

METHOD AND SYSTEM FOR DETECTING AND PREVENTING MALWARE EXECUTION IN A TARGET SYSTEM

BACKGROUND

Field

[0001] This application generally relates to enterprise system security. In particular, this application describes a method and system for detecting malware on computer systems of the enterprise and for preventing execution of the malware.

Description of Related Art

[0002] Small and large computer systems may become infected with malware. Often times, the presence of the malware goes unnoticed until, for example, a ransom is received, computers become locked, etc. Such an attack may result in loss of intellectual property, lost productivity, lost revenue, and in some cases necessitate repair/replacement of components of the computer systems.

SUMMARY

[0003] A system for detecting and preventing execution of malware on a target system includes an interface for receiving training data. The training data includes domain names known to be legitimate and domain names known to be associated with malware. The system includes a processor in communication with the interface; and non-transitory computer readable media in communication with the processor that stores instruction code. The instruction code is executed by the processor and causes the processor to train a first model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names using a supervised learning methodology. The processor trains a second model to predict a correct domain name associated with domain names in the training data using an unsupervised learning methodology. The processor trains a third model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names based on an output of the first learning model and the second learning model. Subsequent to training of the first, second, and third models, the processor receive a new domain name and processes the new domain name through the trained models to determine whether the domain name is considered to be a legitimate domain name or a malware-associated domain name. The processor communicates a report to the target system based on the processing to thereby facilitate removal of the malware from the target system.

BRIEF DESCRIPTION OF THE DRAWINGS

- [0004]** Fig. 1 illustrates an exemplary environment in which a malware detection system (MDS) for detecting and preventing execution of malware in a target system operates;
- [0005]** Fig. 2 logically illustrates processing of domain names through a group of models of the MDS to determine whether the domain names are legitimate domain names;
- [0006]** Fig. 3 illustrates operations performed by the MDS for training machine-learning models of the MDS;
- [0007]** Fig. 4 logically illustrates training of a first model of the MDS;
- [0008]** Fig. 5 logically illustrates training of a second model of the MDS;
- [0009]** Fig. 6 logically illustrates training of a third model of the MDS; and
- [0010]** Fig. 7 illustrates an exemplary computer system that may form part of or implement the systems described in the figures or in the following paragraphs.

DETAILED DESCRIPTION

[0011] The embodiments described below overcome the problems described above by providing a system that utilizes machine learning algorithms to continuously monitor domain name logs generated by an enterprise system for the presence of malware related domain names. The system proactively alerts an operator of the enterprise system of the possibility that a given domain name is associated with malware. This allows the operator of the target system to address potential malware attacks in a way that minimizes or eliminates unplanned downtime and any loss of revenue incurred as a result.

[0012] Fig. 1 illustrates an exemplary environment 100 that includes various systems/devices that facilitate detecting and preventing execution of malware on a target system. Exemplary systems/devices of the environment 100 include a malware detection system MDS 102, a user terminal 104, an enterprise/target system 105, and training data source(s) 106. Other entities/systems of the environment 100 may include malware domains 108 and legitimate domains 109.

[0013] The various entities of the environment 100 may communicate with one another via a network 107, such as the Internet and may correspond to computer systems such as an Intel®, AMD®, or PowerPC® based computer system or a different computer system and can include application specific computer systems. The computer systems may include an operating system, such as Microsoft

Windows®, Linux, Unix® or other operating system. Any terminals described herein may be desktop PCs and/or mobile terminals.

[0014] The enterprise system 105 may correspond to a networked group of user computers that connect to an outside network 107 via a gateway computer system. Users of the computers (e.g., such as employees of the enterprise) may utilize services reliant upon the outside network 107 such as web browsers, email applications for sending and receiving email messages, chat programs, etc. Network traffic generated by these services may be routed through a firewall/gateway device of the enterprise system 105. The firewall/gateway device may log the network traffic. For example, information in URL requests and responses (e.g., headers, cookies, body of entity) may be logged to facilitate management of network usage and infrastructure of the enterprise system 105.

[0015] In an embodiment, the logs may be communicated to the MDS 102. The logs may be communicated in real-time or in batches. In this regard, the enterprise system 105 may be configured to facilitate communicating the network transaction logs to the MDS 102 via one or more APIs of the MDS 102. For example, the enterprise system 105 may be configured to communicate with the MDS 102 via an API such as a webserver API, a SOAP-based web service, a RESTful API, and/or a different type of API.

[0016] The malware domains 108 and legitimate domains 109 may be associated with computer systems either setup for legitimate purposes (legitimate domains 109) or nefarious purposes (malware domains 108). Communications between the enterprise system 105 and these domains 108 and 109 may initiate via HTTP get and post requests originating from the enterprise system 105 or via a different communication protocol. For example, the enterprise system 105 and the computer systems associated with the malware domains 108 and legitimate domains 109 may be configured to communicate with one another via an API such as a webserver API, a SOAP-based web service, a RESTful API, and/or a different type of API.

[0017] The requests may specify a domain name that is used to direct the requests to a computer system associated with the domain name. In the case of a malware domain, a response to the request may direct malware operating on an infected computer within the enterprise system 105 to communicate information to the same malware domain or a different malware domain. The malware on the infected computer and the computer system associated with the malware domain may cooperate to continually change domain names to avoid detection. In practice, the domain names used

by the malware-associated computer systems may appear as an almost random or unintelligible sequence of characters. This is different from legitimate domain names, which are designed to be easy to remember.

[0018] The training data source(s) 106 may correspond to one or more systems that store information that defines lists of legitimate and/or illegitimate (i.e., malware) domain names. For example, the systems may correspond to web systems such as Majestic's® Top Million, Cisco's® Umbrella Top Million, and Amazon's® Alexa Top Sites. Information from the data source(s) 106 may be communicated to the MDS 102 via an API such as a webserver API, a SOAP-based web service, a RESTful API, and/or a different type of API and aggregated. As will be described further below, the MDS 102 may aggregate the domain names provided by the different sources to facilitate training AI subsystems of the MDS 102.

[0019] The MDS 102 may include a processor 125, input/output subsystem 110, and an AI subsystem 115. The MDS 102 may include other subsystems.

[0020] It is contemplated that the I/O subsystem 110, AI subsystem 115, and any other subsystem referenced herein may correspond to a stand-alone computer system such as an Intel®, AMD®, or PowerPC® based computer system or a different computer system and can include application specific computer systems. The computer systems may include an operating system, such as Microsoft Windows®, Linux, Unix® or other operating system. It is also contemplated that operations performed on the various subsystems may be combined into a fewer or greater number of subsystems to facilitate speed scaling, cost reductions, etc.

[0021] The I/O subsystem 110 of the MDS 102 includes one or more input, output, or input/output interfaces and is configured to facilitate communications with entities outside of the MDS 102. In this regard, the I/O subsystem 110 may be configured to dynamically determine the communication methodology utilized by entities of the environment 100 and may communicate information to the entities with an interface that uses a determined communication methodology. For example, the I/O subsystem 110 may determine that a first entity utilizes a RESTful API and may, therefore, communicate with the entity using an interface that uses a RESTful communication methodology.

[0022] The I/O subsystem 110 may implement a web browser to facilitate generating one or more web-based interfaces through which users of the MDS 102, enterprise system 105, and/or other systems may interact with the MDS 102. The web browser may implement a web services interface to facilitate

automating some of the web-based functionality via a computer. For example, one or more of the entities of the environment 100 may utilize the web services interfaces to access information stored by the MDS 102 and/or to communicate information to the MDS 102.

[0023] The AI subsystem 115 may correspond to hardware specifically configured to perform or assist in the performance of various natural language processing techniques such as latent Dirichlet allocation (LDA) to identify topics associated with conversations, hierarchical density based cluster analysis (H-DBSCAN) to group conversations under one or more topics, Knuth–Morris–Pratt string searching to locate and extract occurrences of a certain words within conversations, possibly linear clustering algorithms to mine text data, and/or other techniques.

[0024] The AI subsystem 115 may be further configured to implement various supervised and unsupervised machine learning models. The machine learning models may be based on algorithms such as a Holt-Winters algorithm, an exponential time smoothing (ETS) algorithm, an artificial neural network (ANN), a recurrent neural network (RNN), a seasonal auto regressive moving average (SARIMA) algorithm, a network of long short-term memories (LSTM), a gated recurring unit (GRU) algorithm.

[0025] Fig. 2 illustrates exemplary logic 200 implemented by the AI subsystem 115. Referring to Fig. 2, section 205 represents logic for receiving one or more domain names that may be associated with legitimate or malware domains.

[0026] In section 210, the received domains are processed through a pair of machine learning models. Each machine learning model may implement so-called deep learning logic (i.e., recurrent neural networks that include several interconnected neural layers.) Such models are given data to ingest and in some cases told a result to be interpreted from the data (supervised) or are allowed to elicit different categories from the data (unsupervised). The models are trained to arrive at the result based on the ingested data and the quality of the results is dependent in part on the diversity of the data being ingested.

[0027] A first so-called “right brain” model utilizes a bi-directional long short-term memory (LSTM) recurrent neural network (RNN), trained using a supervised learning methodology, to classify domain names as being either legitimate domain names or malware domain names based on the sequence of characters in a domain name. That is, the output of the first learning model may be a binary value indicative of the legitimacy of the domain name. In some implementations, the first learning model may generate a confidence score indicative of the confidence the model has in the prediction.

[0028] A second so-called “left brain” model utilizes a pair of long short-term memory (LSTM) RNNs, trained using an unsupervised learning methodology, to generate a result based on the sequence of characters in a domain name. The implemented model may correspond to an autoencoder. The output of the second learning model is a predicted legitimate domain name associated with the received domain name. For example, if the received domain were “www.accenture123.com,” the predicted legitimate domain may be www.accenture.com.” The second learning model may generate a confidence score indicative of the confidence that the predicted legitimate domain is correct. In the first example, the score may be 9 out of 10, where 10 indicates 100% confidence that the result is correct. On the other hand, a domain such as “asd4533ggsw3545.com” may be predicted to be “apple.com” and have a confidence score of 1, to indicate that the predicted result is in all likelihood incorrect. The score may be utilized during training of the model.

[0029] In section 315, a third learning model receives the binary prediction from the first learning model and the predicted legitimate domain name from the second learning model and classifies the received domain name as being a legitimate domain name or a malware domain name. The third learning model may utilize an LSTM RNN, trained using a supervised methodology.

[0030] A report that lists domain names and the corresponding prediction result associated with the domain names may be generated and communicated to the enterprise system 105 to facilitate removal of any detected malware. Alternatively, logic 200 may be employed on an ad hoc basis to evaluate a received domain name as being a legitimate domain name or a malware domain name, and to produce metadata associated with this determination, such as, for example, a risk score, a URL breakdown, and the outputs (scores) from the “right brain” and “left brain” models.

[0031] The CPU 125 executes instruction code stored in a memory device 127 for coordinating activities performed between the various subsystems. The processor 125 may correspond to a stand-alone computer system such as an Intel®, AMD®, or PowerPC® based computer system or a different computer system and can include application specific computer systems. The computer systems may include an operating system, such as Microsoft Windows®, Linux, Unix® or other operating system.

[0032] Fig. 2 illustrates exemplary operations performed by the CPU 125 and/or other modules of the MDS 102 for training the various machine-learning models to detect malware communications between the enterprise system 105 and the malware domains 105. In this regard, the operations may be implemented via instruction code stored in non-transitory computer readable media 127 that resides

within the subsystems configured to cause the respective subsystems to perform the operations illustrated in the figures and discussed herein. The operations of Fig. 2 are best understood with reference to the diagrams of Figs. 3-6.

[0033] Referring to Fig. 3, at operation 300, training data that defines lists of legitimate and/or illegitimate domain names is ingested from the data source(s) 106. In some instances, the ingested data may be pre-processed by performing operations such as deduplication, removal of foreign-language domains, removal of anomalous domains, etc.

[0034] At operation 305, the first learning model (i.e., right brain) may be trained using the legitimate (i.e., good) and malware (i.e., bad) domains. Referring to Fig. 4., during training of the first learning model, the domains are embedded (i.e., represented as a vector) to facilitate processing within a bi-directional LSTM encoder. First and second dropout operations are performed to exclude the activation and weight updates to certain neurons of the model to thereby reduce overfitting and to improve model performance.

[0035] The model generates a binary result (i.e., good domain or bad domain). A determination is made as to the correctness of the prediction and feedback into the LSTM encoder. For example, the result is compared against the known legitimacy of the domain (i.e., that the domain is actually known to be good or bad). The result of the comparison is fed back into the model to update the various weights, activation states, etc. of the LSTM model

[0036] At operation 310, the second learning model (i.e., left brain) may be trained using the legitimate domains. In some cases, only the legitimate domains are used in training the second learning model. Referring to Fig. 5., during training of the first learning model, the good domains are embedded (i.e., represented as a vector) to facilitate processing within the LSTM RNNs (i.e., the encoder and decoder LSTM). Repeat vector processing is utilized to connect the respective LSTMs together.

[0037] The output of the second learning model is a predicted legitimate domain name associated with the received domain name. For example, if the received domain were “www.accenture123.com,” the predicted legitimate domain may be “www.accenture.com.” The second learning model may generate a confidence score indicative of the confidence that the predicted legitimate domain is correct. The result of the comparison is fed back into the model to update the various weights, activation states, etc. of the LSTM model.

[0038] At operation 315, the third learning model (i.e., controller brain) may be trained using the results of the first and second models to classify the domain names as either legitimate (i.e., good) and malware (i.e., bad) domains. The result is compared against the known legitimacy of the domain (i.e., that the domain is actually known to be good or bad). The result of the comparison is fed back into the third learning to update the various weights, activation states, etc. of the LSTM RNN of the third learning model.

[0039] Once the models are trained, the MDS 102 may be ready to process new domain names received from, for example, the enterprise system. The MDS 102 may communicate the results of the processing to the enterprise system 105. An operator of the system may take note that a particular domain name is associated with malware, track down the computer system that sent or received information to/from the domain name, and remove the malware. Automatic actions may also be taken in the enterprise system 105, to, e.g., quarantine the computer system that sent or received information to/from the malware domain name.

[0040] Fig. 6 illustrates a computer system 700 that may form part of or implement the systems, environments, devices, etc., described above. The computer system 700 may include a set of instructions 745 that the processor 705 may execute to cause the computer system 700 to perform any of the operations described above. The computer system 700 may operate as a stand-alone device or may be connected, e.g., using a network, to other computer systems or peripheral devices.

[0041] In a networked deployment, the computer system 700 may operate in the capacity of a server or as a client computer in a server-client network environment, or as a peer computer system in a peer-to-peer (or distributed) environment. The computer system 700 may also be implemented as or incorporated into various devices, such as a personal computer or a mobile device, capable of executing instructions 745 (sequential or otherwise) causing a device to perform one or more actions. Further, each of the systems described may include a collection of subsystems that individually or jointly execute a set, or multiple sets, of instructions to perform one or more computer operations.

[0042] The computer system 700 may include one or more memory devices 710 communicatively coupled to a bus 720 for communicating information. In addition, code operable to cause the computer system to perform operations described above may be stored in the memory 710. The memory 710 may be a random-access memory, read-only memory, programmable memory, hard disk drive or any other type of memory or storage device.

[0043] The computer system 700 may include a display 730, such as a liquid crystal display (LCD), a cathode ray tube (CRT), or any other display suitable for conveying information. The display 730 may act as an interface for the user to see processing results produced by processor 705.

[0044] Additionally, the computer system 700 may include an input device 725, such as a keyboard or mouse or touchscreen, configured to allow a user to interact with components of system 700.

[0045] The computer system 700 may also include a disk or optical drive unit 715. The drive unit 715 may include a computer-readable medium 740 in which the instructions 745 may be stored. The instructions 745 may reside completely, or at least partially, within the memory 710 and/or within the processor 705 during execution by the computer system 700. The memory 710 and the processor 705 also may include computer-readable media as discussed above.

[0046] The computer system 700 may include a communication interface 735 to support communications via a network 750. The network 750 may include wired networks, wireless networks, or combinations thereof. The communication interface 735 may enable communications via any number of communication standards, such as 802.11, 802.12, 802.20, WiMAX, cellular telephone standards, or other communication standards.

[0047] Accordingly, methods and systems described herein may be realized in hardware, software, or a combination of hardware and software. The methods and systems may be realized in a centralized fashion in at least one computer system or in a distributed fashion where different elements are spread across interconnected computer systems. Any kind of computer system or other apparatus adapted for carrying out the methods described herein may be employed.

[0048] The methods and systems described herein may also be embedded in a computer program product, which includes all the features enabling the implementation of the operations described herein and which, when loaded in a computer system, is able to carry out these operations. Computer program as used herein refers to an expression, in a machine-executable language, code or notation, of a set of machine-executable instructions intended to cause a device to perform a particular function, either directly or after one or more of a) conversion of a first language, code, or notation to another language, code, or notation; and b) reproduction of a first language, code, or notation.

[0049] While methods and systems have been described with reference to certain embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be

substituted without departing from the scope of the claims. Therefore, it is intended that the present methods and systems not be limited to the particular embodiment disclosed, but that the disclosed methods and systems include all embodiments falling within the scope of the appended claims.

CLAIMS

1. A system for detecting and preventing execution of malware on a target system, the system comprising:

an interface for receiving training data that includes domain names known to be legitimate and domain names known to be associated with malware;

a processor in communication with the interface; and

non-transitory computer readable media in communication with the processor that stores instruction code, which when executed by the processor, causes the processor to:

train a first model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names using a supervised learning methodology;

train a second model to predict a correct domain name associated with domain names in the training data using an unsupervised learning methodology;

train a third model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names based on an output of the first learning model and an output of the second learning model;

subsequent to training of the first, second, and third models, receive a new domain name;

process the new domain name through the trained models to determine whether the domain name is considered to be a legitimate domain name or a malware-associated domain name; and

communicate with the target system based on the processing to thereby facilitate removal of the malware.

2. The system according to claim 1, wherein the domain names utilized to train the first model include both legitimate domain names and malware-associated domain names, and the domain names utilized to train the second model include legitimate domain names.

3. The system according to claim 1, wherein the domain names utilized to train the third model include both legitimate domain names and malware-associated domain names.

4. The system according to claim 1, wherein the first model utilizes a long short-term memory configured as an encoder.

5. The system according to claim 1, wherein the second model implements an autoencoder that includes a first long short-term memory configured as an encoder and a second-long short-term memory configured as a decoder.

6. A method for detecting and preventing execution of malware on a target system, the method comprising:

receiving training data that includes domain names known to be legitimate and domain names known to be associated with malware;

training a first model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names using a supervised learning methodology;

training a second model to predict a correct domain name associated with domain names in the training data using an unsupervised learning methodology;

training a third model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names based on an output of the first learning model and an output of the second learning model;

subsequent to training of the first, second, and third models, receiving a new domain name;

processing the new domain name through the trained models to determine whether the domain name is considered to be a legitimate domain name or a malware-associated domain name; and

communicating with the target system based on the processing to thereby facilitate removal of the malware from the target system.

7. The method according to claim 6, wherein the domain names utilized to train the first model include both legitimate domain names and malware-associated domain names, and the domain names utilized to train the second model include legitimate domain names.

8. The method according to claim 6, wherein the domain names utilized to train the third model include both legitimate domain names and malware-associated domain names.

9. The method according to claim 6, wherein the first model utilizes a long short-term memory configured as an encoder.

10. The method according to claim 6, wherein the second model implements an autoencoder that includes a first long short-term memory configured as an encoder and a second-long short-term memory configured as a decoder.

11. A non-transitory computer readable media that stores instruction code for detecting and preventing execution of malware on a target system, the instruction code being executable by a machine for causing the machine to:

receive training data that includes domain names known to be legitimate and domain names known to be associated with malware;

train a first model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names using a supervised learning methodology;

train a second model to predict a correct domain name associated with domain names in the training data using an unsupervised learning methodology;

train a third model to classify the domain names in the training data as being legitimate domain names or malware-associated domain names based on an output of the first learning model and an output of the second learning model;

subsequent to training of the first, second, and third models, receive a new domain name;

process the new domain name through the trained models to determine whether the domain name is considered to be a legitimate domain name or a malware-associated domain name; and

communicate with the target system based on the processing to thereby facilitate removal of the malware from the target system.

12. The non-transitory computer readable media according to claim 11, wherein the domain names utilized to train the first model include both legitimate domain names and malware-associated domain names, and the domain names utilized to train the second model include legitimate domain names.

13. The non-transitory computer readable media according to claim 11, wherein the domain names utilized to train the third model include both legitimate domain names and malware-associated domain names.

14. The non-transitory computer readable media according to claim 11, wherein the first model utilizes a long short-term memory configured as an encoder.

15. The non-transitory computer readable media according to claim 11, wherein the second model implements an autoencoder that includes a first long short-term memory configured as an encoder and a second-long short-term memory configured as a decoder.