(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 September 2007 (07.09.2007)

PCT

(10) International Publication Number
WO 2007/100915 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/US2007/005406

(22) International Filing Date:
28 February 2007 (28.02.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/778,008 28 February 2006 (28.02.2006) US
60/790,626 10 April 2006 (10.04.2006) US

(71) Applicant *(for all designated States except US)*: **THE TRUSTEES OF COLUMBIA UNIVERSITY IN THE CITY OF NEW YORK** [US/US]; 412 Low Memorial Library, 535 West 116th Street, New York, NY 10027 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **STOLFO, Salvatore, J.** [US/US]; 80 Kenilworth Road, Ridgewood, NJ 07450 (US). **WANG, Ke** [CN/US]; 435 W. 119th Street, Apt 2E, New York, NY 10027 (US). **PAREKH, Janak** [US/US]; 110 Bayview Road, Manhasset, NY 11030 (US).

(74) Agents: **BYRNE, Matthew, T.** et al.; Wilmer Cutler Pickering Hale and Dorr LLP, 399 Park Avenue, New York, NY 10022 (US).

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

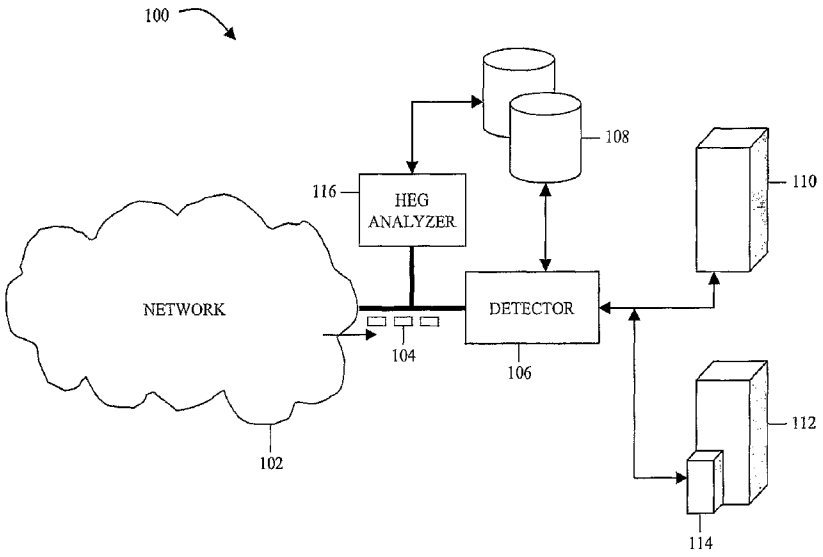
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SYSTEMS, METHODS, AND MEDIA FOR OUTPUTTING DATA BASED ON ANOMALY DETECTION



(57) Abstract: Systems, methods, and media for outputting data based on anomaly detection are provided. In some embodiments, methods for outputting data based on anomaly detection include: receiving a known-good dataset; storing distinct n-grams from the known-good dataset to form a binary anomaly detection model; receiving known-good new n-grams; computing a rate of receipt of distinct n-grams in the new n-grams; determining whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams; using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

WO 2007/100915 A2

SYSTEMS, METHODS, AND MEDIA FOR OUTPUTTING DATA BASED ON ANOMALY DETECTION

Statement Regarding Federally Sponsored Research or Development

[0001] The government may have certain rights in the present invention pursuant to grants by the Army Research Office, Grant No. ARO W911NF-04-1-0442.

Cross-Reference to Related Applications

[0002] This application claims the benefit of United States Provisional Patent Applications No. 60/778,008, filed February 28, 2006, and United States Provisional Patent Application No. 60/790,626, filed April 10, 2006, each of which is hereby incorporated by reference herein in its entirety.

Technical Field

[0003] The disclosed subject matter relates to systems, methods, and media for outputting data based on anomaly detection.

Background

[0004] Content anomaly detectors have been developed to identify anomalous data in an otherwise seemingly normal stream of data. Anomalous data can include instances of malicious code such as worms, viruses, Trojans, etc. In some of these detectors, an n-gram is looked at by the detector to determine if it is anomalous. An n-gram is a set of n units of data. For example, a 1-gram may be a single byte of data, and a 2-gram may be two bytes of data.

[0005] A content anomaly detection model based on 1-gram frequency distribution of datasets is effective at capturing attacks that display abnormal byte distributions, but it is

vulnerable to attacks crafted to resemble normal byte distributions. A content anomaly detection model based on higher order n-grams frequency distribution of datasets can address this shortcoming. However, as the order of the n-grams increases, memory usage increases exponentially. This is because the maximum possible number of distinct n-grams increases exponentially as the order of the n-grams increases. For example, the maximum possible number of distinct 5-grams is 256^5, or 1024 billion.

[0006] As new defensive (e.g., anomaly detection, etc.) techniques are developed to counter fast-spreading network threats, attackers have become more sophisticated as well. A model based on a mixture of high order n-grams frequency distribution can address threats posed by such sophisticated attackers, but only at the expense of heavy memory and computational overhead. For example, even for a small mixture of n-grams of modest orders, such as a mixture of 2-grams, 3-grams, and 4-grams, the total memory capacity may be impracticable.

Summary

[0007] Systems, methods, and media for outputting data based on anomaly detection are provided. In some embodiments, methods for outputting data based on anomaly detection include: receiving a known-good dataset; storing distinct n-grams from the known-good dataset to form a binary anomaly detection model; receiving known-good new n-grams; computing a rate of receipt of distinct n-grams in the new n-grams; determining whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams; using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0008] In some embodiments, methods for outputting data based on anomaly detection include: receiving known anomaly signatures; generating n-grams of different sizes using the known anomaly signatures; storing abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model; using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0009] In some embodiments, methods for outputting data based on anomaly detection include: receiving a shared binary anomaly detection model; comparing the shared binary anomaly detection model with a local anomaly detection model; combining the shared binary anomaly detection model with the local anomaly detection model to form a new binary anomaly detection model; using the model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0010] In some embodiments, methods for outputting data based on anomaly detection include: receiving an input dataset; generating n-grams of different sizes from the input dataset; counting the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model; computing an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset; using the anomaly score to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0011] In some embodiments, methods for outputting data based on anomaly detection include: receive an input dataset; using a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly; if the input dataset is determined to be likely to contain an anomaly, dropping the input dataset; and if the input

dataset is determined to be unlikely to contain an anomaly, outputting the input dataset based on whether the input dataset contains an anomaly.

[0012] In some embodiments, computer-readable media containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, are provided. This method includes: receiving a known-good dataset; storing distinct n-grams from the known-good dataset to form a binary anomaly detection model; receiving known-good new n-grams; computing a rate of receipt of distinct n-grams in the new n-grams; determining whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams; using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0013] In some embodiments, computer-readable media containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, are provided. This method includes: receiving known anomaly signatures; generating n-grams of different sizes using the known anomaly signatures; storing abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model; using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0014] In some embodiments, computer-readable media containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, are provided. This method includes: receiving a shared binary anomaly detection model; comparing the shared binary anomaly detection model with a local anomaly detection model; combining the shared binary anomaly

detection model with the local anomaly detection model to form a new binary anomaly detection model; using the model to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0015] In some embodiments, computer-readable media containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, are provided. This method includes: receiving an input dataset; generating n-grams of different sizes from the input dataset; counting the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model; computing an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset; using the anomaly score to determine whether an input dataset contains an anomaly; and outputting the input dataset based on whether the input dataset contains an anomaly.

[0016] In some embodiments, computer-readable media containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, are provided. This method includes: receive an input dataset; using a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly; if the input dataset is determined to be likely to contain an anomaly, dropping the input dataset; and if the input dataset is determined to be unlikely to contain an anomaly, outputting the input dataset based on whether the input dataset contains an anomaly.

[0017] In some embodiments, systems for outputting data based on anomaly detection are provided. The systems include a digital processing device that: receives a known-good dataset; stores distinct n-grams from the known-good dataset to form a binary anomaly detection model; receives known-good new n-grams; computes a rate of receipt of distinct n-grams in the new n-grams; determines whether further training of the anomaly detection

model is necessary based on the rate of receipt on distinct n-grams; uses the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputs the input dataset based on whether the input dataset contains an anomaly.

[0018] In some embodiments, systems for outputting data based on anomaly detection are provided. The systems include a digital processing device that: receives known anomaly signatures; generates n-grams of different sizes using the known anomaly signatures; stores abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model; uses the binary anomaly detection model to determine whether an input dataset contains an anomaly; and outputs the input dataset based on whether the input dataset contains an anomaly.

[0019] In some embodiments, systems for outputting data based on anomaly detection are provided. The systems include a digital processing device that: receives a shared binary anomaly detection model; compares the shared binary anomaly detection model with a local anomaly detection model; combines the shared binary anomaly detection model with the local anomaly detection model to form a new binary anomaly detection model; uses the model to determine whether an input dataset contains an anomaly; and outputs the input dataset based on whether the input dataset contains an anomaly.

[0020] In some embodiments, systems for outputting data based on anomaly detection are provided. The systems include a digital processing device that: receives an input dataset; generates n-grams of different sizes from the input dataset; counts the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model; computes an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset; uses the anomaly score to determine whether an input dataset contains an anomaly; and outputs the input dataset based on whether the input dataset contains an anomaly.

[0021] In some embodiments, systems for outputting data based on anomaly detection are provided. The systems include a digital processing device that: receives an input dataset; uses a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly; if the input dataset is determined to be likely to contain an anomaly, drops the input dataset; and if the input dataset is determined to be unlikely to contain an anomaly, outputs the input dataset based on whether the input dataset contains an anomaly.

Brief Description of the Drawings

[0022] FIG. 1 is a schematic diagram of a system for generating, training, and sharing a binary-based content anomaly model and for using the content anomaly model to detect content anomalies in accordance with some embodiments of the disclosed subject matter.

[0023] FIG. 2 is a simple illustration of a method for generating, training, and sharing a binary-based content anomaly detection model and for using the content anomaly model to detect content anomalies in accordance with some embodiments of the disclosed subject matter.

[0024] FIG. 3 is a simple illustration of a method for generating and training a binary-based content anomaly detection model using known-good training datasets in accordance with some embodiments of the disclosed subject matter.

[0025] FIG. 4 is a simple illustration of a method for generating and training a binary-based content anomaly detection model using known anomaly signatures in accordance with some embodiments of the disclosed subject matter.

[0026] FIG. 5 is a simple illustration of a method for sharing binary-based content anomaly detection models in accordance with some embodiments of the disclosed subject matter.

[0027] FIG. 6 is a simple illustration of a method for using binary-based content anomaly detection models to detect content anomalies in accordance with some embodiments of the disclosed subject matter.

[0028] FIG. 7 is a simple illustration of a method for training a content anomaly detection model and using the model to detect content anomalies in accordance with some embodiments of the disclosed subject matter.

Detailed Description

[0029] Systems, methods, and media for outputting data based on anomaly detection are provided. In some embodiments of the disclosed subject matter, systems, methods, and media are provided for generating and/or training binary-based content anomaly detection models. The presence and/or absence of each of the distinct n-grams in a training dataset can be used to generate the detection models. For instance, a detection model can be generated using a set of n-grams in a training dataset that are observed during a training phase of the model. The model can be referenced during a detecting or testing phase to detect the payload of a data packet containing one or more never-before-seen n-grams.

[0030] In some embodiments of the disclosed subject matter, systems, methods, and media are provided for generating and updating an anomaly signature model. A binary-based content anomaly detection model has advantages in speed and memory efficiency, but it can be sensitive to noisy training datasets (i.e., training datasets that are not totally free of malicious code). An anomaly signature model containing a collection of signatures of known malicious code can be used to compensate for the risk of using corrupted training datasets that are associated with binary-based content anomaly detection models. For example, the signature content of samples of known malicious code can be used to build and update an

anomaly signature model that can be used as a reference. For instance, such an anomaly signature model can be used to filter out malicious code from training datasets.

[0031] In some embodiments of the disclosed subject matter, systems, methods, and media are provided for sharing binary-based content anomaly detection models and anomaly signature models. A group of protected sites facing similar network security threats can share their models for enhancing the common defense against such threats. For example, the binary-based content anomaly detection model of each site in the group can be shared regularly to identify suspicious content commonly detected by multiple sites.

[0032] In some embodiments of the disclosed subject matter, systems, methods, and media are provided for creating and using a feedback loop between a binary-based content anomaly detector and a host-based detector. Interactions between a host-based detector and a binary-based detector can be developed over time. For example, a host-based detector can further examine a dataset suspected by a binary-based detector of containing malicious code and either confirm or correct the suspicion, thereby reducing the false-positive rate of the detector. The binary-based detector, in turn, can reduce the volume of the network traffic directed to the host-based detector, thereby reducing the overhead associated with running the host-based detector.

[0033] FIG. 1 is a schematic diagram of a system 100 for detecting content anomalies in accordance with some embodiments. As illustrated, system 100 can include a network 102, data traffic 104, a detector 106, a data structure 108, a production server 110, a shadow server 112, a host-based detector 114, and a high-entropy-gram (HEG) analyzer 116. In some embodiments, detector 106, data structure 108, production server 110, shadow server 112, host-based detector 114, and HEG analyzer 116 can be implemented in a single device or a combination of devices. These device(s) can includes various suitable mechanisms for performing the functions associated with detector 106, data structure 108, production server

110, shadow server 112, host-based detector 114, and HEG analyzer 116. For example, such mechanisms can include a processor, digital processing device, memory, communications interfaces, displays, etc., such a general purpose computer, a special purpose computer, a server, a mobile phone, a personal data assistant, an email device, and/or various other suitable devices.

[0034] Network 102 can be a local area network (LAN), a wide area network (WAN), a wireless network, the Internet, a cable television network, a telephone network, and/or various other suitable networks from which malicious attacks can be launched.

[0035] Data traffic 104 can include one or more network data packets, data frames, one or more files that contain various types of data, such as text, graphic images, sound samples, video samples, and computer-executable codes, a stream of data in bytes or a stream of various other suitable symbols or tokens in one or more communication sessions, and/or various other forms of data in suitable formats.

[0036] In some embodiments, n-grams can be generated in detector 106 by sliding windows of arbitrary lengths over data traffic 104. Detector 106 can train a content anomaly detection model by storing the distinct n-grams observed during a training phase in data structure 108.

[0037] During a detection phase, detector 106 can score data traffic 104 on the basis of the number of never-before-seen n-grams contained in data traffic 104. The score can also be weighted by the number of malicious n-grams contained in data traffic 104. Detector 106 can capture the order dependence of byte sequences in data traffic 104 by modeling higher order n-grams. This can enable detector 106 to capture more sophisticated attacks.

[0038] Data structure 108 can be a data structure that allows for the modeling of a mixture of different sizes of n-grams. Data structure 108 may be implemented in random

access memory (RAM), flash memory, a disk drive, optical media, and/or various other suitable storage technologies. In some embodiments, Bloom filters are used.

[0039] A Bloom filter can be defined as a bit array of m bits, where any individual bit i is set if the hash of an input value (i.e., input value mod m) is i. A Bloom filter can act as a one-way data structure that can contain many items. An advantage of using a Bloom filter is that operations on a Bloom filter takes a constant amount of time regardless of the size of the Bloom filter, keeping computational overhead low. In some embodiments, the H3 hash function or SHA-1 hash function may be used in connection with a Bloom filter, although other hash functions may additionally or alternatively be used.

[0040] A Bloom filter may contain false positives if a collision occurs while performing a check operation. Collisions may occur because two distinct inputs into a hash function may produce identical outputs. Suppose, for example, that there are two distinct n-grams, A (which occurs in the training dataset only) and B (which occurs in the input dataset only), which both produce the same hash value. Because A is in the training dataset, the Bloom filter contains a bit set for A. If the Bloom filter is checked for B, however, because B hashes to the same value, then B can be mistakenly believed to be represented in the Bloom filter. This is a false positive.

[0041] Production server 110 and shadow server 112 can be used to run application programs that ultimately use data traffic 104. In some embodiments, detector 106 directs data traffic 104 to production server 110 when detector 106 determines that it is unlikely that data traffic 104 contains malicious code. In some embodiments, shadow server 112 and production server 110 can be configured to have the same software programs running, except that shadow server 112 can be operating in a protected environment using an emulator, virtual machine, sandbox or other suitable mechanism for protecting server 112 from potentially malicious code. In some embodiments, server 112 includes host-based detector

114 which can additionally or alternatively provide a protected environment using an emulator, virtual machine, sandbox or other suitable mechanism for protecting server 112 and detector 114 from potentially malicious code. Server 112 and/or host-based detector 114 can include one or more host-based fault detectors and patch generation techniques, such as StackGuard/MemGuard, Selective Transactional Emulator (STEM), and Dynamic Buffer Overflow Containment (DYBOC), and anti-virus scanners that can collect and maintain content anomaly signatures of malicious code, such as stealthy worms, etc. StackGuard and MemGuard can be obtained from "http://www.freebsd.com." STEM is discussed in detail in Building a Reactive Immune System for Software Services by Stelios Sidiroglou-Douskos, et al. DYBOC is discussed in detail in A Dynamic Mechanism for Recovering from Buffer Overflow Attacks by Stelios Sidiroglou-Douskos, et al.

[0042] In some instances, data traffic 104 that is directed to shadow server 112 may be falsely identified as containing an anomaly. Using host based detector 114, shadow server 112 may validate the data as not containing an instance of malicious code. Detector 106 and shadow server 112 can interact so that false positives that have been validated by shadow server 112 serve as training data to improve the accuracy of the content anomaly detection model of detector 106. Through this process, the false positive rate of detector 106 can decrease. This, in turn, can result in workload reduction for shadow server 112.

[0043] In some embodiments, shadow server 112 acts as a training supervisor, wherein server requests are sent to shadow server 112 and only those requests that generate a normal response are sent to detector 106 for training the content anomaly detection model of detector 106.

[0044] In some embodiments, detector 106 is deployed with no trained model and, instead it initially deems 100 % of data traffic 104 as containing malicious requests. Shadow

server 112 can then provide false-positive feedback and relevant training data to detector 106 for incremental training.

[0045] High-entropy-gram (HEG) analyzer 116 can be used to detect HEG grams and analyze scarcity of commonly occurring HEGs and HEG distributions of normal and malicious data. An HEG is a type of gram that has a high level (e.g., 98-100%) of its contents as distinct byte values. Using HEG grams can help reduce the amount of data that a Bloom filter must contain. This can be accomplished by making Bloom filters to contain non-HEG grams, such as data between commonly occurring HEGs.

[0046] HEG analyzer 116 can be a separate device running one or more analyzer programs. It can also be one or more programs run by detector 106.

[0047] FIG. 2 is a simple illustration of a method for generating, training, and sharing a binary-based content anomaly model and for using the content anomaly model to detect content anomalies in accordance with some embodiments. As shown at 202, a binary-based content anomaly detection model is generated and trained. In some embodiments, a binary-based content anomaly detection model is generated and trained using known good training datasets, for example, as described in connection with FIG. 3. In some embodiments, a binary-based content anomaly detection model is generated and trained using known anomaly signatures, for example, as described in connection with FIG. 4. In some embodiments, a binary-based content anomaly detection model is trained in cooperation with a host-based detector, for example, as described in connection with part 703 of FIG. 7.

[0048] At 204, the binary-based content anomaly detection model is shared. In some embodiments, a binary-based content anomaly detection model is received from one or more remote sites to compare with and update the local model, for example, as described in connection with FIG. 5.

[0049] At 206, the binary-based content anomaly detection model is used to detect content anomalies. In some embodiments, an anomaly score of an input dataset is used to determine whether the input dataset is anomalous, for example, as described in connection with FIG. 6. In some embodiments, a binary-based content anomaly detection model is used to compute the likelihood of an input dataset containing malicious code and classify the input dataset based on the computed likelihood, for example, as described in connection with part 701 of FIG. 7.

[0050] FIG. 3 is a simple illustration of a method 300 for training a content anomaly detection model in accordance with some embodiments. As shown, at 302, a training dataset is received. In some embodiments, the training dataset can include one or more network data packets or data frames. In some embodiments, the training dataset can include one or more files that contain various types of data, such as text, graphic images, sound samples, video samples, computer-executable codes, various other suitable types of data, and/or one or more combinations thereof. In some embodiments, the training dataset can also include a stream of data in bytes, a stream of tokens, and a stream of various other suitable symbols or units in one or more communication sessions.

[0051] In some embodiments, the training dataset can be received from another site through a network, such as network 102. In some embodiments, it can be received from data structure 108, production server 110, shadow server 112, host-based detector 114, HEG analyzer 116, various other suitable sources, and one or more combinations thereof.

[0052] In some embodiments, the training dataset can be checked to ascertain its validity before it is used for training content anomaly detection models to ward off potential training attacks. Such a validation effort can also help avoid inadvertently immunizing one or more instances of malicious code from being detected. For example, a training dataset can

be processed to determine whether it harbors any data that produces erroneous or otherwise unusual or invalid outcomes.

[0053] At 304, distinct n-grams of different sizes contained in the training dataset can be stored in a data structure, such as data structure 108, to form a model. In some embodiments, n-grams of different sizes can be generated by sliding windows of corresponding sizes over the training dataset and the distinct n-grams of different sizes contained in the training dataset can be stored when they are observed for the first time.

[0054] In some embodiments, hashed n-grams formed when the Bloom filter stores an n-gram can be cached to speed up the Bloom filter's check operations being used as part of a detection process, as described below in connection with data structure 108 in FIG. 1. This is advantageous because hash values can be looked up instead of being computed.

[0055] In some embodiments, a class of universal hash functions, such as H_3, can be used to reduce computational overhead associated with inserting and/or checking n-grams over a large amount of data. Given such universal hash functions, for example, re-computing hashes can be avoided when sliding n-grams windows and/or when using different window sizes. Suppose, for instance, that the hash of a 7-gram is needed and a 5-gram is hashed. The universal hash functions can hash an incremental 2-gram and combine it with the 5-gram hash value to generate a 7-gram hash.

[0056] In some embodiments, a detector, such as detector 106, can be used to generate the n-grams of different sizes and store the distinct n-grams. In some embodiments, an HEG analyzer 116 can be used to generate the n-grams of different sizes and store the distinct n-grams.

[0057] At 306, known-good new n-grams are received and a rate at which a new distinct n-gram is observed is computed. In some embodiments, detector 106 can be used to

compute the new n-gram rate. In some embodiments, HEG analyzer 116 can be used to compute the new n-gram rate.

[0058] In some embodiments, new distinct n-grams that are observed over a time period can be used to compute the new n-gram rate. For example, the number of new distinct n-grams counted every 100 hours (or other period of time) can be used to compute the new n-gram rate. In some embodiments, new distinct n-grams that are observed from a number of data packets can be counted to compute the new n-gram rate. For example, a number of new distinct n-grams counted from every 1,000 data packets (or other number of data packets) can be used to compute the new n-gram rate.

[0059] At 308, it is determined whether further training of the content anomaly detection model is warranted using the new n-gram rate computed at 306. During the initial training period, it can be expected that many new distinct n-grams are observed. Over time, however, fewer distinct never-before-seen n-grams may be observed.

[0060] In some embodiments, a content anomaly detection model is deemed to have been sufficiently trained when the new n-gram rate becomes stable and low. If, for example, three consecutive new n-gram rates computed every 10,000 data packets are very close in value, a content anomaly detection model can be said to have been sufficiently trained in some embodiments. In some embodiments, a content anomaly detection model can also be said to have been sufficiently trained if four consecutive new n-gram rates computed every 30 days are very close in value. Various other metrics can be used additionally or alternatively to determine when the model is sufficiently trained.

[0061] If it is determined at 308 that further training is necessary, 306 and 308 can be repeated. If, however, it is determined at 310 that no further training is necessary, the content anomaly detection model can be deployed for detecting content anomalies of data traffic, such as data traffic 104.

[0062] FIG. 4 is a simple illustration of another method for generating and training a content anomaly detection model in accordance with some embodiments. As shown, at 402, known anomaly signatures are received. In some embodiments, the signature content of Snort rules from Sourcefire® and a collection of known virus samples are received. For example, such signatures can be purchased and/or downloaded from a trust-worthy web site, such as the one maintained by Sourcefire®.

[0063] In some embodiments, the anomaly signatures are stored in data structure 108. In some embodiments, it can be stored at detector 106, or HEG analyzer 116.

[0064] At 404, n-grams of different sizes are generated from the anomaly signatures. In some embodiments, as in 304 of FIG. 3 or 606 of FIG. 6, n-grams of different sizes can be generated by sliding windows of corresponding sizes over the content anomaly signatures.

[0065] At 406, the n-grams of different sizes generated in 404 are filtered to remove normal n-grams using a known-clean dataset. This may be necessary because the anomaly signatures may still have some normal n-grams. For example, an attack disguising as an HTTP request can still contain normal keywords, such as GET. In some embodiments, a Bloom filter containing known-clean datasets can be compared with the input dataset to identify normal n-grams. In some embodiments, the filtering operation is performed by detector 106.

[0066] At 408, distinct n-grams from the abnormal n-grams of different sizes are stored. In some embodiments, a Bloom filter can be used to store the distinct n-grams of different sizes. Instead of using n bytes to represent an n-gram, for example, a Bloom filter can store an n-gram using just few bits. For instance, a 24-bit Bloom filter is capable of holding $2^{24}/N_h$ elements, where N_h represents the number of hash functions used.

[0067] At 410, it is determined whether the content anomaly detection model needs to be updated. In some embodiments, the content anomaly model can be updated incrementally

following one or more releases of new anomaly signatures due to identifications of new viruses and/or an update of the Snort rules. If it is determined that the content anomaly model needs to be updated, 402, 404, 406, and 408 can be repeated.

[0068] In some embodiments, training datasets are scrutinized using the content anomaly detection model. For example, n-grams in the training datasets matching the content anomaly detection model can be dropped. In some embodiments, an entire data packet can be dropped if the packet contains too many n-grams that match the content anomaly detection model. In some embodiments, a 5% (or other per cent threshold) bad n-gram threshold is used to determine whether to drop an entire packet out of the training datasets.

[0069] In some embodiments, if a never-before-seen n-gram with respect to both the good and bad content anomaly detection model appear, its detection score is further weighted by a factor of 5 (or other factors) over other malicious n-grams. This enables further separation of malicious packets from normal ones in order to achieve higher detection accuracy.

[0070] FIG. 5 is a simple illustration of a method for sharing content anomaly detection models in accordance with some embodiments. As shown, at 502, a trained binary-based detection model is shared. The models may be distributed from one detector to another using email, ftp, http, and/or various other suitable mechanisms.

[0071] In some embodiments, Bloom filters are used to provide mechanism for sharing n-grams generated from potentially malicious code among multiple sites. Bloom filters are capable of preserving privacy of traffic content of each site because Bloom filters share signature information with little risk of revealing content information. For example, a Bloom filter can confirm the presence/absence of an n-gram with little risk of disclosing the original n-gram.

[0072] In some embodiments, a single Bloom filter contains n-grams associated with more than one type of potentially malicious code to reduce memory overhead further than if separate Bloom filters were used. This also enables a reduction in the computational overhead. For example, by holding multiple types of potentially malicious code in a single Bloom filter, the common n-grams that are shared between different types of potentially malicious code are stored only once, reducing memory overhead when compared to storing duplicate copies of identical n-grams.

[0073] In some embodiments, a Bloom filter is compressed before it is transmitted for efficient transmission. For example, Bloom filters can be compressed using LZW compression algorithm before they are transmitted to remote sites.

[0074] At 504, the content of a local model is compared with the content of a shared model from a remote site. In some embodiments, the contents of the local model and the shared model are compared by using a bitwise AND operation. In some embodiments, a similarity score between the local model and each of the shared models is computed. In some embodiments, the similarity score is computed using the following formula:

$$\text{Score} = 2 * \frac{N_c}{(N_1 + N_2)},$$

where N_c is the number of common n-grams and N_i the number of suspicious n-grams in alert i. If a Bloom filter is used, a count of items in the filter is be kept in some embodiments. In some embodiments, the count is estimated by N_b / N_h, where N_b is the number of bits set in the filter and N_h is the number of hash function used by the filter.

[0075] A higher score implies that the local model and a shared model have many common n-grams. In some embodiments, the more commonly observed n-grams are given more weight for determining the likelihood of being a part of an instance of malicious code.

[0076] At 506, the local model is updated using each of the shared models. In one embodiments, one or more shared models from remote sites are merged into the local model. For example, the content of the shared model from the remote site can be merged into the content of the local model by performing a bitwise OR operation on the pair of models. This is advantageous because the local model can learn the signature of a new type of malicious code before confronting an instance of the new malicious code.

[0077] FIG. 6 is a simple illustration of a method 600 for detecting content anomalies in accordance with some embodiments. As shown, at 602, an input dataset, or a portion thereof, is received. As in 302 for the training dataset, the input dataset can be network data packets or frames, files containing data in variety of types and formats, or a stream of bytes or tokens of different lengths. In some embodiments, the input dataset can be data traffic, such as data traffic 104.

[0078] In some embodiments, the input dataset can be a test dataset that is designed to measure how well a content anomaly detection model is trained. In some embodiments, the input dataset can be a stream of incoming bytes that should be scrutinized before reaching its destinations.

[0079] At 604, n-grams of different sizes are generated from the input dataset. In some embodiments, as in 304, n-grams of different sizes can be generated by sliding windows of corresponding sizes over the input dataset.

[0080] At 606, the number of distinct n-grams of different sizes in the input dataset that are not found in the training dataset are counted. In some embodiments, a counter is incremented when a distinct n-gram is observed for the first time by checking a Bloom filter that was previously trained with known-good n-grams to see if the distinct n-gram has seen in the training dataset. In some embodiments, the number of distinct n-grams is counted at the content anomaly detector. In some embodiment, it can be counted at the HEG analyzer.

[0081] At 608, an anomaly score for the input dataset is computed using the number of distinct n-grams counted in 606. In some embodiments, the anomaly score is computed using the following formula:

$$\text{Anomaly Score} = \frac{N}{T},$$

where T is the total number of n-grams in the input dataset and N is the number of new distinct n-grams not found in the training dataset. The higher the anomaly score, the more likely that the input dataset contains an instance of malicious code.

[0082] In some embodiments, the anomaly score can be calculated at the content anomaly detector. In some embodiments, it can be computed at the HEG analyzer.

[0083] At 610, the anomaly score can be used to determine whether the input dataset is anomalies. In some embodiments, if the input dataset generates an anomaly score above a threshold value, it can be used to generate content anomaly signatures. In some embodiments, for example, different sizes of n-grams can be generated by sliding windows of corresponding sizes over the input data. In some embodiments, the n-grams of different sizes are filtered to remove normal n-grams using a known-clean dataset, as described above in connection with 406 of FIG. 4.

[0084] FIG. 7 is a simple illustration of another method 700 for training and detecting content anomalies in accordance with some embodiments. As shown, at 702, an input dataset is classified. In some embodiments, content anomaly detector 106 can act as a network anomaly flow classifier to determine whether the input dataset contains instances of malicious code. Detector 106 classifies the data traffic and directs it to an appropriate server, such as production server 110 or shadow server 112.

[0085] At 704, the likelihood of the input dataset containing malicious n-grams is determined. In some embodiments, the likelihood of the input dataset containing malicious

n-grams based on anomaly scores, which can be computed as discussed above in connection with 608, is determined.

[0086] If it is determined at 704 that the likelihood of the input dataset containing malicious n-grams is low, the input dataset is further processed at 706. In some embodiments, an input dataset, such as data traffic 104, that is determined not to contain malicious n-grams can be sent to a production server, such as production server 110, to be processed further.

[0087] If, however, it is determined at 704 that the likelihood of the input dataset containing malicious n-grams is above a threshold, the input dataset can be flagged and sent at 708 to a non-production server that runs an additional content anomaly detector. In some embodiments, an input dataset, such as data traffic 104, that is deemed likely to contain malicious n-grams is sent to host-based detector 114 that can be part of, or attached to, a shadow server, such as shadow server 112. The host-based detector can employ more sophisticated techniques that require more processing time and computational overhead to examine the input dataset that is sent from detector 106.

[0088] At 710, it is determined whether the flagged input dataset contains malicious n-grams. In some embodiments, the host-based detector examines the flagged input dataset and determine whether to process it further or to drop it.

[0089] If it is determined at 710 that the flagged input dataset contains malicious n-grams, the flagged dataset can be dropped or isolated at 716. In some embodiments, the host-based detector can be used to isolate the input dataset.

[0090] At 718, the isolated dataset is used to generate content anomalous signatures. In some embodiments, the host-based detector can generate new content anomaly signatures and feed it back to the classifier for further training of its content anomaly detection model, as described above in connection with FIG. 4.

[0091] If, however, it is determined at 710 that the flagged input dataset is unlikely to contain malicious n-grams, the input dataset can be processed further at 712. In some embodiments, the host-based detector can send the input dataset to the shadow server for further processing. For example, in some embodiments, the shadow server processes the input dataset in a protected environment, such as a sandbox, and monitors the state change of the sandbox. In some embodiments, the sandbox is implemented using an emulator or a virtual machine.

[0092] At 714, the content anomaly detection model is updated to include the n-grams contained in the flagged input dataset, as described above in connection with 302 and 304 of FIG. 3. This may prevent them from causing false positives again in the future, thereby reducing the false positive rate of the content anomaly detection model. In some embodiments, the host-based detector can provide feedback necessary for the classifier to update its content anomaly detection model.

[0093] Although the invention has been described and illustrated in the foregoing illustrative embodiments, it is understood that the present disclosure has been made only by way of example, and that numerous changes in the details of implementation of the invention can be made without departing from the spirit and scope of the invention, which is only limited by the claims which follow. Features of the disclosed embodiments can be combined and rearranged in various ways.

What is claimed is:

1. A method for outputting data based on anomaly detection, comprising:

receiving a known-good dataset;

storing distinct n-grams from the known-good dataset to form a binary anomaly detection model;

receiving known-good new n-grams;

computing a rate of receipt of distinct n-grams in the new n-grams;

determining whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams;

using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

2. The method of claim 1, wherein the binary detection model is represented using a Bloom filter.

3. The method of claim 1, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given time period.

4. The method of claim 1, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given volume of data.

5. The method of claim 1, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

6. A method for outputting data based on anomaly detection, comprising:

receiving known anomaly signatures;

generating n-grams of different sizes using the known anomaly signatures;

storing abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model;

using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

7. The method of claim 6, wherein the anomaly signatures include Snort rules.

8. The method of claim 6, wherein the binary anomaly detection model is represented using a Bloom filter.

9. The method of claim 6, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

10. A method for outputting data based on anomaly detection, comprising:

receiving a shared binary anomaly detection model;

comparing the shared binary anomaly detection model with a local anomaly detection model;

combining the shared binary anomaly detection model with the local anomaly detection model to form a new binary anomaly detection model;

using the model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

11. The method of claim 10, wherein the shared binary anomaly detection model is represented using a Bloom filter.

12. The method of claim 10, wherein the local binary anomaly detection model is represented using a Bloom filter.

13. The method of claim 10, wherein the comparing is performed using a bitwise AND operation.

14. The method of claim 10, wherein the combining is performed using a bitwise OR operation.

15. The method of claim 10, wherein the new binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the new binary anomaly detection model for an n-gram in the input dataset.

16. A method for outputting data based on anomaly detection, comprising:

receiving an input dataset;

generating n-grams of different sizes from the input dataset;

counting the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model;

computing an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset;

using the anomaly score to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

17. The method of claim 16, wherein the binary anomaly detection model is represented using a Bloom filter.

18. The method of claim 16, wherein the n-grams of different sizes are generated using a window applied to the input dataset.

19. The method of claim 16, wherein the anomaly score is compared to a threshold level to determine whether the input dataset contains an anomaly.

20. A method for outputting data based on anomaly detection, comprising:

receive an input dataset;

using a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly;

if the input dataset is determined to be likely to contain an anomaly, dropping the input dataset; and

if the input dataset is determined to be unlikely to contain an anomaly, outputting the input dataset based on whether the input dataset contains an anomaly.

21. The method of claim 20, wherein the binary anomaly detection model is represented using a Bloom filter.

22. The method of claim 20, wherein the binary anomaly detection model is used to determine if the input dataset is likely to contain an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

23. The method of claim 20, further comprising:

determining whether the input dataset contains malicious n-grams; and

if the input dataset is determined to not contain malicious n-grams, using the input dataset to perform a function related to the input dataset.

24. The method of claim 23, further comprising updating the binary anomaly detection model using the input dataset.

25. The method of claim 20, further comprising generating a content anomaly signature using the input dataset.

26. A computer-readable medium containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, the method comprising:

receiving a known-good dataset;

storing distinct n-grams from the known-good dataset to form a binary anomaly detection model;

receiving known-good new n-grams;

computing a rate of receipt of distinct n-grams in the new n-grams;

determining whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams;

using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

27. The medium of claim 26, wherein the binary detection model is represented using a Bloom filter.

28. The medium of claim 26, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given time period.

29. The medium of claim 26, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given volume of data.

30. The medium of claim 26, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

31. A computer-readable medium containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, the method comprising:

receiving known anomaly signatures;

generating n-grams of different sizes using the known anomaly signatures;

storing abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model;

using the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

32. The medium of claim 31, wherein the anomaly signatures include Snort rules.

33. The medium of claim 31, wherein the binary anomaly detection model is represented using a Bloom filter.

34. The medium of claim 31, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

35. A computer-readable medium containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, the method comprising:

receiving a shared binary anomaly detection model;

comparing the shared binary anomaly detection model with a local anomaly detection model;

combining the shared binary anomaly detection model with the local anomaly detection model to form a new binary anomaly detection model;

using the model to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

36. The medium of claim 35, wherein the shared binary anomaly detection model is represented using a Bloom filter.

37. The medium of claim 35, wherein the local binary anomaly detection model is represented using a Bloom filter.

38. The medium of claim 35, wherein the comparing is performed using a bitwise AND operation.

39. The medium of claim 35, wherein the combining is performed using a bitwise OR operation.

40. The medium of claim 35, wherein the new binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the new binary anomaly detection model for an n-gram in the input dataset.

41. A computer-readable medium containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, the method comprising:

receiving an input dataset;

generating n-grams of different sizes from the input dataset;

counting the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model;

computing an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset;

using the anomaly score to determine whether an input dataset contains an anomaly; and

outputting the input dataset based on whether the input dataset contains an anomaly.

42. The medium of claim 41, wherein the binary anomaly detection model is represented using a Bloom filter.

43. The medium of claim 41, wherein the n-grams of different sizes are generated using a window applied to the input dataset.

44. The medium of claim 41, wherein the anomaly score is compared to a threshold level to determine whether the input dataset contains an anomaly.

45. A computer-readable medium containing computer-executable instructions that, when executed by a processor, cause the processor to perform a method for outputting data based on anomaly detection, the method comprising:

receiving an input dataset;

using a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly;

if the input dataset is determined to be likely to contain an anomaly, dropping the input dataset; and

if the input dataset is determined to be unlikely to contain an anomaly, outputting the input dataset based on whether the input dataset contains an anomaly.

46. The medium of claim 45, wherein the binary anomaly detection model is represented using a Bloom filter.

47. The medium of claim 45, wherein the binary anomaly detection model is used to determine if the input dataset is likely to contain an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

48. The medium of claim 45, the method further comprising:

determining whether the input dataset contains malicious n-grams; and

if the input dataset is determined to not contain malicious n-grams, using the input dataset to perform a function related to the input dataset.

49. The medium of claim 48, the method further comprising updating the binary anomaly detection model using the input dataset.

50. The medium of claim 45, the method further comprising generating a content anomaly signature using the input dataset.

51. A system for outputting data based on anomaly detection, comprising:

a digital processing device that:

receives a known-good dataset;

stores distinct n-grams from the known-good dataset to form a binary anomaly detection model;

receives known-good new n-grams;

computes a rate of receipt of distinct n-grams in the new n-grams;

determines whether further training of the anomaly detection model is necessary based on the rate of receipt on distinct n-grams;

uses the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputs the input dataset based on whether the input dataset contains an anomaly.

52. The system of claim 51, wherein the binary detection model is represented using a Bloom filter.

53. The system of claim 51, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given time period.

54. The system of claim 51, wherein the rate of receipt of distinct n-grams is based on how many distinct n-grams are received in a given volume of data.

55. The system of claim 51, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

56. A system for outputting data based on anomaly detection, comprising:

a digital processing device that:

receives known anomaly signatures;

generates n-grams of different sizes using the known anomaly signatures;

stores abnormal n-grams in the n-grams of different sizes in a binary anomaly detection model;

uses the binary anomaly detection model to determine whether an input dataset contains an anomaly; and

outputs the input dataset based on whether the input dataset contains an anomaly.

57. The system of claim 56, wherein the anomaly signatures include Snort rules.

58. The system of claim 56, wherein the binary anomaly detection model is represented using a Bloom filter.

59. The system of claim 56, wherein the binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

60. A system for outputting data based on anomaly detection, comprising:

a digital processing device that:

receives a shared binary anomaly detection model;

compares the shared binary anomaly detection model with a local anomaly detection model;

combines the shared binary anomaly detection model with the local anomaly detection model to form a new binary anomaly detection model;

uses the model to determine whether an input dataset contains an anomaly; and

outputs the input dataset based on whether the input dataset contains an anomaly.

61. The system of claim 60, wherein the shared binary anomaly detection model is represented using a Bloom filter.

62. The system of claim 60, wherein the local binary anomaly detection model is represented using a Bloom filter.

63. The system of claim 60, wherein the comparing is performed using a bitwise AND operation.

64. The system of claim 60, wherein the combining is performed using a bitwise OR operation.

65. The system of claim 60, wherein the new binary anomaly detection model is used to determine if the input dataset contains an anomaly by checking the new binary anomaly detection model for an n-gram in the input dataset.

66. A system for outputting data based on anomaly detection, comprising:

a digital processing device that:

receives an input dataset;

generates n-grams of different sizes from the input dataset;

countes the number of distinct n-grams in the n-grams of different sizes that are not present in a binary anomaly detection model;

computes an anomaly score based upon the number of distinct n-grams and a total count of the n-grams in the input dataset;

uses the anomaly score to determine whether an input dataset contains an anomaly; and

outputs the input dataset based on whether the input dataset contains an anomaly.

67. The system of claim 66, wherein the binary anomaly detection model is represented using a Bloom filter.

68. The system of claim 66, wherein the n-grams of different sizes are generated using a window applied to the input dataset.

69. The system of claim 66, wherein the anomaly score is compared to a threshold level to determine whether the input dataset contains an anomaly.

70. A system for outputting data based on anomaly detection, comprising:

a digital processing device that:

receives an input dataset;

uses a binary anomaly detection model to determine whether an input dataset is likely to contain an anomaly;

if the input dataset is determined to be likely to contain an anomaly, drops the input dataset; and

if the input dataset is determined to be unlikely to contain an anomaly, outputs the input dataset based on whether the input dataset contains an anomaly.

71. The system of claim 70, wherein the binary anomaly detection model is represented using a Bloom filter.

72. The system of claim 70, wherein the binary anomaly detection model is used to determine if the input dataset is likely to contain an anomaly by checking the binary anomaly detection model for an n-gram in the input dataset.

73. The system of claim 70, wherein the digital processing device also:

determines whether the input dataset contains malicious n-grams; and

if the input dataset is determined to not contain malicious n-grams, uses the input dataset to perform a function related to the input dataset.

74. The system of claim 73, wherein the digital processing device also updates the binary anomaly detection model using the input dataset.

75. The system of claim 70, wherein the digital processing device also generates a content anomaly signature using the input dataset.

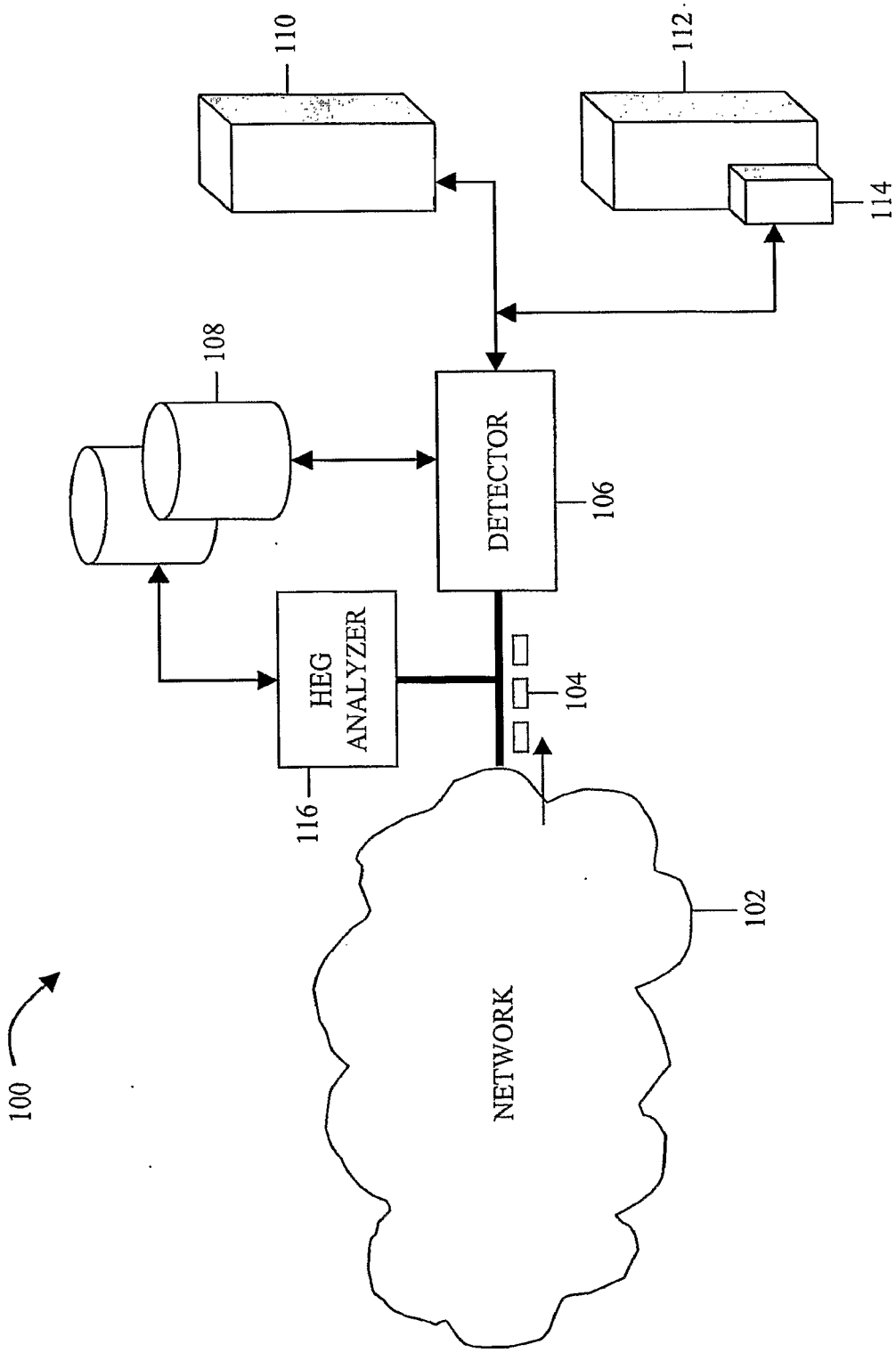

100
NETWORK
102
HEG ANALYZER
116
104
DETECTOR
106
108
110
112
114

FIG. 1

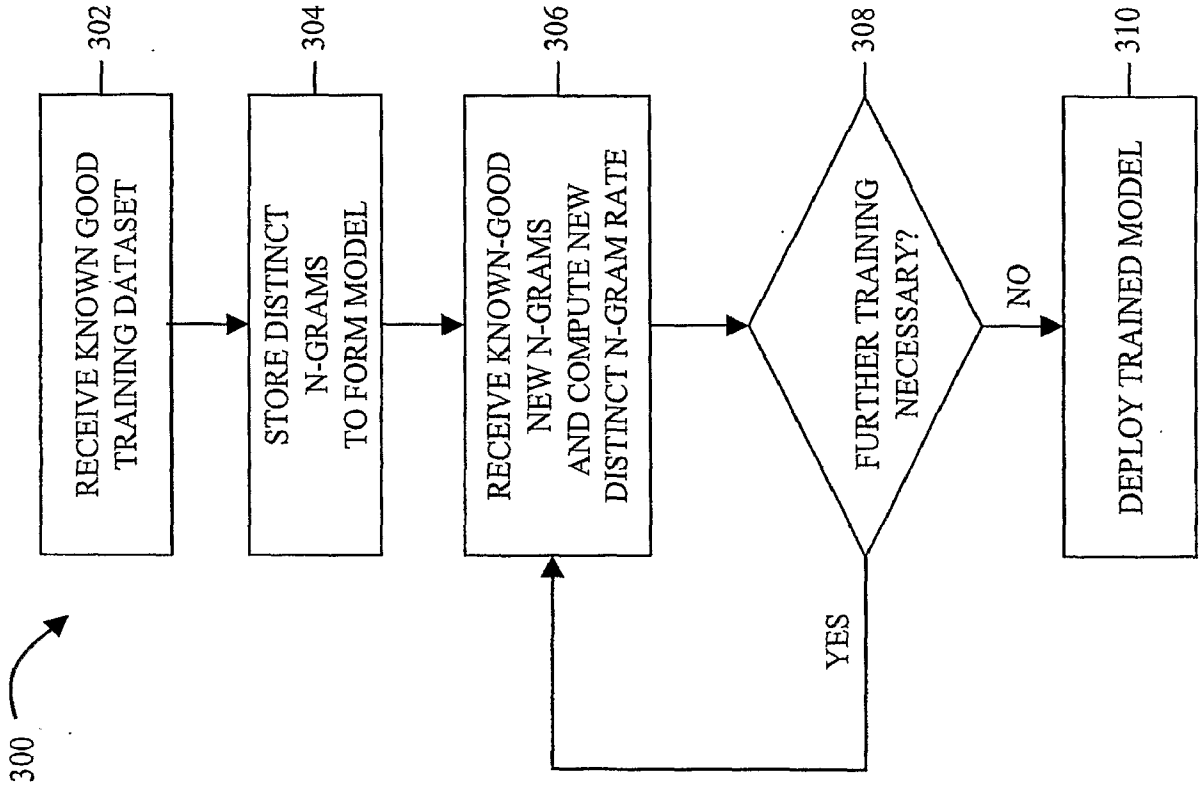

300
302
RECEIVE KNOWN GOOD TRAINING DATASET
304
STORE DISTINCT N-GRAMS TO FORM MODEL
306
RECEIVE KNOWN-GOOD NEW N-GRAMS AND COMPUTE NEW DISTINCT N-GRAM RATE
308
FURTHER TRAINING NECESSARY?
YES
NO
310
DEPLOY TRAINED MODEL

FIG. 3

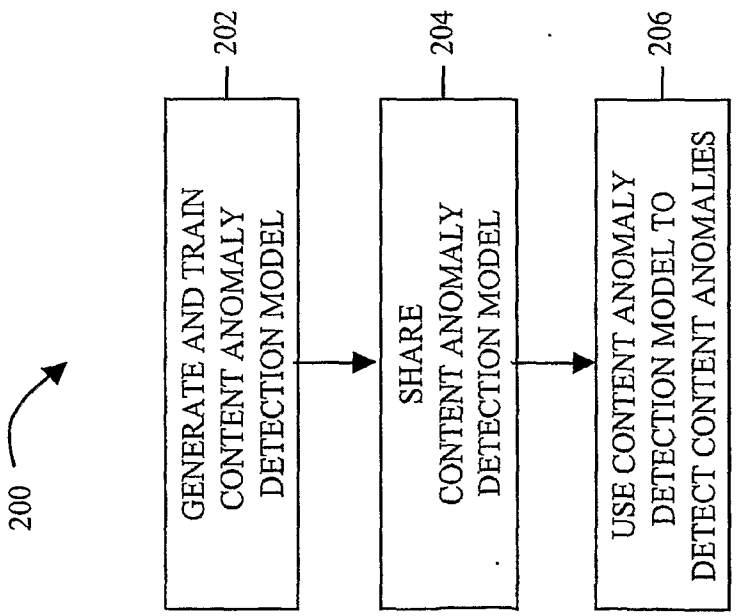

200
202
GENERATE AND TRAIN CONTENT ANOMALY DETECTION MODEL
204
SHARE CONTENT ANOMALY DETECTION MODEL
206
USE CONTENT ANOMALY DETECTION MODEL TO DETECT CONTENT ANOMALIES

FIG. 2

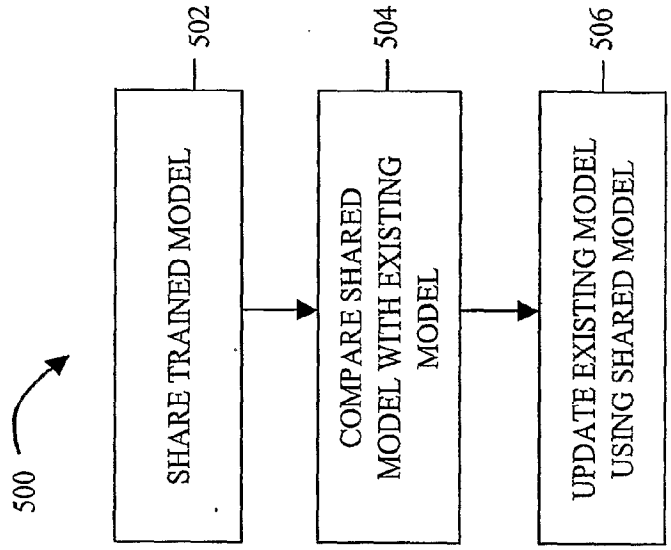

500
SHARE TRAINED MODEL
502
COMPARE SHARED MODEL WITH EXISTING MODEL
504
UPDATE EXISTING MODEL USING SHARED MODEL
506

FIG. 5

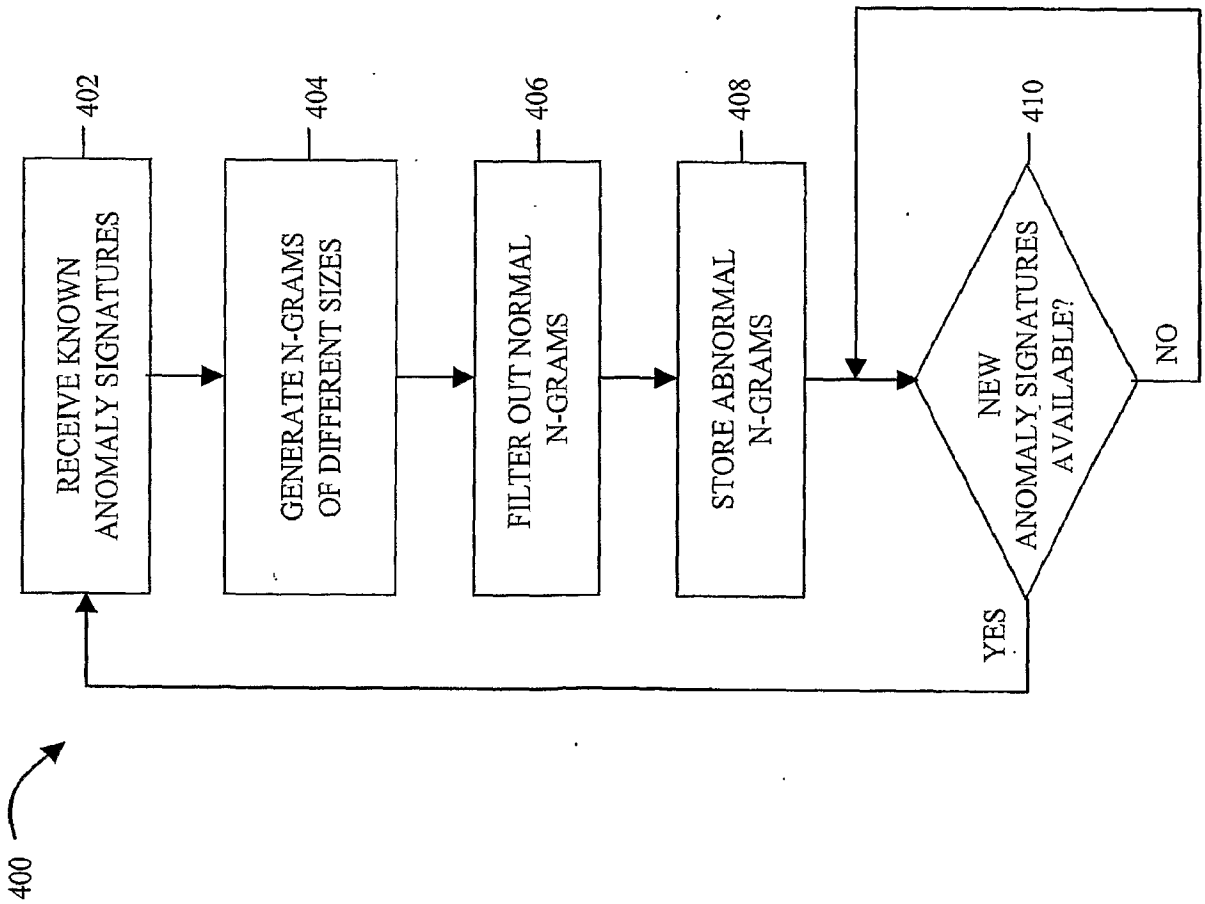

400
RECEIVE KNOWN ANOMALY SIGNATURES
402
GENERATE N-GRAMS OF DIFFERENT SIZES
404
FILTER OUT NORMAL N-GRAMS
406
STORE ABNORMAL N-GRAMS
408
NEW ANOMALY SIGNATURES AVAILABLE?
410
YES
NO

FIG. 4

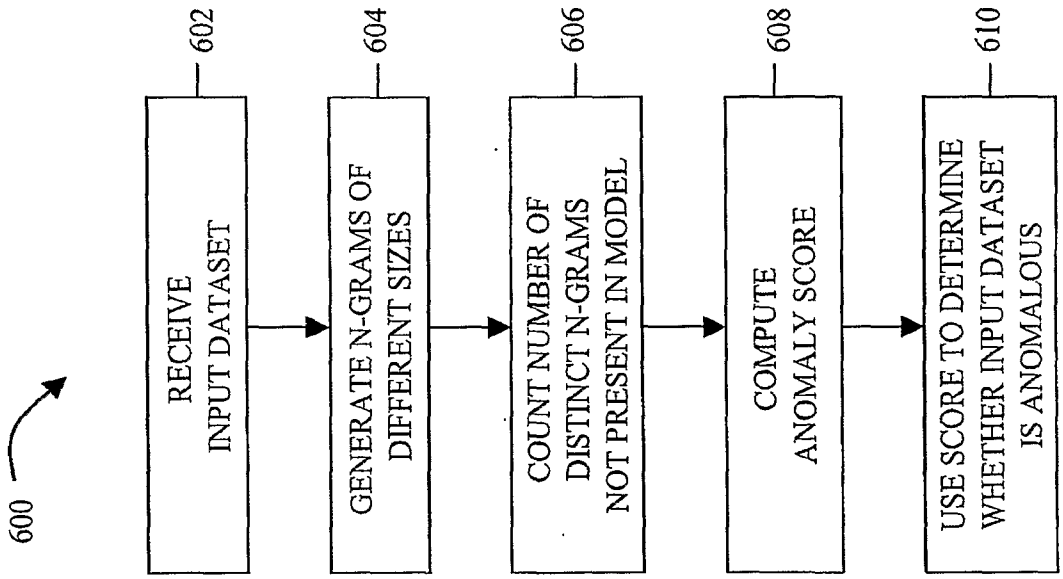

600
602
RECEIVE INPUT DATASET
604
GENERATE N-GRAMS OF DIFFERENT SIZES
606
COUNT NUMBER OF DISTINCT N-GRAMS NOT PRESENT IN MODEL
608
COMPUTE ANOMALY SCORE
610
USE SCORE TO DETERMINE WHETHER INPUT DATASET IS ANOMALOUS

FIG. 6

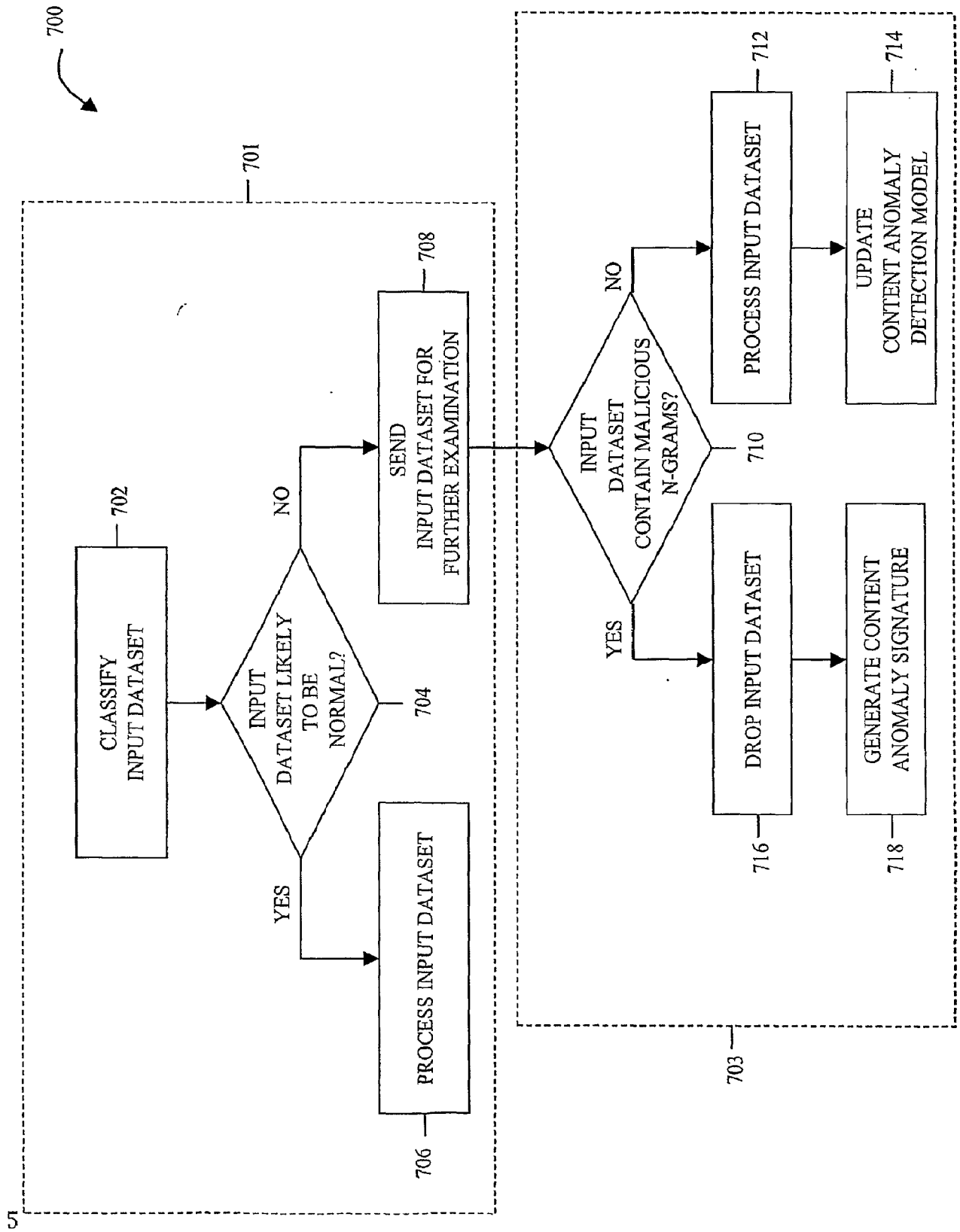

700
701
CLASSIFY INPUT DATASET
702
INPUT DATASET LIKELY TO BE NORMAL?
704
YES
NO
PROCESS INPUT DATASET
706
SEND INPUT DATASET FOR FURTHER EXAMINATION
708
INPUT DATASET CONTAIN MALICIOUS N-GRAMS?
710
YES
NO
DROP INPUT DATASET
716
GENERATE CONTENT ANOMALY SIGNATURE
718
PROCESS INPUT DATASET
712
UPDATE CONTENT ANOMALY DETECTION MODEL
714
703
5

FIG. 7