arrays. If desired, the flushing/erasing cycle can be reduced by computing hash values only for a subset of the e-mails received by mail server **120**. While this approach reduces the flushing/erasing cycle, it increases the possibility that a target e-mail may be missed (i.e., a hash value is not computed over a portion of it).

[0063] Non-zero storage locations within hash memories **420** may be decremented periodically rather than being erased. This may ensure that the "random noise" from normal e-mail traffic would not remain in a counter array indefinitely. Replicated traffic (e.g., e-mails containing a virus/worm that are propagating repeatedly across the network), however, would normally cause the relevant storage locations to stay substantially above the "background noise" level.

[0064] One way to decrement the count values in the counter array fairly is to keep a total count, for each hash memory **420**, of every time one of the count values is incremented. After this total count reaches some threshold value (probably in the millions), for every time a count value is incremented in hash memory **420**, another count value gets decremented. One way to pick the count value to decrement is to keep a counter, as a decrement pointer, that simply iterates through the storage locations sequentially. Every time a decrement operation is performed, the following may done: (a) examine the candidate count value to be decremented and if non-zero, decrement it and increment the decrement pointer to the next storage location; and (b) if the candidate count value is zero, then examine each sequentially-following storage location until a non-zero count value is found, decrement that count value, and advance the decrement pointer to the following storage location.

[0065] It may be important to avoid decrementing any counters below zero, while not biasing decrements unfairly. Because it may be assumed that the hash is random, this technique should not favor any particular storage location, since it visits each of them before starting over. This technique may be superior to a timer-based decrement because it keeps a fixed total count population across all of the storage locations, representing the most recent history of traffic, and is not subject to changes in behavior as the volume of traffic varies over time.

[0066] A variation of this technique may include randomly selecting a count value to decrement, rather than processing them cyclically. In this variation, if the chosen count value is already zero, then another one could be picked randomly, or the count values in the storage locations following the initially-chosen one could be examined in series, until a non-zero count value is found.

Exemplary Processing for Unwanted E-Mail Detection/Prevention

[0067] FIGS. **5**A-**5**E are flowcharts of exemplary processing for detecting and/or preventing transmission of unwanted e-mail, such as an e-mail containing a virus or worm, including a polymorphic virus or worm, or an unsolicited commercial e-mail (span), according to an implementation consistent with the principles of the invention. The processing of FIGS. **5**A-**5**E will be described in terms of a series of acts that may be performed by mail server **120**. In implementations consistent with the principles of the invention, some of the acts may be optional and/or performed in an order different than that described. In other implementations, different acts may be substituted for described acts or added to the process.

[0068] Processing may begin when hash processor **410** (FIG. **4**) receives, or otherwise observes, an e-mail message (act **502**) (FIG. **5**A). Hash processor **410** may hash the main text of the message body, excluding any attachments (act **504**). When hashing the main text, hash processor **410** may perform one or more conventional hashes covering one or more portions, or all, of the main text. For example, hash processor **410** may perform hash functions on fixed or variable sized blocks of the main text. It may be beneficial for hash processor **410** to perform multiple hashes on each of the blocks using the same or different hash functions.

[0069] It may be desirable to pre-process the main text to remove attempts to fool pattern-matching mail filters. An example of this is HyperText Markup Language (HTML) e-mail, where spammers often insert random text strings in HTML comments between or within words of the text. Such e-mail may be referred to as "polymorphic spam" because it attempts to make each message appear unique. This method for evading detection might otherwise defeat the hash detection technique, or other string-matching techniques. Thus, removing all HTML comments from the message before hashing it may be desirable. It might also be useful to delete HTML tags from the message, or apply other specialized, but simple, pre-processing techniques to remove content not actually presented to the user. In general, this may be done in parallel with the hashing of the message text, since viruses and worms may be hidden in the non-visible content of the message text.

[0070] Hash processor **410** may also hash any attachments, after first attempting to expand them if they appear to be known types of compressed files (e.g., "zip" files) (act **506**). When hashing an attachment, hash processor **410** may perform one or more conventional hashes covering one or more portions, or all, of the attachment. For example, hash processor **410** may perform hash functions on fixed or variable sized blocks of the attachment. It may be beneficial for hash processor **410** to perform multiple hashes on each of the blocks using the same or different hash functions.

[0071] Hash processor **410** may compare the main text and attachment hashes with known viruses, worms, or spam content in a hash memory **420** that is pre-loaded with information from known viruses, worms, and spam content (acts **508** and **510**). If there are any hits in this hash memory **420**, there is a probability that the e-mail message contains a virus or worm or is spam. A known polymorphic virus may have only a small number of hashes that match in this hash memory **420**, out of the total number of hash blocks in the message. A non-polymorphic virus may have a very high fraction of the hash blocks hit in hash memory **420**. For this reason, storage locations within hash memory **420** that contain entries from polymorphic viruses or worms may be given more weight during the pre-loading process, such as by giving them a high initial suspicion count value.

[0072] A high fraction of hits in this hash memory **420** may cause the message to be marked as a probable known virus/worn or spam. In this case, the e-mail message can be sidetracked for remedial action, as described below.

[0073] A message with a significant "score" from polymorphic virus/worm hash value hits may or may not be a virus/worm instance, and may be sidetracked for further investigation, or marked as suspicious before forwarding to the recipient. An additional check may also be made to determine the level of suspicion.

[0074] For example, hash processor **410** may hash a concatenation of the From and To header fields of the e-mail message (act **512**) (FIG. **5**B). Hash processor **410** may then check the suspicion counts in hash memories **420** for the hashes of the main text, any attachments, and the concatenated From/To (act **514**). Hash processor **410** may determine whether the main text or attachment suspicion count is significantly higher than the From/To suspicion count (act **516**). If so, then the content is appearing much more frequently outside the messages between this set of users (which might otherwise be due to an e-mail exchange with repeated message quotations) and, thus, is much more suspicious.

[0075] When this occurs, hash processor **410** may take remedial action (act **518**). The remedial action taken might take different forms, which may be programmable or determined by an operator of mail server **120**. For example, hash processor **410** may discard the e-mail. This is not recommended for anything but virtually-certain virus/worm/spam identification, such as a perfect match to a known virus.

[0076] As an alternate technique, hash processor **410** may mark the e-mail with a warning in the message body, in an additional header, or other user-visible annotation, and allow the user to deal with it when it is downloaded. For data that appears to be from an unknown mailing list, a variant of this option is to request the user to send back a reply message to the server, classifying the suspect message as either span or a mailing list. In the latter case, the mailing list source address can be added to the "known legitimate mailing lists" hash memory **420**.

[0077] As another technique, hash processor **410** may subject the e-mail to mote sophisticated (and possibly more resource-consuming) detection algorithms to make a more certain determination. This is recommended for potential unknown viruses/worms or possible detection of a polymorphic virus/worn.

[0078] As yet another technique, hash processor **410** may hold the e-mail message in a special area and create a special e-mail message to notify the user of the held message (probably including From and Subject fields). Hash processor **410** may also give instructions on how to retrieve the message.

[0079] As a further technique, hash processor **410** may mark the e-mail message with its suspicion score result, but leave it queued for the user's retrieval. If the user's quota would overflow when a new message arrives, the score of the incoming message and the highest score of the queued messages are compared. If the highest queued message has a score above a settable threshold, and the new message's score is lower than the threshold, the queued message with the highest score may be deleted from the queue to make room for the new message. Otherwise, if the new message has a score above the threshold, it may be discarded or "bounced" (e.g., the sending e-mail server is told to hold the message and retry it later). Alternatively, if it is desired to never bounce incoming messages, mail server **120** may accept the incoming message into the user's queue and repeatedly delete messages with the highest suspicion score from the queue until the total is below the user's quota again.

[0080] As another technique, hash processor **410** may apply hash-based functions as the e-mail message starts arriving from the sending server and determine the message's suspicion score incrementally as the message is read in. If the message has a high-enough suspicion score (above a threshold) during the early part of the message, mail server **120** may reject the message, optionally with either a "retry later" or a "permanent refusal" result to the sending server (which one is used may be determined by settable thresholds applied to the total suspicion score, and possibly other factors, such as server load). This results in the unwanted e-mail using up less network bandwidth and receiving server resources, and penalizes servers sending unwanted mail, relative to those that do not.

[0081] If the suspicion count for the main text or any attachment is not significantly higher than the From/To suspicion count (act **516**), hash processor **410** may determine whether the main text or any attachment has significant replicated content (non-zero or high suspicion count values for many hash blocks in the text/attachment content in all storage locations of hash memories **420**) (act **520**) (FIG. **5**A). If not, the message is probably a normal user-to-user e-mail. These types of messages may be "passed" without further examination. When appropriate, hash processor **410** may also record the generated hash values by incrementing the suspicion count value in the corresponding storage locations in hash memory **420**.

[0082] If the message text is substantially replicated (e.g., greater than 90%), hash processor **410** may check one or more portions of the e-mail message against known legitimate mailing lists within hash memory **420** (act **522**) FIG. **5**C). For example, hash processor **410** may hash the From or Sender fields of the e-mail message and compare it/them to known legitimate mailing lists within hash memory **420**. Hash processor **410** may also determine whether the e-mail actually appears to originate from the correct source for the mailing list by examining, for example, the sequence of Received headers. Hash processor **410** may further examine a combination of the From or Sender fields and the recipient address to determine if the recipient has previously received e-mail from the sender. This is typical for mailing lists, but atypical of unwanted e-mail, which will normally not have access to the actual list of recipients for the mailing list. Failure of this examination may simply pass the message on, but mark it as "suspicious," since the recipient may simply be a new subscriber to the mailing list, or the mailings may be infrequent enough to not persist in the hash counters between mailings.

[0083] If there is a match with a legitimate mailing list (act **524**), then the message is probably a legitimate mailing list duplicate and may be passed with no further examination. This assumes that the mailing list server employs some kind of filtering to exclude unwanted e-mail (e.g., refusing to forward e-mail that does not originate with a known list recipient or refusing e-mail with attachments).

[0084] If there is no match with any legitimate mailing lists within hash memory **420**, hash processor **410** may hash the sender-related fields (e.g., From, Sender, Reply-To) (act **526**). Hash processor **410** may then determine the suspicion count for the sender-related hashes in hash memories **420** (act **528**).

[0085] Hash processor **410** may determine whether the suspicion counts for the sender-related hashes are similar to the suspicion count(s) for the main text hash(es) (act **530**) (FIG. **5**D). If both From and Sender fields are present, then the Sender field should match with roughly the same suspicion count value as the message body hash. The From field may or may not match. For a legitimate mailing list, it may be a legitimate mailing list that is not in the known legitimate mailing lists hash memory **420** (or in the case where there is no known legitimate mailing lists hash memory **420**). If only the From field is present, it should match about as well as the message text for a mailing list. If none of the sender-related fields match as well as the message text, the e-mail message may be considered moderately suspicious (probably spam, with a variable and fictitious From address or the like).

[0086] As an additional check, hash processor **410** may hash the concatenation of the sender-related field with the highest suspicion count value and the e-mail recipient's address (act **532**). Hash processor **410** may then check the suspicion count for the concatenation in a hash memory **420** used just for this check (act **534**). If it matches with a significant suspicion count value (act **536**) (FIG. **5**E), then the recipient has recently received multiple messages from this source, which makes it probable that it is a mailing list. The e-mail message may then be passed without further examination.

[0087] If the message text or attachments are mostly replicated (e.g., greater than 90% of the hash blocks), but with mostly low suspicion count values in hash memory **420** (act **538**), then the message is probably a case of a small-scale replication of a single message to multiple recipients. In this case, the e-mail message may then be passed without further examination.

[0088] If the message text or attachments contain some significant degree of content replication (say, greater than 50% of the hash blocks) and at least some of the hash values have high suspicion count values in hash memory **420** (act **540**), then the message is fairly likely to be a virus/worm or spam. A virus or worm should be considered more likely if the high-count matches are in an attachment. If the highly-replicated content is in the message text, then the message is more likely to be spam, though it is possible that e-mail text employing a scripting language (e.g., Java script) might also contain a virus.

[0089] If the replication is in the message text, and the suspicion count is substantially higher for the message text than for the From field, the message is likely to be spam (because spammers generally vary the From field to evade simpler spam filters). A similar check can be made for the concatenation of the From and To header fields, except that in this case, it is most suspicious if the From/To hash misses (finds a zero suspicion count), indicating that the sender does not ordinarily send e-mail to that recipient, making it unlikely to be a mailing list, and very likely to be a spammer (because they normally employ random or fictitious From addresses).

[0090] In the above cases, hash processor **410** may take remedial action (act **542**). The particular type of action taken by hash processor **410** may vary as described above.

CONCLUSION

[0091] Systems and methods consistent with the present invention provide mechanisms within an e-mail server to detect and/or prevent transmission of unwanted e-mail, such as e-mail containing viruses or worms, including polymorphic viruses and worms, and unsolicited commercial e-mail (spam).

[0092] Implementation of a hash-based detection mechanism in an e-mail server at the e-mail message level provides advantages over a packet-based implementation in a router or other network node device. For example, the entire e-mail message has been re-assembled, both at the packet level (i.e., IP fragment re-assembly) and at the application level (multiple packets into a complete e-mail message). Also, the hashing algorithm can be applied more intelligently to specific parts of the e-mail message (e.g., header fields, message body, and attachments). Attachments that have been compressed for transport (e.g., ".zip" files) can be expanded for inspection. Without doing this, a polymorphic virus could easily hide inside such files with no repeatable hash signature visible at the packet transport level.

[0093] With the entire message available for a single pass of the hashing process, packet boundaries and packet fragmentation do not split sequences of bytes that might otherwise provide useful hash signatures. A clever attacker might otherwise obscure a virus/worm attack by causing the IP packets carrying the malicious code to be fragmented into pieces smaller than that for which the hashing process is effective, or by forcing packet breaks in the middle of otherwise-visible fixed sequences of code in the virus/worm. Also, the entire message is likely to be longer than a single packet, thereby reducing the probability of false alarms (possibly due to insufficient hash-block sample size and too few hash blocks per packet) and increasing the probability of correct identification of a virus/worm (more hash blocks will match per message than per packet, since packets will be only pats of the entire message).

[0094] Also, fewer hash-block alignment issues arise when the hash blocks can be intelligently aligned with fields of the e-mail message, such as the start of the message body, or the start of an attachment block. This results in faster detection of duplicate contents than if the blocks are randomly aligned (as is the case when the method is applied to individual packets).

[0095] Email-borne malicious code, such as viruses and worms, also usually includes a text message designed to cause the user to read the message and/or perform some other action that will activate the malicious code. It is harder for such text to be polymorphic, because automatic scrambling of the user-visible text will either render it suspicious-looking, or will be very limited in variability. This fact, combined with the ability to start a hash block at the start of the message text by parsing the e-mail header, reduces the variability in hash signatures of the message, making it easier to detect with fewer examples seen.

[0096] Further, the ability to extract and hash specific headers from an e-mail message separately may be used to help classify the type of replicated content the message body carries. Because many legitimate cases of message replication exist (e.g., topical mailing lists, such as Yahoo Groups), intelligent parsing and hashing of the message headers is very useful to reduce the false alarm rate, and to increase the accuracy of detection of real viruses, worms, and spam.

[0097] This detection technique, compared to others which might extract and save fixed strings to be searched for in other pieces of e-mail, includes hash-based filters that are one-way functions (i.e., it is possible, given a piece of text, to determine if it has been seen before in another message). Given the state data contained in the filter, however, it is virtually impossible to reconstruct a prior message, or any piece of a prior message, that has been passed through the filter previously. Thus, this technique can maintain the privacy of e-mail, without retaining any information that can be attributed to a specific sender or receiver.

[0098] The foregoing description of preferred embodiments of the present invention provides illustration and description, but is not intended to be exhaustive or to limit the invention to the precise form disclosed, Modifications and variations are possible in light of the above teachings or may be acquired from practice of the invention.

[0099] For example, systems and methods have been described with regard to a mail server. In other implementations, the systems and methods described herein may be used within other devices, such as a mail client. In such a case, the mail client may periodically obtain suspicion count values for its hash memory from one or more network devices, such as a mail server.

[0100] It may be possible for multiple mail servers to work together to detect and prevent unwanted e-mails. For example, high-scoring entries from the hash memory of one mail server might be distributed to other mail servers, as long as the same hash functions are used by the same cooperating servers. This may accelerate the detection process, especially for mail servers that experience relatively low volumes of traffic.

[0101] Further, certain portions of the invention have been described as "blocks" that perform one or more functions. These blocks may include hardware, such as an ASIC or a FPGA, software, or a combination of hardware and software.

[0102] No element, act, or instruction used in the description of the present application should be construed as critical or essential to the invention unless explicitly described as such. Also, as used herein, the article "a" is intended to include one or more items. Where only one item is intended, the term "one" or similar language is used. The scope of the invention is defined by the claims and their equivalents.

What is claimed is:

1. A mail server, comprising:

one or more hash memories configured to store count values associated with a plurality of hash values; and

a hash processor configured to:

receive an e-mail message,

hash one or more portions of the e-mail message to generate hash values, as generated hash values,

increment the count values corresponding to the generated bash values, as incremented count values, and

determine whether the e-mail message is a potentially unwanted e-mail message based on the incremented count values.

2. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to perform a plurality of hashes on a plurality of variable-sized blocks of a main text of the e-mail message.

3. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to perform a plurality of hashes on a plurality of fixed-sized blocks of a main text of the e-mail message.

4. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to perform a plurality of hashes on a main text of the e-mail message using a plurality of different hash functions.

5. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to:

attempt to expand an attachment of the e-mail message, and

hash the attachment after attempting to expand the attachment.

6. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to perform a plurality of hashes on a plurality of variable-sized blocks of an attachment of the e-mail message.

7. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, die hash processor is configured to perform a plurality of hashes on a plurality of fixed-sized blocks of an attachment of the e-mail message.

8. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to perform a plurality of hashes on an attachment of the e-mail message using a plurality of different hash functions.

9. The server of claim **1**, wherein the hash processor is further configured to compare the generated hash values to hash values corresponding to known unwanted e-mails.

10. The server of claim **9**, wherein the known unwanted e-mails include at least one of e-mails containing a virus, e-mails containing a worm, and unsolicited commercial e-mails.

11. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to:

hash at least one of a main text and an attachment of the e-mail message to generate one or more first bash values, and

hash a concatenation of first and second header fields of the e-mail message to generate a second hash value.

12. The server of claim **11**, wherein the first and second header fields include a From header field and a To header field.

13. The server of claim **11**, wherein when determining whether the e-mail message is a potentially unwanted e-mail message, the hash processor is configured to identify the e-mail message as a potentially unwanted e-mail message when the count value corresponding to one or more first hash values is significantly higher than the count value corresponding to the second hash value.

14. The server of claim **1**, wherein the hash processor is further configured to take remedial action when the e-mail message is a potentially unwanted e-mail message, when taking remedial action, the hash processor is configured to at least one of.

discard the e-mail message,

bounce the e-mail message,

mark the e-mail message with a warning,

subject the e-mail message to a virus or worm detection process,

create a notification message, and

generate a suspicion score for the e-mail message and use the suspicion score to identify further processing for the e-mail message.

15. The server of claim **1**, wherein the hash processor is further configured to:

generate a suspicion score for the e-mail message based on the incremented count values,

determine whether a newly received e-mail message exceeds a mail quota,

identify an earlier-received e-mail message with a highest suspicion score,

determine whether a suspicion score of the newly received e-mail message is lower than the suspicion score of the earlier-received e-mail message when the newly received e-mail message exceeds the mail quota,

delete the earlier-received e-mail message when the suspicion score of the newly received e-mail message is lower than the suspicion score of the earlier-received e-mail message, and

store the newly received e-mail message.

16. The server of claim **1**, wherein the hash processor is configured to hash the one or more portions of the e-mail message and increment the count values incrementally as the e-mail message is being received.

17. The server of claim **16**, wherein the hash processor is further configured to:

generate a suspicion score for the e-mail message based on the incremented count values,

reject the e-mail message when the suspicion score of the e-mail message is above a threshold.

18. The server of claim **17**, wherein the rejecting occurs before the e-mail message is completely received.

19. The server of claim **1**, wherein the hash processor is further configured to:

compare the generated hash values to known legitimate mailing lists, and

pass the e-mail message without further examination when the generated hash values match one of the known legitimate mailing lists.

20. The server of claim **19**, wherein the hash processor is configured to:

determine whether the e-mail message originated from one of the known legitimate mailing lists.

21. The server of claim **1**, wherein the hash processor is configured to:

hash a main text of the e-mail message to generate a first hash value, and

hash sender-related header fields of the e-mail message to generate one or more second hash values.

22. The server of claim **21**, wherein the sender-related header fields include at least one of a From header field, a Sender header field, and a Reply-To header field.

23. The server of claim **21**, wherein when determining whether the e-mail message is a potentially unwanted e-mail message, the hash processor is configured to identify the e-mail message as a potentially unwanted e-mail message when the count value corresponding to the first hash value is higher than the count values corresponding to the one or more second hash values.

24. The server of claim **1**, wherein when hashing one or more portions of the e-mail message, the hash processor is configured to:

perform a plurality of hashes on a main text of the e-mail message to generate main text hashes, and

hash at least one header field of the e-mail message to generate at least one header hash.

25. The server of claim **24**, when determining whether the e-mail message is a potentially unwanted e-mail message, the hash processor is configured to:

generate a score for the main text based on count values corresponding to the main text hashes and a score for the at least one header field based on the count value corresponding to the at least one header hash, and

identify the e-mail message as a potentially unwanted e-mail message when the score for the main text is substantially higher than the score for the at least one header hash.

26. A mail server, comprising:

one or more hash memories configured to store count values associated with a plurality of hash values; and

a hash processor configured to:

receive e-mail messages,

hash one or more portions of the received e-mail messages to generate hash values, as generated hash values,

increment the count values corresponding to the generated hash values, as incremented count values, and

generate suspicion scores for the received e-mail messages based on the incremented count values.

27. The server of claim **26**, wherein the hash processor is further configured to:

maintain a counter corresponding to each of the one or more hash memories, and

decrement ones of the count values based on the counter.

28. The server of claim **27**, wherein the hash processor is configured to:

determine when a value of the counter reaches a threshold, and

decrement one of the count values each time another one of the count values is incremented after the value of the counter reaches the threshold.

29. The server of claim **28**, wherein the hash processor is further configured to:

identify a count value to decrement,

determine whether the identified count value is non-zero, and

decrement the identified count value when the identified count value is non-zero.

30. The server of claim **29**, wherein the hash processor is further configured to:

examine next sequential ones of the count values until a non-zero count value is found when the identified count value is zero, and

decrement the non-zero count value.

* * * * *