(12) UK Patent Application (19) GB (11) 2555176 (13) A

(43) Date of A Publication 25.04.2018

(21) Application No: 1711880.3

(22) Date of Filing: 24.07.2017

(30) Priority Data:
(31) 16184387 (32) 16.08.2016 (33) EP
(31) 1614025 (32) 16.08.2016 (33) GB

(71) Applicant(s):
British Telecommunications public limited company
81 Newgate Street, London, EC1A 7AJ,
United Kingdom

(72) Inventor(s):
Fadi El-Moussa
Ian Herwono

(74) Agent and/or Address for Service:
BT Group Legal Intellectual Property Department
Ground Floor, Faraday Building, 1 Knightrider Street,
London, EC4V 5BT, United Kingdom

(51) INT CL:
G06F 21/56 (2013.01) ***G06F 21/50*** (2013.01)

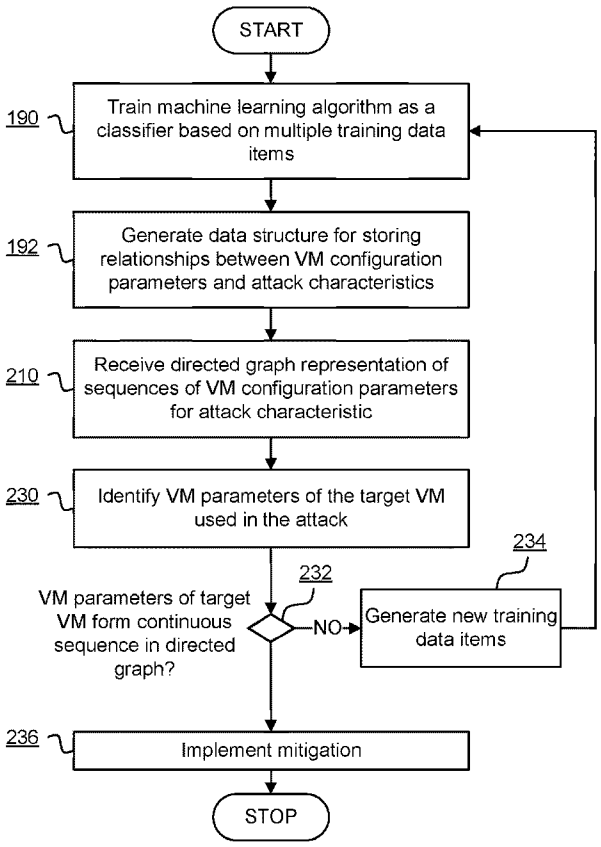
(56) Documents Cited:
US 8479276 B1 **US 20160164894 A1**
US 20130055398 A1 **US 20100199351 A1**

(58) Field of Search:
INT CL **G06F**
Other: **EPODOC, WPI, Patent Fulltext**

(54) Title of the Invention: **Machine learning for attack mitigation in virtual machine**
Abstract Title: **Machine learning for attack mitigation in a target VM**

(57) A machine learning algorithm is trained as a classifier based on a plurality of training data items, each training data item corresponding to a training virtual machine (VM) and including a representation of parameters for a configuration of the training VM and a representation of characteristics of security attacks for the training VM. One or more relationships are identified between VM configuration parameters and attack characteristics by sampling the trained machine learning algorithm. A directed graph representation of one or more sequences of VM configuration parameters for achieving a particular attack characteristic is further received. VM parameters of the target VM used in a security attack against the target VM are identified. If it is determined that the VM parameters of the target VM do not form a continuous sequence in the directed graph, then new training data items are generated and the machine learning algorithm is retrained to provide for regeneration of the identified relationships and directed graph representation. The target VM configuration may then be supplemented with a security facility associated with one of the identified VM parameters so as to protect the target VM from the attack.

FIGURE 30



GB 2555176 A

FIGURE 1

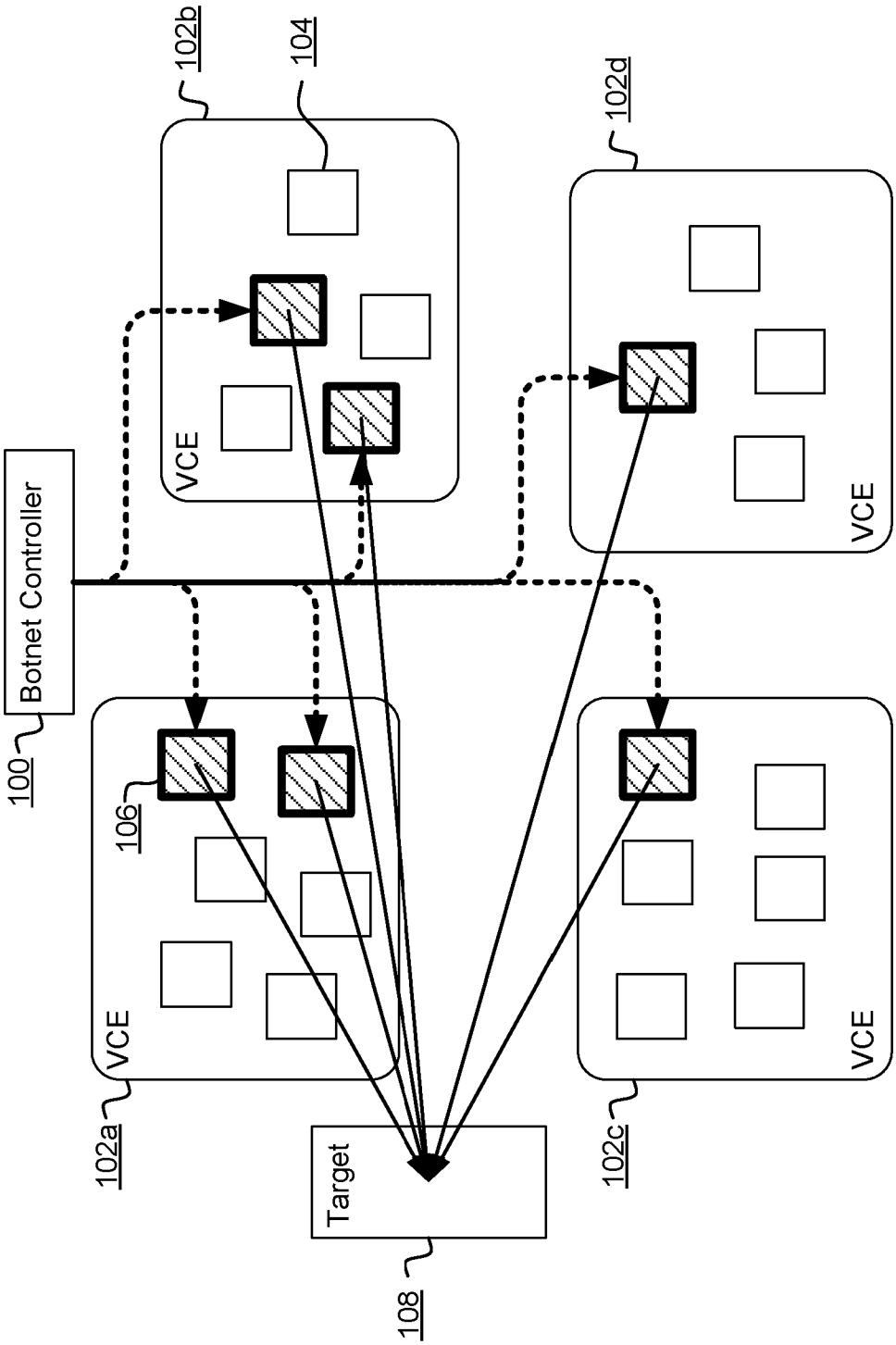

Botnet Controller
100
VCE
102a
106
VCE
102b
104
VCE
102c
VCE
102d
Target
108

FIGURE 2

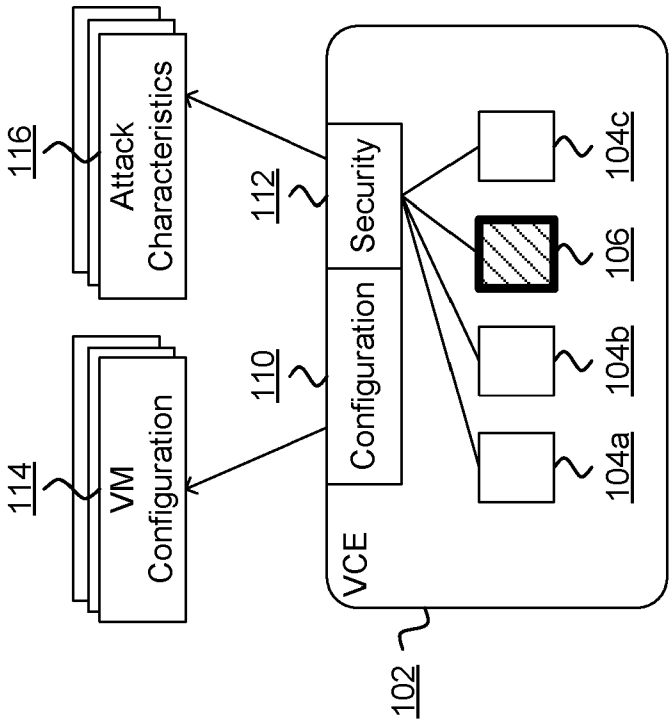


FIGURE 3

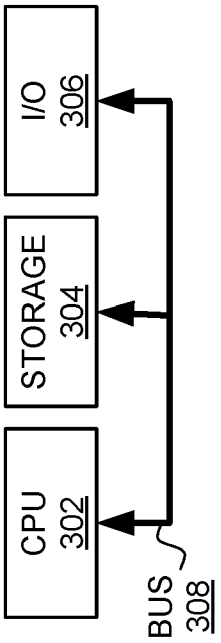


FIGURE 4

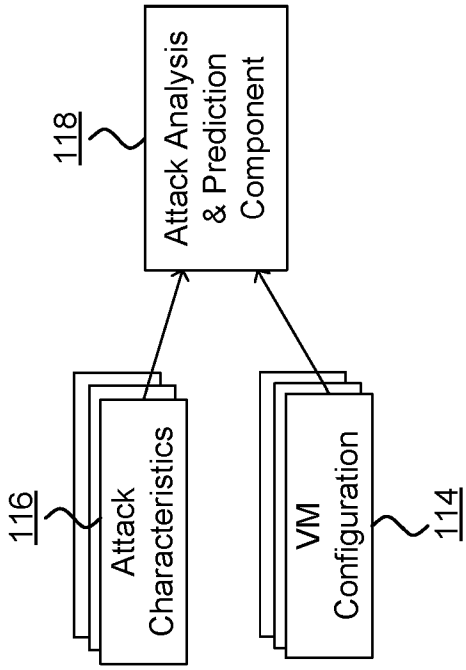


FIGURE 5

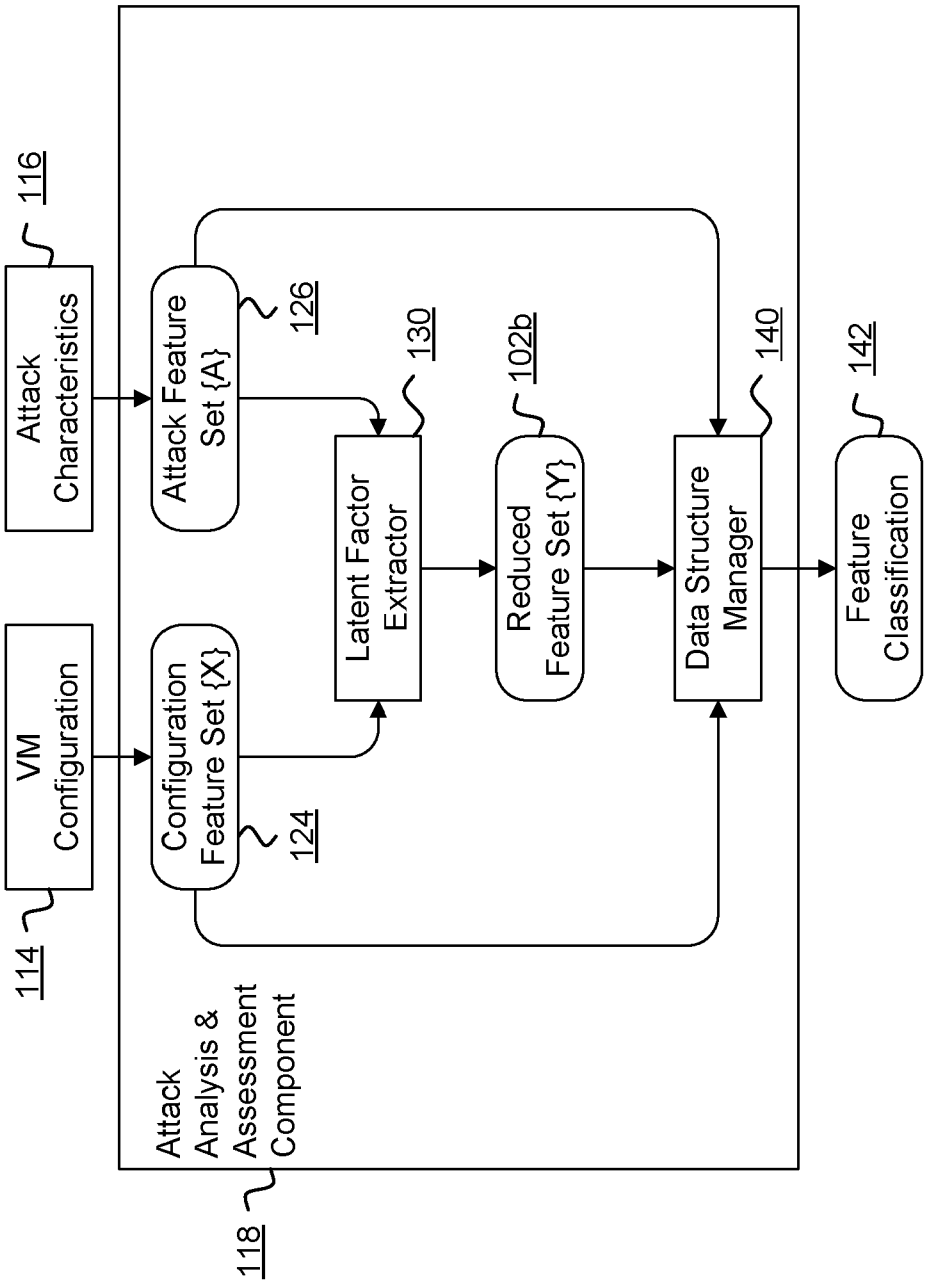

Attack Characteristics
116
Attack Feature Set {A}
126
VM Configuration
114
Configuration Feature Set {X}
124
Latent Factor Extractor
130
Reduced Feature Set {Y}
102b
Data Structure Manager
140
Feature Classification
142
Attack Analysis & Assessment Component
118

FIGURE 6

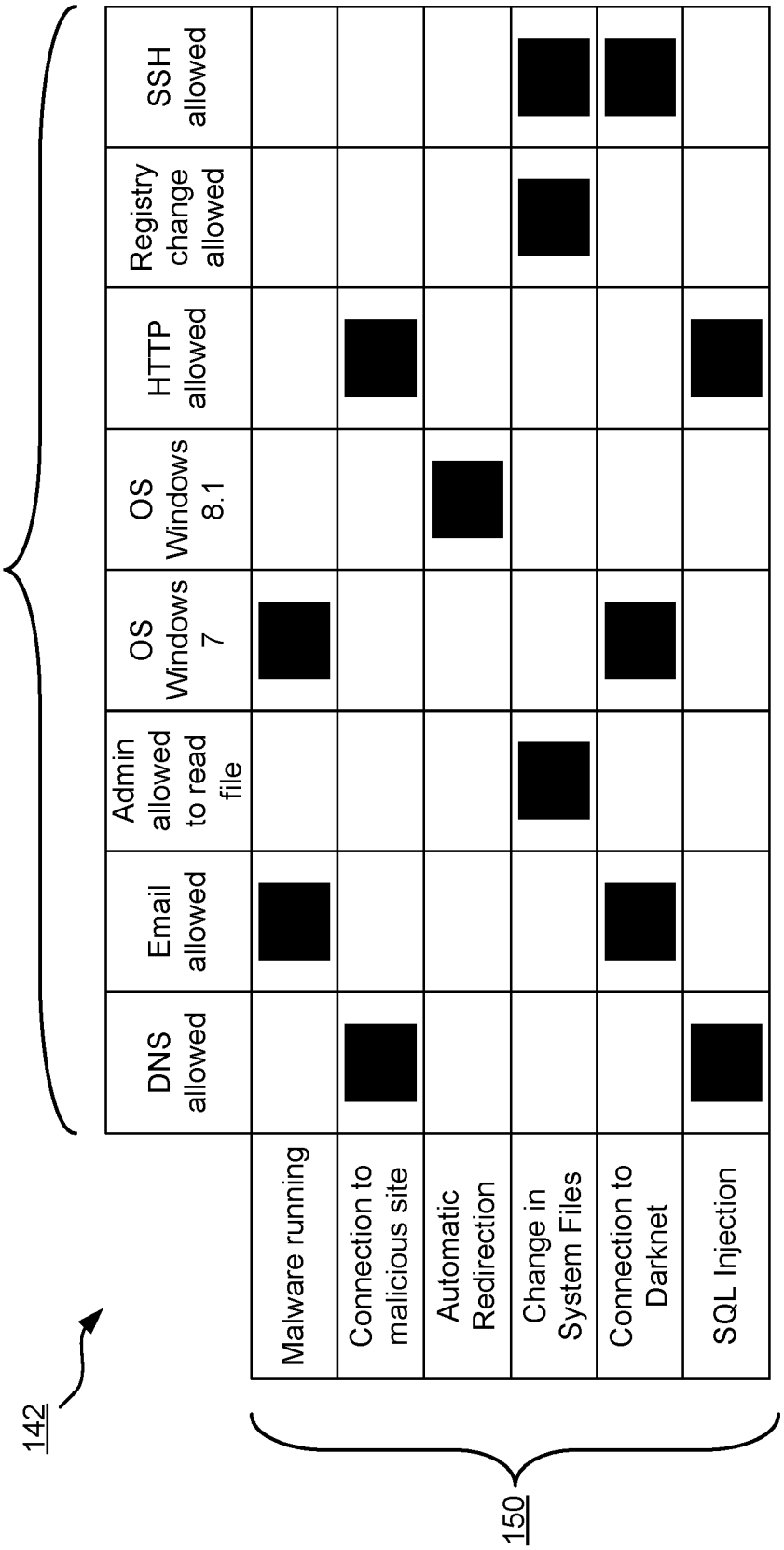

142
150
152
Malware running
Connection to malicious site
Automatic Redirection
Change in System Files
Connection to Darknet
SQL Injection
DNS allowed
Email allowed
Admin allowed to read file
OS Windows 7
OS Windows 8.1
HTTP allowed
Registry change allowed
SSH allowed

FIGURE 7

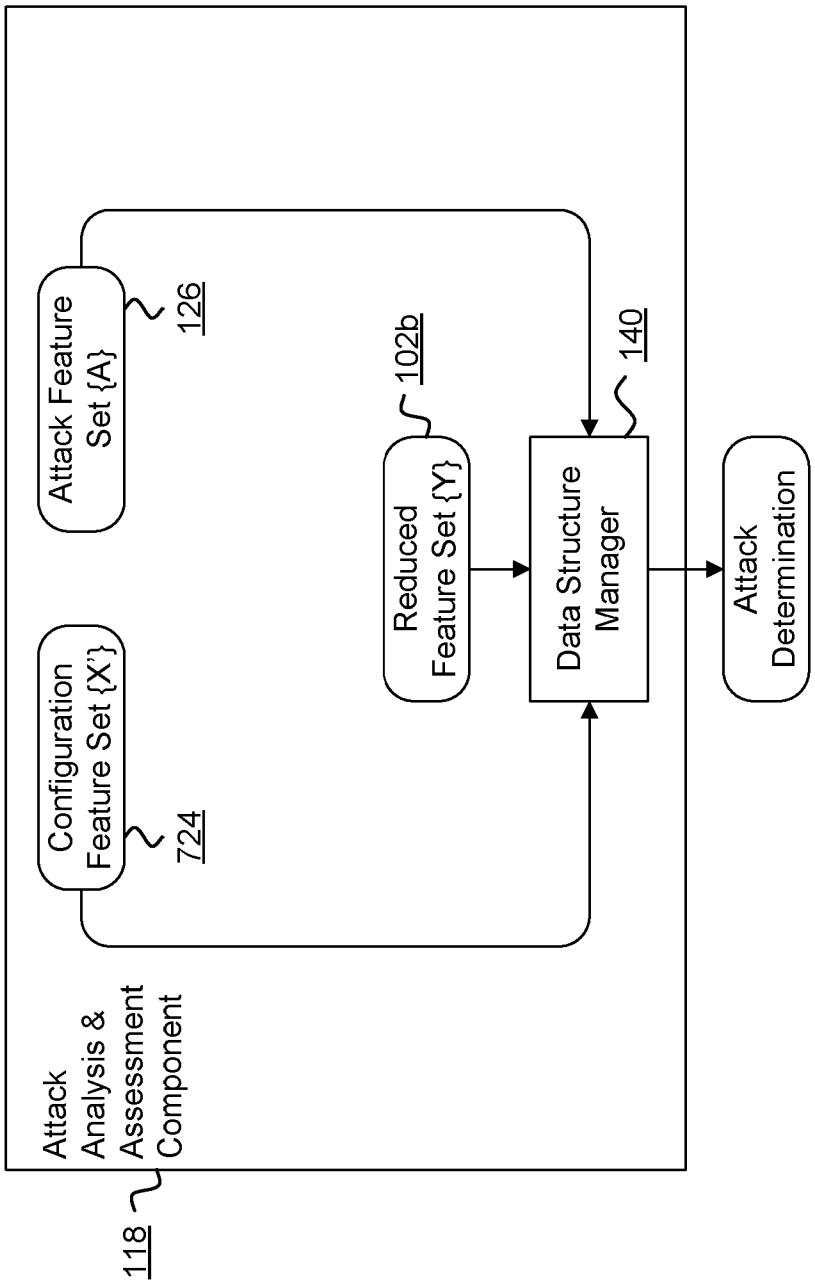

Attack Analysis & Assessment Component
118
Configuration Feature Set {X'}
724
Attack Feature Set {A}
126
Reduced Feature Set {Y}
102b
Data Structure Manager
140
Attack Determination

FIGURE 8

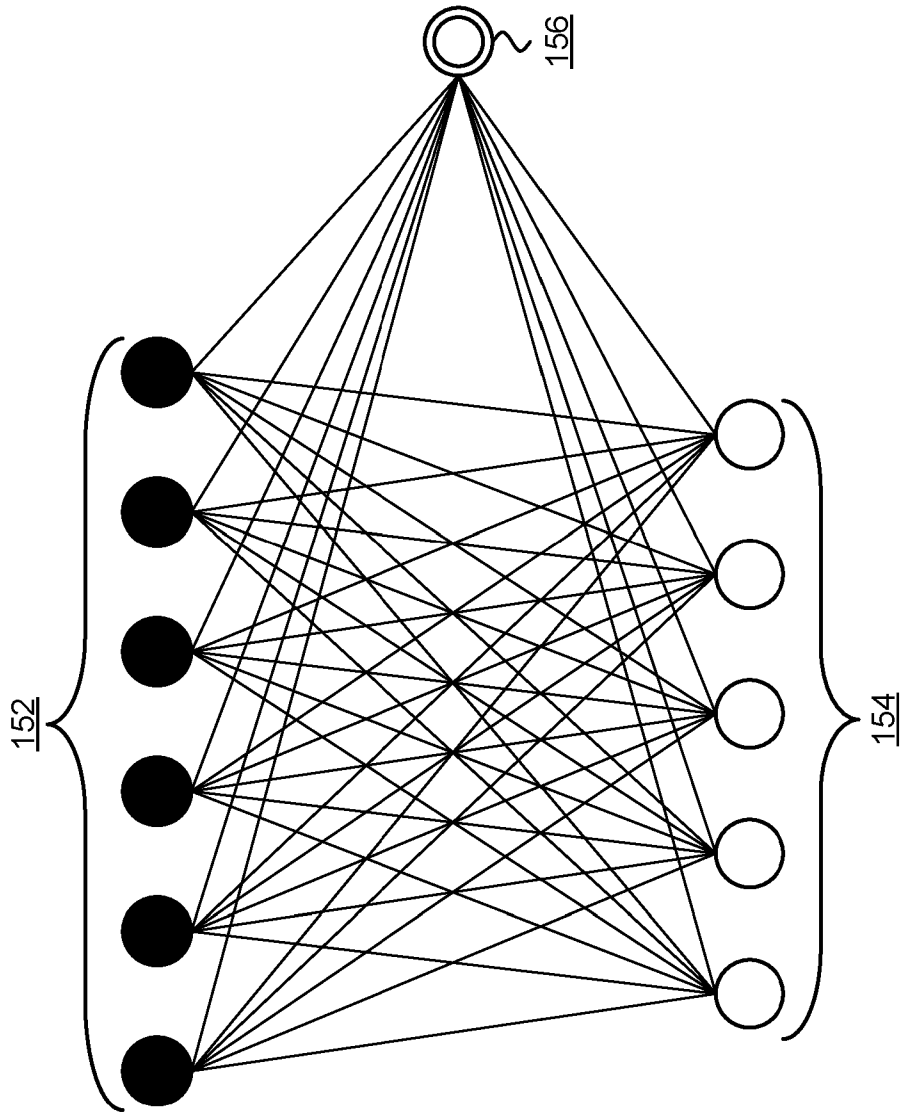


FIGURE 9

VM ID	VM Configuration Features	Attack Features
VM1	DNS allowed; HTTP allowed; Registry Change Allowed; OS Windows 8	Connection to malicious site; Automatic redirection
VM1	E-mail allowed; HTTP allowed; Admin allowed to read file; OS Windows 7	Malware running; Change in system files

Set	Features
{X}	{DNS allowed; HTTP allowed; Registry Change Allowed; E-mail allowed; Admin allowed to read file; OS Window 7; OS Windows 8}
{A}	{Connection to malicious site; Automatic redirection; Malware running; Change in system files}

FIGURE 10

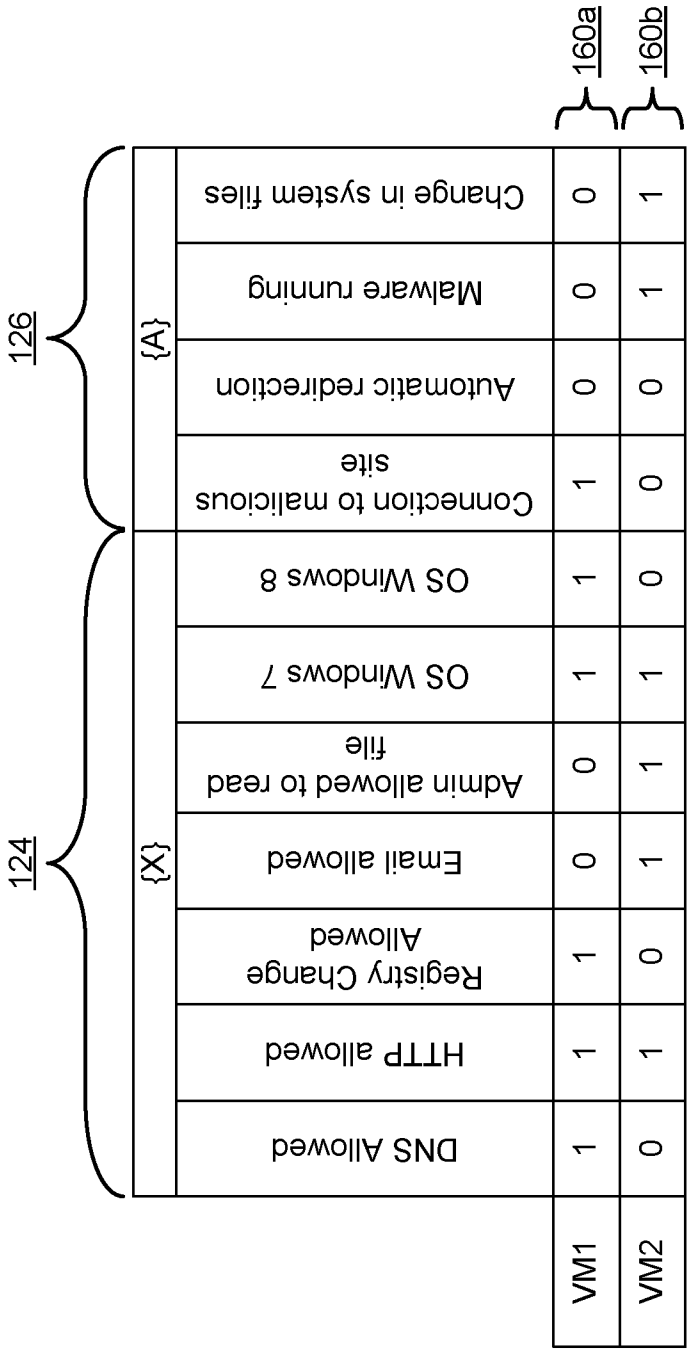

124
126
{X}
{A}
DNS Allowed
HTTP allowed
Registry Change Allowed
Email allowed
Admin allowed to read file
OS Windows 7
OS Windows 8
Connection to malicious site
Automatic redirection
Malware running
Change in system files
VM1
VM2
160a
160b
1 1 0 1 1 1 0 1 0 0 0 1
0 1 1 0 1 1 1 0 0 0 1 0

FIGURE 11

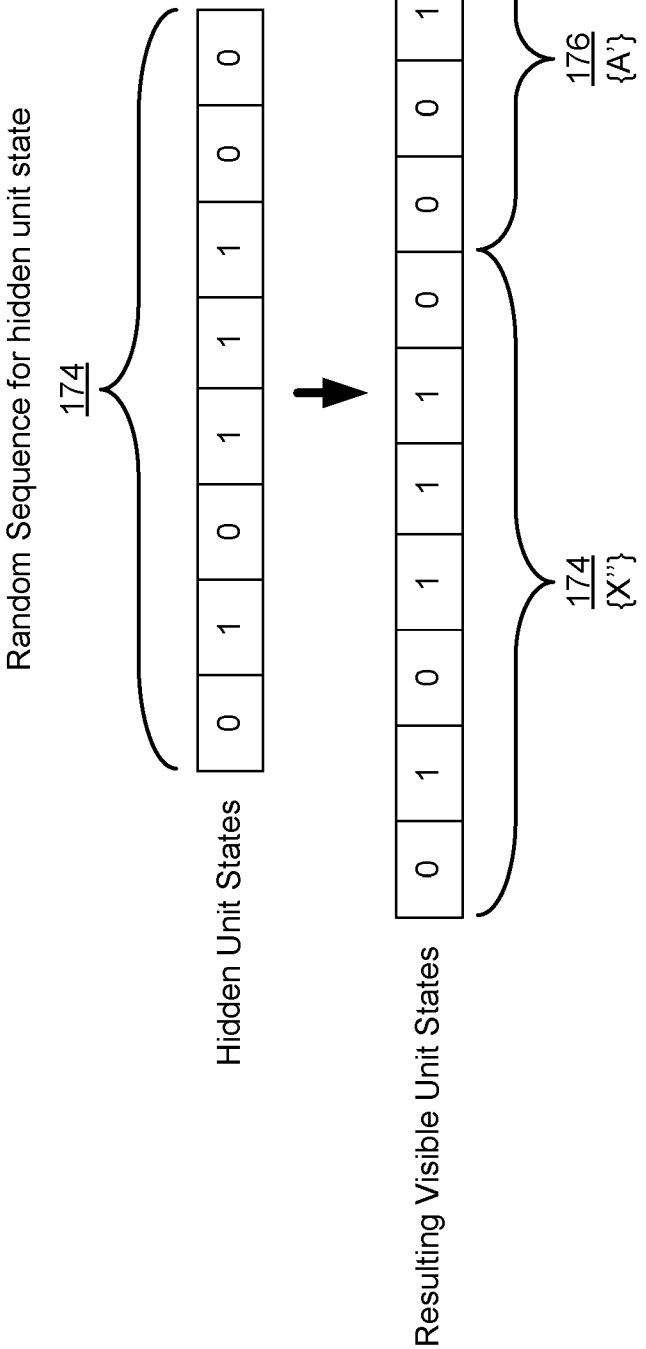

Random Sequence for hidden unit state
174
0 1 0 1 1 1 0 0
Hidden Unit States
Resulting Visible Unit States
0 1 0 1 1 1 0 0 0 1 1
174
{X''}
176
{A'}

FIGURE 12

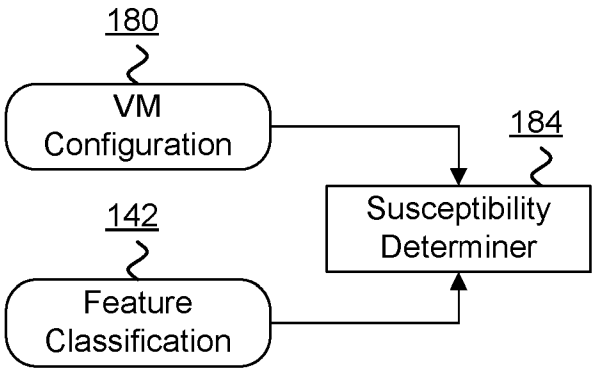


FIGURE 13

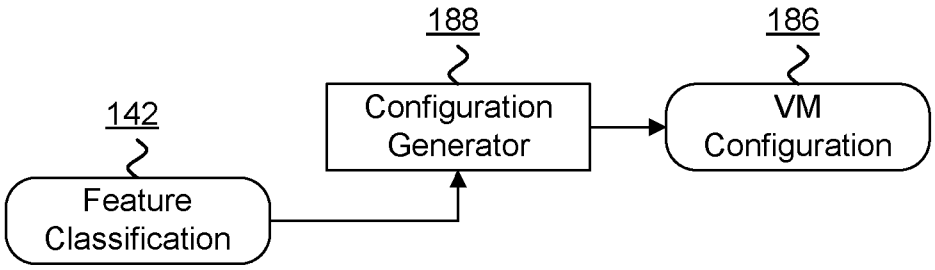


FIGURE 14

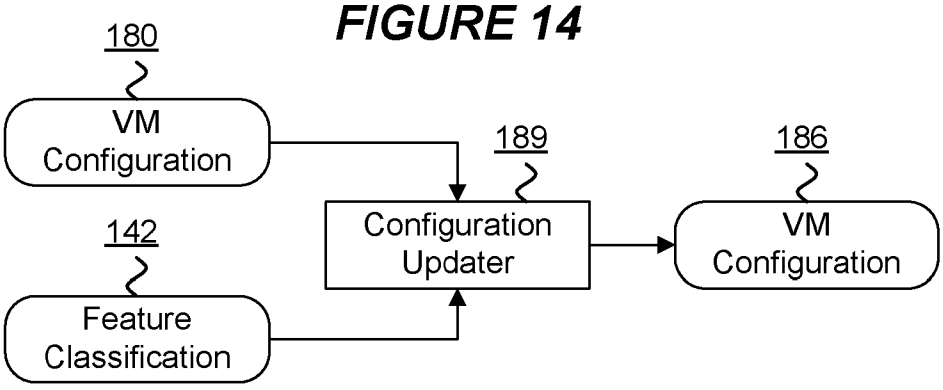


FIGURE 15

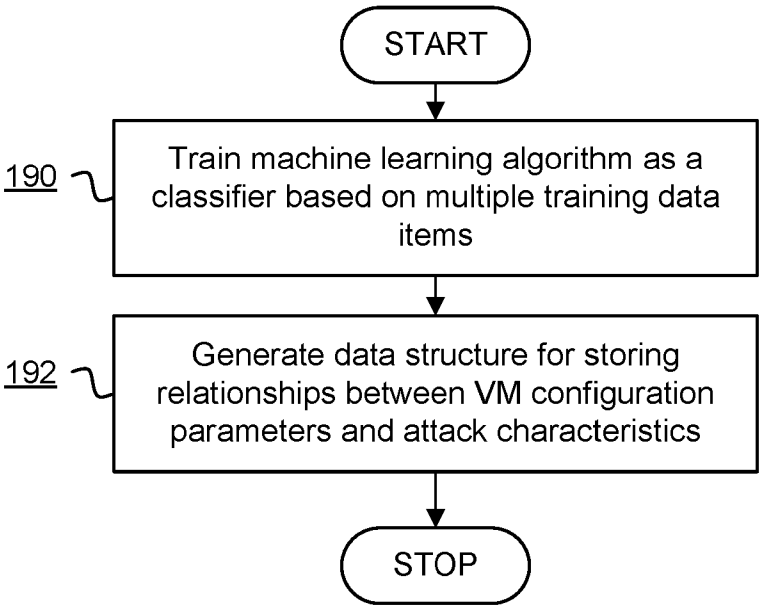


FIGURE 16

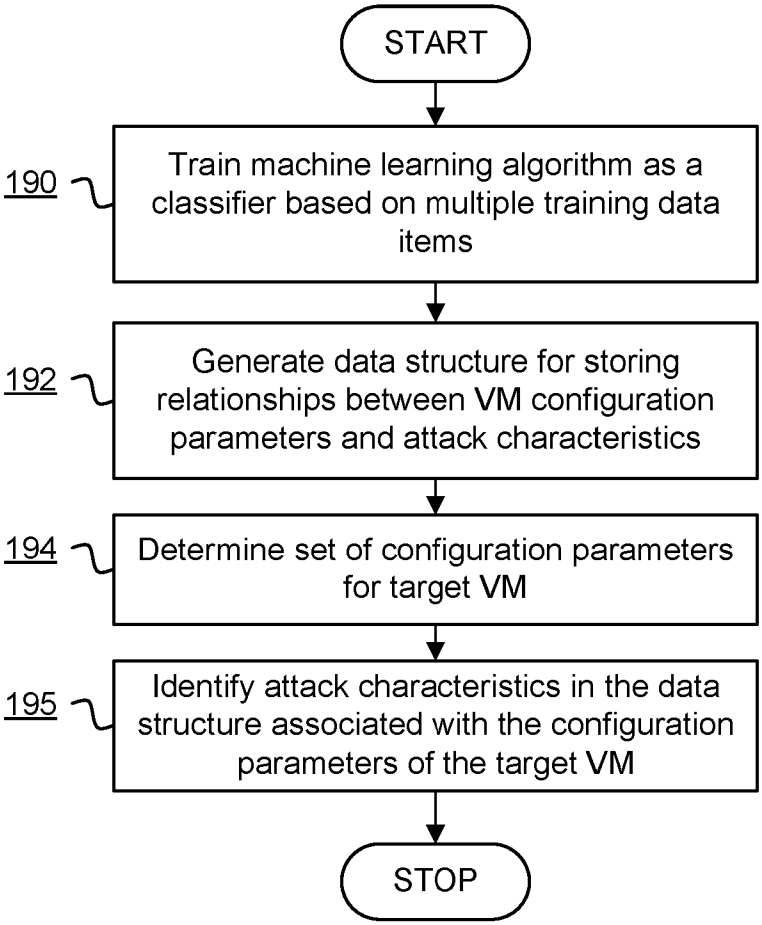


FIGURE 17

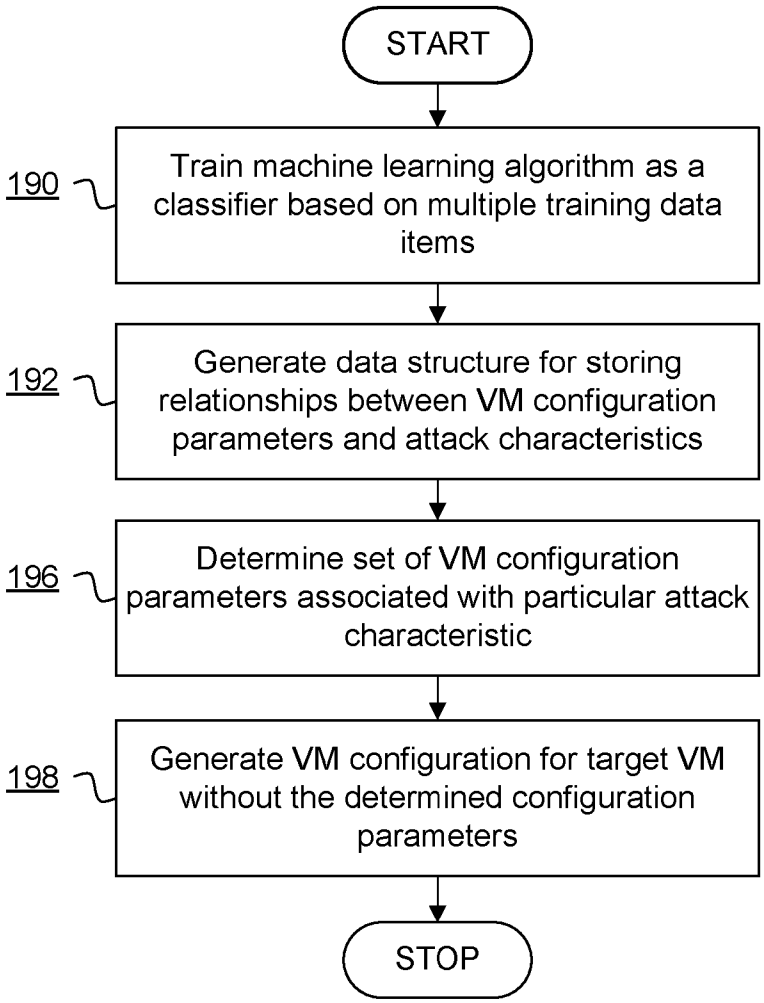

START
190
Train machine learning algorithm as a classifier based on multiple training data items
192
Generate data structure for storing relationships between VM configuration parameters and attack characteristics
196
Determine set of VM configuration parameters associated with particular attack characteristic
198
Generate VM configuration for target VM without the determined configuration parameters
STOP

FIGURE 18

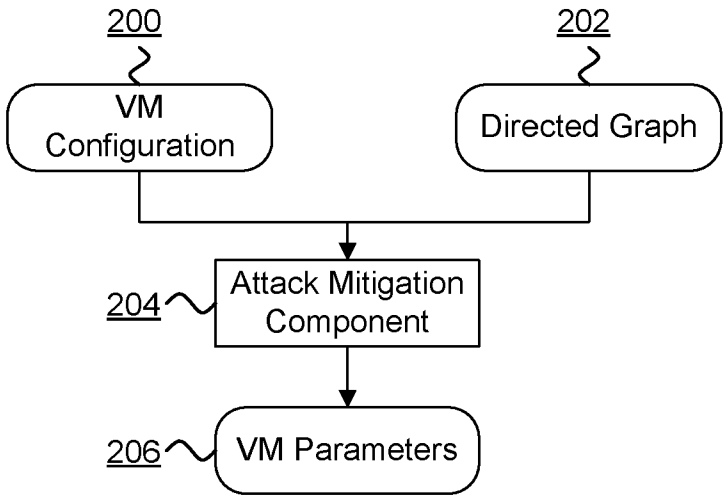


FIGURE 19

	Email allowed	Windows 8 OS	Windows 10 OS	FTP allowed	DNS redirection permitted	HTTP allowed	Directory write access permitted	Database Server Admin Access Permitted	Admin Login Permitted	Superuser Privilege Permitted	Firewall Installed
Malware	■		■	■	■	■	■		■	■	

FIGURE 20

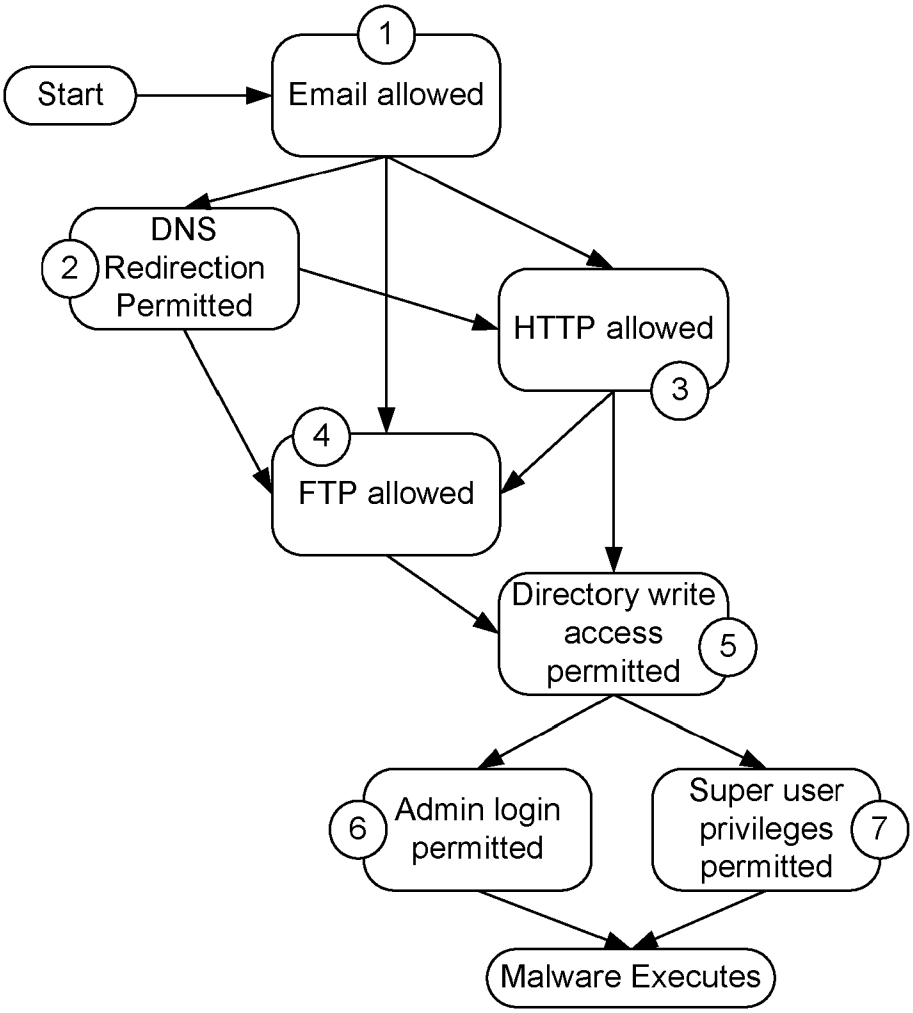


FIGURE 21

Email allowed	Windows 8 OS	Windows 10 OS	FTP allowed	DNS redirection permitted	HTTP allowed	Directory write access permitted	Database Server Admin Access Permitted	Admin Login Permitted	Superuser Privilege Permitted	Firewall Installed
■		■		■	■	■			■	

FIGURE 22

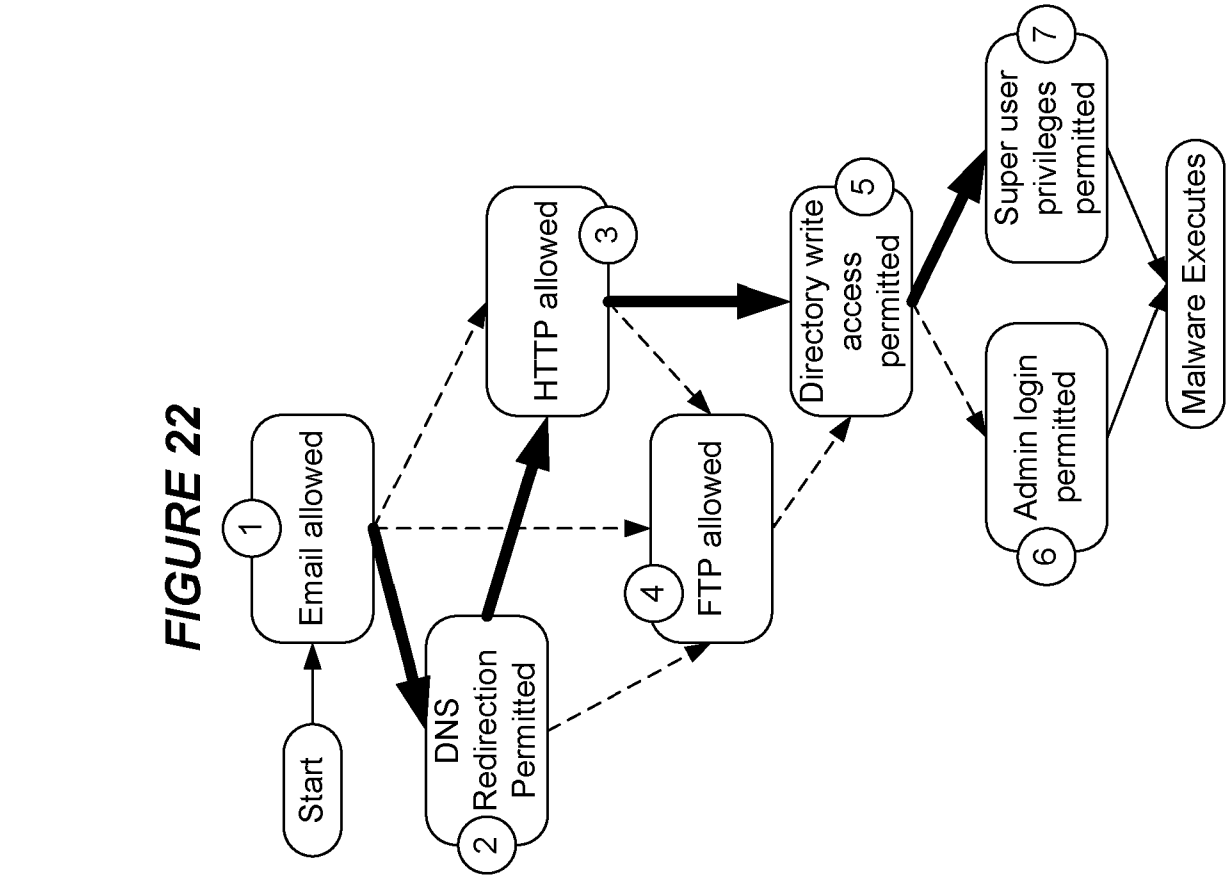


FIGURE 23

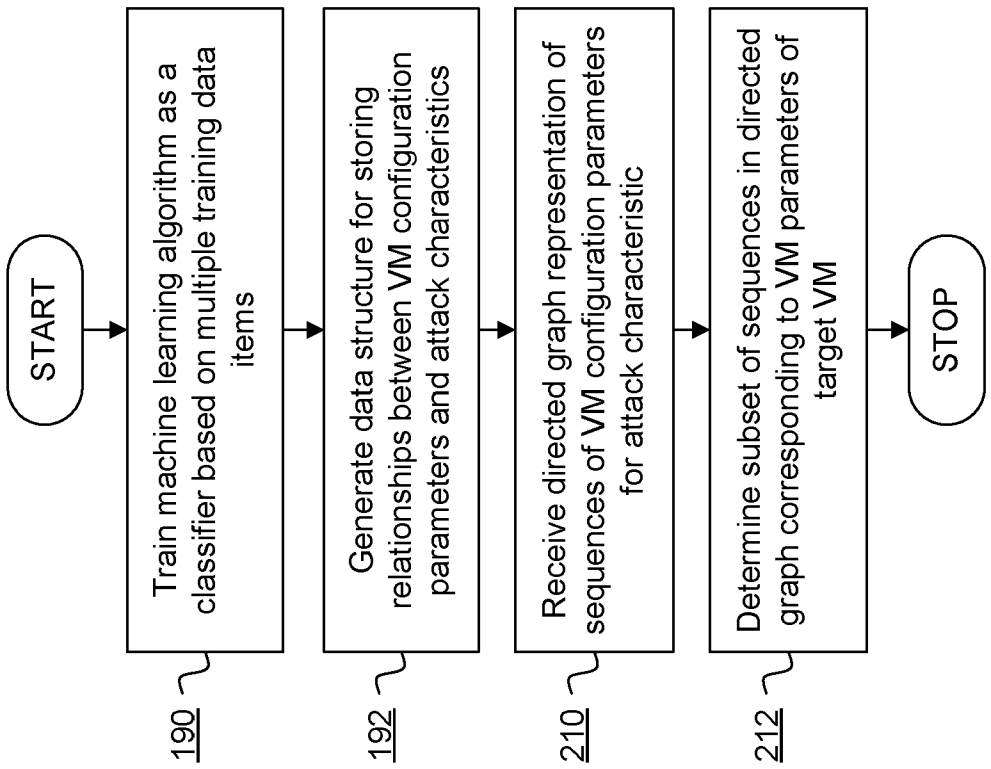


FIGURE 24

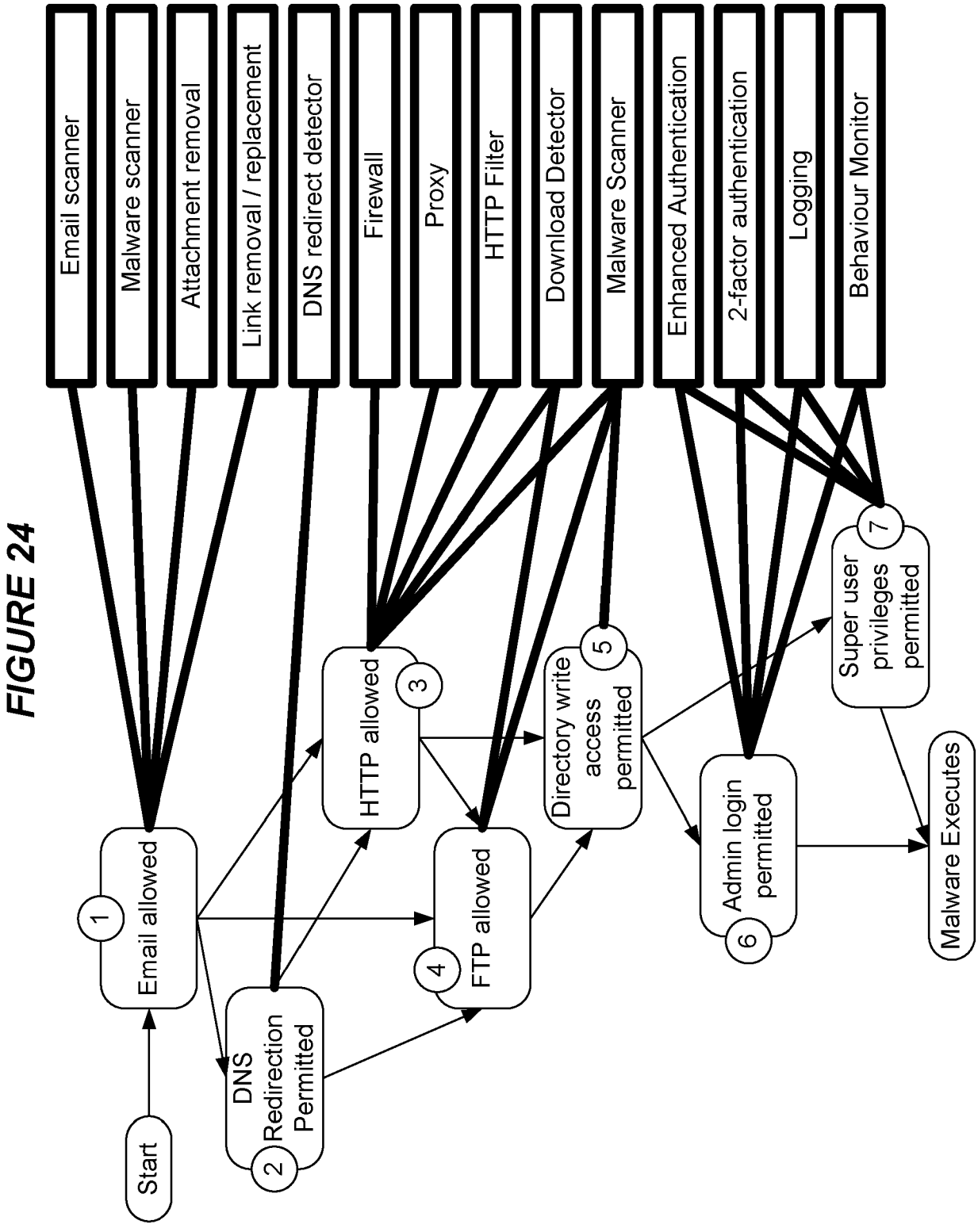

Email scanner
Malware scanner
Attachment removal
Link removal / replacement
DNS redirect detector
Firewall
Proxy
HTTP Filter
Download Detector
Malware Scanner
Enhanced Authentication
2-factor authentication
Logging
Behaviour Monitor
Start
Email allowed
1
DNS Redirection Permitted
2
HTTP allowed
3
FTP allowed
4
Directory write access permitted
5
Admin login permitted
6
Super user privileges permitted
7
Malware Executes

FIGURE 25

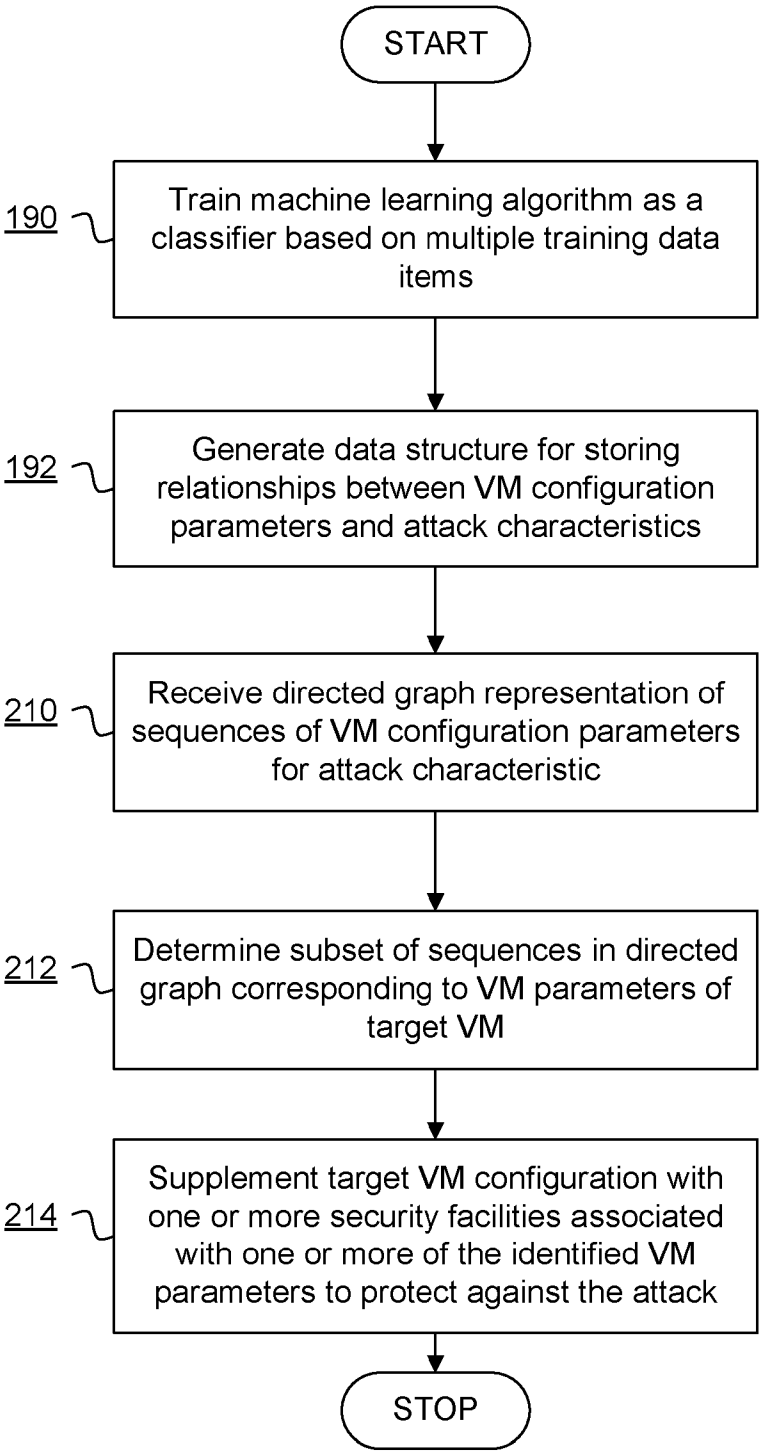

START
190
Train machine learning algorithm as a classifier based on multiple training data items
192
Generate data structure for storing relationships between VM configuration parameters and attack characteristics
210
Receive directed graph representation of sequences of VM configuration parameters for attack characteristic
212
Determine subset of sequences in directed graph corresponding to VM parameters of target VM
214
Supplement target VM configuration with one or more security facilities associated with one or more of the identified VM parameters to protect against the attack
STOP

FIGURE 26

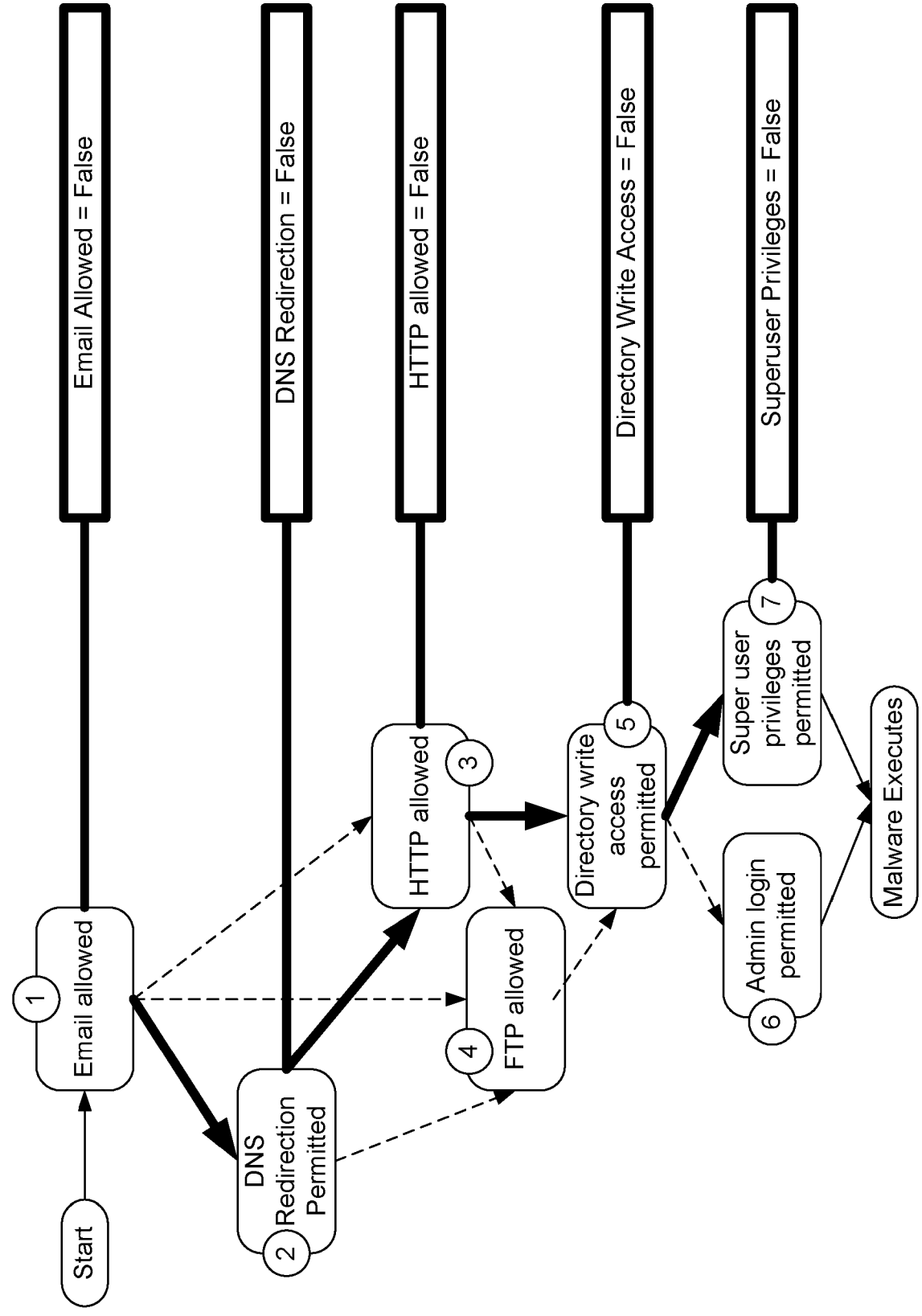

Email Allowed = False
DNS Redirection = False
HTTP allowed = False
Directory Write Access = False
Superuser Privileges = False
Start
1
Email allowed
2
DNS Redirection Permitted
3
HTTP allowed
4
FTP allowed
5
Directory write access permitted
6
Admin login permitted
7
Super user privileges permitted
Malware Executes

FIGURE 27

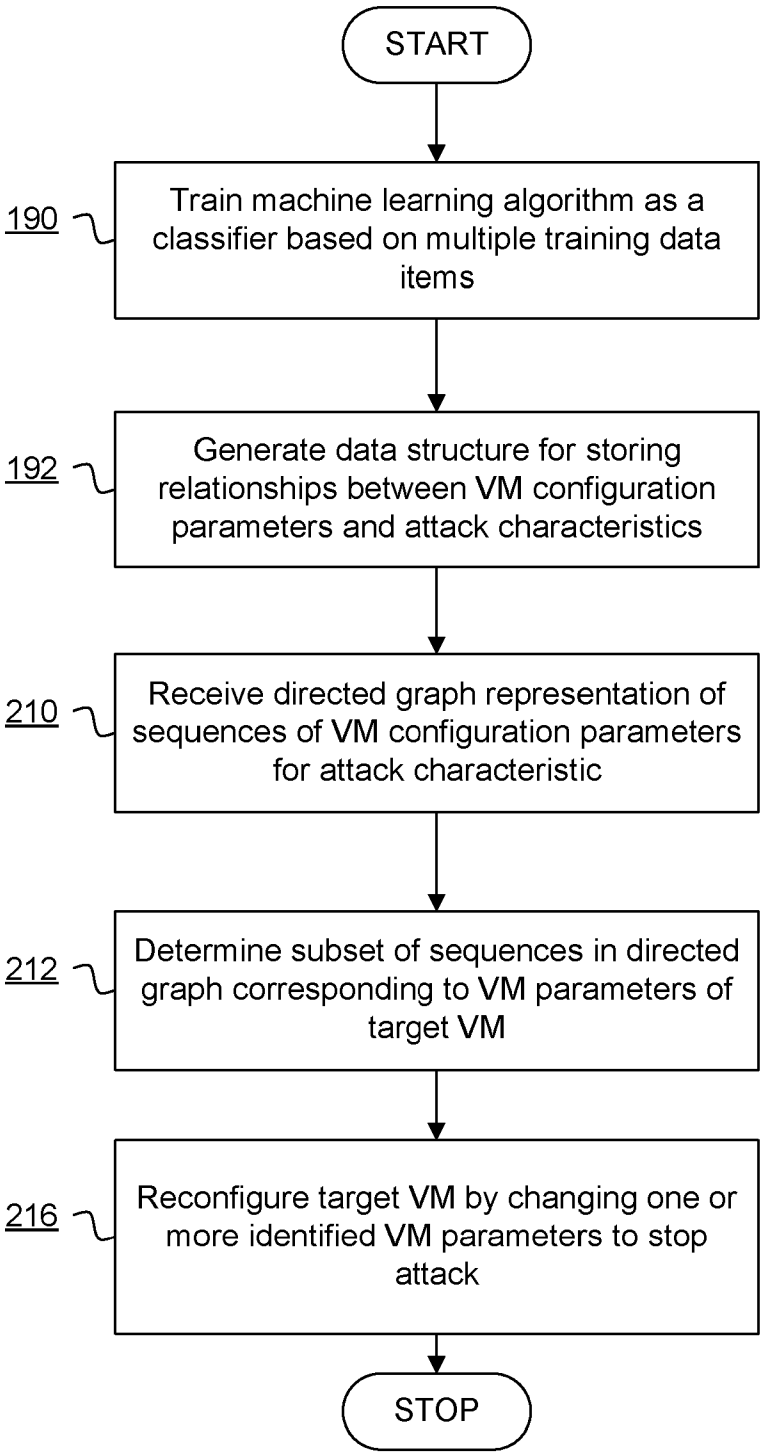

START
190
Train machine learning algorithm as a classifier based on multiple training data items
192
Generate data structure for storing relationships between VM configuration parameters and attack characteristics
210
Receive directed graph representation of sequences of VM configuration parameters for attack characteristic
212
Determine subset of sequences in directed graph corresponding to VM parameters of target VM
216
Reconfigure target VM by changing one or more identified VM parameters to stop attack
STOP

FIGURE 28

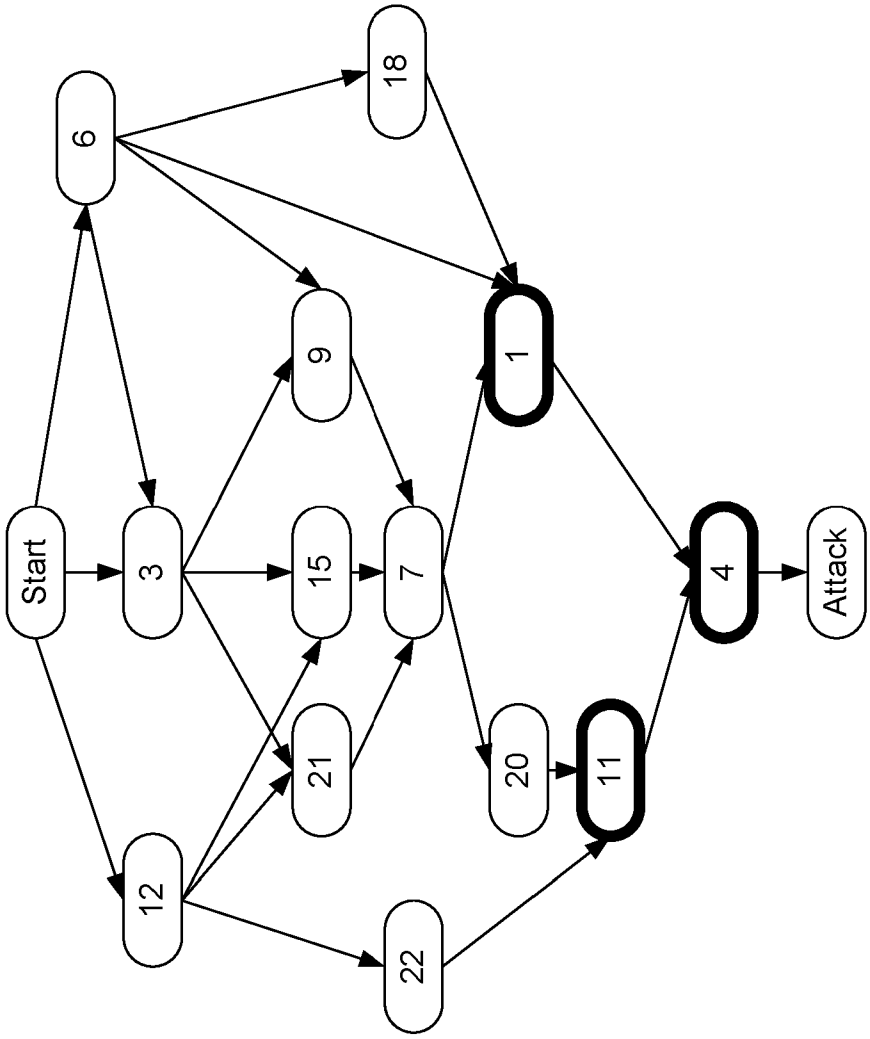


FIGURE 29

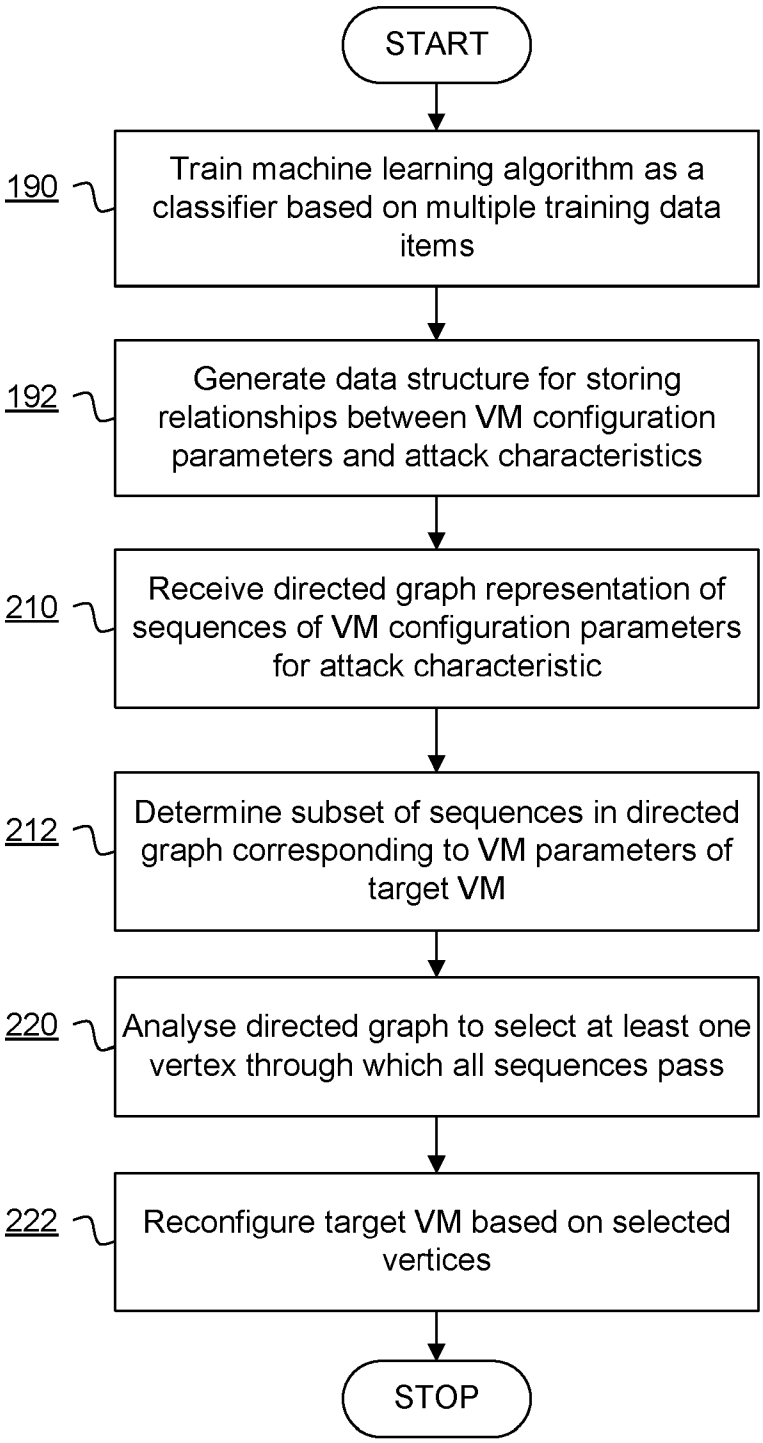
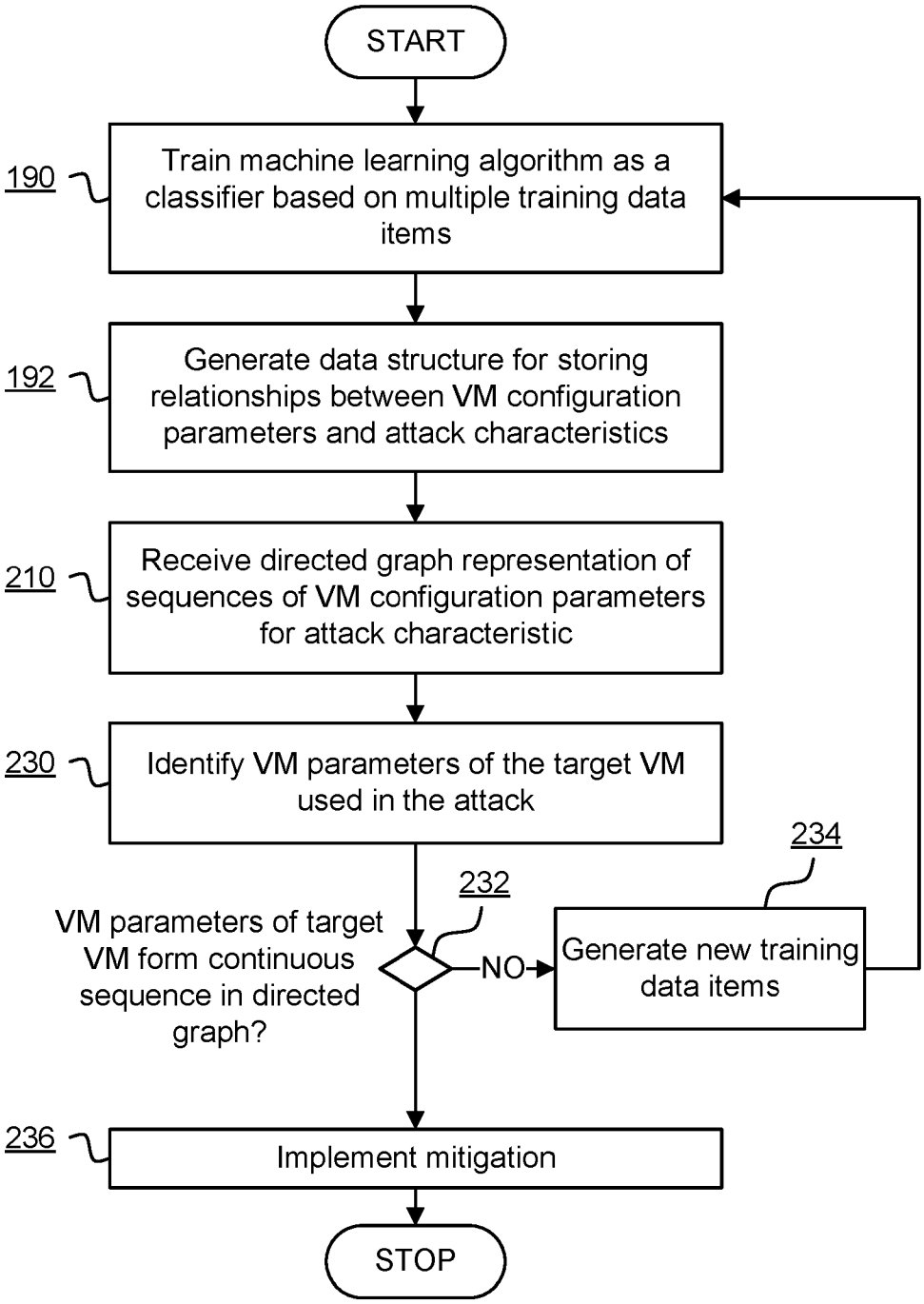

START
190
Train machine learning algorithm as a classifier based on multiple training data items
192
Generate data structure for storing relationships between VM configuration parameters and attack characteristics
210
Receive directed graph representation of sequences of VM configuration parameters for attack characteristic
212
Determine subset of sequences in directed graph corresponding to VM parameters of target VM
220
Analyse directed graph to select at least one vertex through which all sequences pass
222
Reconfigure target VM based on selected vertices
STOP

FIGURE 30


START
190
Train machine learning algorithm as a classifier based on multiple training data items
192
Generate data structure for storing relationships between VM configuration parameters and attack characteristics
210
Receive directed graph representation of sequences of VM configuration parameters for attack characteristic
230
Identify VM parameters of the target VM used in the attack
232
VM parameters of target VM form continuous sequence in directed graph?
NO
234
Generate new training data items
236
Implement mitigation
STOP

Machine Learning for Attack Mitigation in Virtual Machines

The present invention relates to the detection of computer security threats.

Computer systems such as virtual machines (VMs) executing in virtualised computing environments (VCEs) such as cloud computing environments may look like any physical, networked or standalone computer system such as a personal computing device and are therefore equally susceptible to any kind of cyber-attack if not properly protected. For example, a VM may become infected by malware communicated via network communication or when a user opens an infected email attachment or connects to malicious websites. Once a VM is infected it may become part of a group of collectively controlled systems such as a "botnet" for use by an adversary or hacker to coordinate further cyber-attacks on other systems communicatively connected to compromised systems, such as via the Internet.

Thus there is a need to protect such virtualised computer systems from such attacks.

The present invention accordingly provides, in a first aspect, a computer implemented method to mitigate a security attack against a target virtual machine (VM) in a virtualised computing environment, the target VM having a target VM configuration including configuration parameters, and the security attack exhibiting a particular attack characteristic, the method comprising: training a machine learning algorithm as a classifier based on a plurality of training data items, each training data item corresponding to a training VM and including a representation of parameters for a configuration of the training VM and a representation of characteristics of security attacks for the training VM; generating a first data structure for storing one or more relationships between VM configuration parameters and attack characteristics, wherein the first data structure is generated by sampling the trained machine learning algorithm to identify the relationships; receiving a second data structure storing a directed graph representation of one or more sequences of VM configuration parameters for achieving the particular attack characteristic of the security attack, the VM parameters in the directed graph being determined based on the first data structure; identifying VM parameters of the target VM used in the security attack; in response to a determination that the VM parameters of the target VM do not form a continuous sequence in the directed graph, triggering the steps of: a) generating new training data items for one or more training VMs including at least one VM being subject to the attack; b) repeating the training and generating steps so as to generate a new first data structure of relationships; and c) receiving a new second data structure based on the new first data structure.

Preferably each of the attack characteristics has associated a protective measure, the method further comprising, in response to the identification of an attack characteristic to which the target VM is susceptible, implementing the protective measure so as to protect the VM from attacks having the attack characteristic.

Preferably the method further comprises: identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph of the new second data structure corresponding to VM parameters of the target VM; and supplementing the target VM configuration with a security facility associated with at least one of the identified VM parameters so as to protect the target VM from the attack.

Preferably the method further comprises: identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph of the new second data structure corresponding to VM parameters of the target VM; and reconfiguring the target VM by changing at least one of the identified VM parameters so as to stop the attack.

Preferably the method further comprises: identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph corresponding to VM parameters of the target VM; analysing the second data structure to select one or more vertices of the directed graph each indicating a VM parameter, wherein all sequences of VM configuration parameters for achieving the attack pass through at least one of the vertices; reconfiguring the target VM by changing VM parameters indicated in each of the identified vertices, wherein the vertices are selected to include VM parameters according to predetermined criteria.

Preferably the predetermined criteria are defined to require a minimum number of VM parameters.

Preferably each vertex in the directed graph has associated a predetermined weighting based on a VM parameter indicated by the vertex and wherein the predetermined criteria are defined to require that each selected vertex meets a predetermined condition in relation to their associated weighting.

Preferably each vertex in the directed graph has associated a predetermined weighting based on a VM parameter indicated by the vertex and wherein the predetermined criteria are defined to require that a total of all weightings of all selected vertices meets a predetermined condition.

Preferably the predetermined condition is a maximum weight.

Preferably the weighting is an indication of importance of a VM parameter such that parameters that are more important have more impact on the overall weight.

Preferably the machine learning algorithm is a restricted Boltzmann machine.

Preferably the characteristics of security attacks include an indication of the consequence of a security attack executing in the training VM.

Preferably each training data item comprises a vector of binary values indicating each indicating a presence or absence of a configuration feature and an attack characteristic of a corresponding training VM.

Preferably the data structure is a matrix data structure for mapping VM configuration parameters against attack characteristics.

Preferably the restricted Boltzmann machine includes a plurality of hidden units and a plurality of visible units, and sampling the trained machine learning algorithm includes generating sample inputs for the hidden units to determine values of the visible units.

Preferably each generated sample input is a vector of binary values wherein each binary value is determined using a randomisation algorithm.

Preferably each protective measure is a configuration parameter or a change to a configuration parameter for a VM to protect against an attack characteristic.

The present invention accordingly provides, in a second aspect, a computer system including a processor and memory storing computer program code for performing the steps of the method set out above.

The present invention accordingly provides, in a third aspect, a computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of the method set out above.

Embodiments of the present invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a block diagram illustrating computer systems executing in virtualised computing environments under control of a botnet controller;

Figure 2 is a block diagram of a virtualised computing environment in accordance with embodiments of the present invention;

Figure 3 is a block diagram of a computer system suitable for the operation of embodiments of the present invention;

Figure 4 illustrates an arrangement of an attack analysis and assessment component in accordance with embodiments of the present invention;

Figure 5 is a block diagram of the attack analysis and assessment component of Figure 4 in accordance with embodiments of the present invention;

Figure 6 illustrates a matrix mapping VM configuration features against attack features in an exemplary embodiment of the present invention;

Figure 7 illustrates a further arrangement of the attack analysis and assessment component of Figure 4 in accordance with embodiments of the present invention;

Figure 8 illustrates a restricted Boltzmann machine for use in exemplary embodiments of the present invention;

Figure 9 illustrates the determination of an aggregate set of VM configuration features {X} and an aggregate set of attack features {A} in an exemplary embodiment of the present invention;

Figure 10 illustrates exemplary input vectors for a restricted Boltzmann machine based on the features of Figure 9;

Figure 11 illustrates states of hidden and visible units of a restricted Boltzmann machine as part of a sampling process in an exemplary embodiment of the present invention;

Figure 12 is a component diagram illustrating an arrangement including a susceptibility determiner component for determining whether a target VM is susceptible to a security attack based on a pre-existing VM configuration for the target VM in accordance with some embodiments of the present invention;

Figure 13 is a component diagram illustrating an arrangement including a configuration generator for determining a configuration of a target VM to protect against a security attack exhibiting a particular attack characteristic in accordance with some embodiments of the present invention;

Figure 14 is a component diagram illustrating an arrangement including a configuration updater for determining a configuration of a VM to protect against a security attack exhibiting a particular attack characteristic and updating a pre-existing VM configuration for a target VM to protect against attacks having the attack characteristic based

on the determined configuration in accordance with some embodiments of the present invention;

Figure 15 is a flowchart of a method to generate a classification scheme for configuration parameters of VMs in accordance with some embodiments of the present invention;

Figure 16 is a flowchart of a method to determine whether a target VM is susceptible to a security attack in accordance with some embodiments of the present invention;

Figure 17 is a flowchart of a method to determine a configuration of a target VM to protect against a security attack exhibiting a particular attack characteristic in accordance with some embodiments of the present invention;

Figure 18 is a component diagram of an arrangement for attack mitigation in accordance with embodiments of the present invention;

Figure 19 illustrates an exemplary entry in a feature classification data structure for a malware attack characteristic in accordance with an exemplary embodiment of the present invention;

Figure 20 illustrates a data structure storing a directed graph representation of sequences of VM configuration parameters for the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention;

Figure 21 illustrates states of an exemplary configuration of a VM in accordance with the VM configuration parameters of Figure 19 and in accordance with an exemplary embodiment of the present invention;

Figure 22 illustrates a subset of sequences in the directed graph of Figure 20 corresponding to VM parameters of the VM of Figure 21 in accordance with an exemplary embodiment of the present invention;

Figure 23 is a flowchart of a method to identify configuration parameters of a target VM used in a security attack against the target VM in accordance with embodiments of the present invention;

Figure 24 illustrates exemplary security facilities that can be employed to mitigate the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention;

Figure 25 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention;

Figure 26 illustrates exemplary VM configuration parameter changes that can be employed to mitigate the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention;

Figure 27 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention;

Figure 28 illustrates a data structure storing a directed graph representation of sequences of VM configuration parameters for an attack characteristic in accordance with an exemplary embodiment of the present invention;

Figure 29 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention; and

Figure 30 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention.

One example of an attack employing compromised VMs is coordinated by a "botnet controller" – known as "Command and Control" (C&C) – which may control a number of infected machines (any of which may be physical, virtual, cloud-hosted or standalone machines) to launch different kinds of attack. Figure 1 is a block diagram illustrating computer systems 106 executing in VCEs 102a to 102d under control of a botnet controller 100. Figure 1 shows an example scenario where the botnet controller 100 controls a number of VMs 106 (shown hatched) hosted in potentially different VCEs 102a to 102d to launch one or more attacks on a target computer system 108. Such an attack can include a distributed denial of service (DDoS) attack on the target 108. Notably the network communication between infected VMs and the controller 100 may not employ a direct connection and may be routed via other machines including other infected machines.

In order to protect a VM from becoming compromised by a malicious attack and potentially infected and/or recruited into a botnet a user (or system administrator) needs to apply appropriate security measures such as, inter alia, installing up-to-date anti-malware software, configuring firewalls to block suspicious network communication, and/or apply latest security patches for an operating system etc. Additionally, a user must be vigilant when opening emails from unknown sources or accessing data, files or software communicated via a network such as the internet. While such measures can provide protection in general, it may not be sufficient to protect against more sophisticated attacks or zero-day attacks that

are relatively unknown. There is also a lack of security knowledge among many users which can lead to non-optimal configuration of security software (e.g. firewall) or unsafe access to materials via a network (e.g. unsafe browsing, not being aware of unsecure network connections such as non-HTTPS connections, etc.). In particular, for cloud-hosted machines cloud providers frequently employ VM or system templates to assist users in deploying new VMs. Leaving a VM configuration at least partly in a default, template or original state can pose a security risk since a potential adversary may have knowledge of such a default configuration and may be able to exploit any vulnerability in a deployed VM to compromise it.

Embodiments of the present invention seek to addresses the security issues of virtualised computing environments such as cloud computing environments by obtaining configuration and/or security related features from VMs, combining them with detected attack characteristics and/or an absence of attack information and applying a machine learning approach to determine whether or not a particular VM may be susceptible to attack.

Figure 2 is a block diagram of a virtualised computing environment 102 in accordance with embodiments of the present invention and shows an example implementation of an embodiment of the present invention. The arrangement of Figure 2 includes one of potentially many VCEs 102 each hosting one or more infected VMs 106 among a population of VMs 104a to 104c. The virtualised computing environment 102 is a system for executing one or more virtualised computer systems in a local, distributed or hybrid manner. Such virtualisation can be achieved using virtualisation facilities such as one or more hypervisors or the like. Such virtualisation provides a separation between a computer system implementation and physical hardware with which computer systems execute. Such computer systems are typically VMs such as VMs 104a to 104c and VM 106. Distributed or remotely hosted virtualised environments can provide computer systems as VMs for use, access or consumption by consuming entities. An example of such an arrangement is a cloud hosted VCE.

Infected VMs 106 are controlled by a botnet controller 100 such as to launch an attack campaign. Infected VMs 106 can be part of multiple or different botnets, i.e. controlled by different botnet controllers. VCEs may physically be located in different geographical areas, may be managed by a single or more service providers. In each VCE a service provider manages configuration information 110 and security information 112. Configuration information 110 is information relating to a configuration of one or more VMs executing in the VCE 102. The configuration information may be specific to a VM or apply to multiple VMs and includes an identification and/or definition or resources and/or configurations deployed for a VM. For example, via the configuration information 110 configuration parameters of

each VM can be identified including, inter alia: Operating system identification; Network topology; VPN configuration; DNS settings; Email configuration; a Security configuration, e.g. Antivirus, Firewall, etc. Thus the configuration information 110 is suitable for defining one or more VM characteristics 114 for VMs in the VCE.

The security information 112 is information relating to one or more security facilities of the VCE 102 and/or individual VMs deployed therein. In particular, the security information includes information sufficient to determine characteristics of any attack(s) that have occurred in a VM in the VCE such as, inter alia: an indication of the execution of malware; an indication of unauthorised changes to system files; a connection to a known illicit, malicious or unsecure network such as "darknet"; and other such attack characteristics as will be apparent to those skilled in the art and that can be identified and recorded by security services such as security software. For example, the security information 112 can include information including, inter alia, information from VCE-wide security sensors, i.e. IDS (Intrusion Detection System), Firewall, Web-Proxy, etc. The security information 112 provides characteristics 116 or features of successful attacks on any VM within the VCE, such as: Attack type, e.g. Virus, Trojan, etc.; Attack method, e.g. SQL injection, XSS, etc.; IP domain; Used ports, protocols or user agents, etc. Thus the security information 112 is suitable for defining one or more attack characteristics 116 for VMs in the VCE. In some embodiments the security information 112 is specific to each of one or more VMs 104, 106 and can be obtained, stored, handled and/or managed by such VMs individually.

Figure 3 is a block diagram of a computer system suitable for the operation of embodiments of the present invention. A central processor unit (CPU) 302 is communicatively connected to a storage 304 and an input/output (I/O) interface 306 via a data bus 308. The storage 304 can be any read/write storage device such as a random access memory (RAM) or a non-volatile storage device. An example of a non-volatile storage device includes a disk or tape storage device. The I/O interface 306 is an interface to devices for the input or output of data, or for both input and output of data. Examples of I/O devices connectable to I/O interface 306 include a keyboard, a mouse, a display (such as a monitor) and a network connection.

Figure 4 illustrates an arrangement of an attack analysis and assessment component 118 in accordance with embodiments of the present invention. The attack analysis and assessment component of Figure 4 is a hardware, software, firmware or combination component for the analysis of the attack characteristics 116 and the configuration characteristics 114 to determine if a VM is susceptible to attack. Thus the attack analysis and assessment component 118 is operable to analyse configuration characteristics 114 and

attack characteristics 116 and employs a feature extraction mechanism, such as latent factor extraction by machine learning, to determine associations between configuration characteristics 114 and attack characteristics 116. Further, in some embodiments the attack analysis and assessment component 118 is operable to determine one or more attack characteristics for attacks to which a particular VM configuration is vulnerable based on the identified latent factors. Further, in some embodiments, the attack analysis and assessment component 118 is operable to determine one or more recommendations for VM configuration to mitigate attacks having one or more attack characteristics.

As illustrated in Figure 4 both configuration characteristics 114 and attack characteristics 116 are received or accessed by the attack analysis and assessment component 118 as input. The attack analysis and assessment component 118 produces a set of one or more associations between these characteristics following a learning phase. The inputs may come from multiple VCEs such as VCEs managed by a single cloud provider. Subsequently the associations determined by the attack analysis and assessment component 118 can be employed to determine whether or not a VM with particular configuration is susceptible to an attack having certain attack characteristics. Yet further the associations can be employed to one or more VM configurations suitable for mitigating a particular type of attack.

Figure 5 is a block diagram of the attack analysis and assessment component 118 of Figure 4 in accordance with embodiments of the present invention. The attack analysis and assessment component 118 includes a latent factor extractor 130 and a data structure manager 140, each of which is a software, hardware, firmware or combination component.

The latent factor extractor 130 is a component for identifying latent factors in a set of binary vectors such as a machine learning algorithm. For example, the latent factor extractor 130 can employ a restricted Boltzmann machine as described below. Latent factors (or latent variables) are features that are not directly observed in the binary vectors but that can be inferred such as through a mathematical model from the binary vectors. In particular, latent factors can be used to identify associations between the elements in binary vectors by, for example, categorising binary vectors.

The data structure manager 140 is a component for generating a data structure as a feature classification 142 that classifies latent factors to identify and recognise associations between aspects of the latent factors as will be explained in detail below.

The attack analysis and assessment component 118 receives or accesses configuration characteristics 114 and attack characteristics 116 for each of a plurality of VMs to generate each of a configuration feature set {X} 124 and an attack feature set {A} 126 respectively.

Configuration feature set {X} consists of elements each corresponding to a configuration feature of a VM. Similarly, attack feature set {A} consists of elements each corresponding to a feature of a successful attack against the VM. For each VM the configuration features {X} and attack features {A} are combined together as input to the latent factor extractor 130. The combine sets {{X}, {A}} for each of multiple VMs are used as training data for the latent factor extractor 130. Following all training based on input sets {X} and {A} for multiple VMs the latent factor extractor 130 generates, as an output, a reduced set of features {Y} representing learned underlying latent factors. Notably, the features set {Y} is not necessarily a subset of features in all of the feature sets {X}.

The feature sets {X}, {A} and {Y} are subsequently used by the data structure generator 140 to generate a data structure classifying configuration features, i.e. subsets of {X}, that are indicated as permitting particular classes of attack (i.e. types of attack or attack scenarios). The mappings between the relevant configuration parameters and attack characteristics can be represented in an association data structure such as the matrix 142 depicted in Figure 6.

Figure 6 illustrates a matrix 142 mapping VM configuration features 152 against attack features 150 in an exemplary embodiment of the present invention. As can be seen from the exemplary data structure of Figure 6, the attack feature “Changes in System files” occurred on VMs that, for example, have “Admin Allowed to read files”, “Registry change allowed” and “SSH Allowed”. Thus the set of reduced features {Y} permits the identification of associations between configuration features 152 and attack features 150. Notably the attack features are not specific attacks but rather classes or types of attack (e.g. an attack that involves executing malware is a class of attack, not a specific malware attack).

Thus from the data structure 142 it is possible to determine a configuration of a VM that may be susceptible to particular classes of attack. Equally, it is possible to determine configurations of VM that are indicated to be less susceptible to particular classes of attack. Accordingly, on the basis of the reduced set of features determined by learning of the latent factor extractor 130 an indication of susceptibility of a VM configuration can be evaluated, and further a configuration or modifications to a configuration of a VM can be determined. Thus in some embodiments a component implemented as hardware, software, firmware or a combination component such as monitoring agents instantiated with, within or in association with one or more VMs and in communication with an attack analysis and assessment component 118 according to Figure 5 and/or a feature classification 142 such as the data structure of Figure 6 is operable to one or more of: determine or have determined whether a VM is susceptible to a class of attack based on its configuration; modify a VM configuration

to mitigate or reduce susceptibility to one or more classes of attack; and/or generate a VM configuration for mitigating or reducing susceptibility to one or more classes of attack.

Figure 7 illustrates a further arrangement of the attack analysis and assessment component 118 of Figure 4 in accordance with embodiments of the present invention. Given a particular uninfected VM with a set of configuration parameters, denoted as features set {X'}, the classification process will make use of the outcome from an earlier training phase (i.e. trained algorithms defining a reduced set of features {Y}) in conjunction with a set of detected attack features {A} in order to assess whether or not there will be an attack at the VM. In the following an exemplary implementation of an attack analysis and assessment component 118 using Restricted Boltzmann Machine as its machine learning algorithm is described.

Figure 8 illustrates a restricted Boltzmann machine for use in exemplary embodiments of the present invention. A restricted Boltzmann Machine (RBM) is a stochastic neural network, i.e. a network of neurons where each neuron has some random behaviour when activated. It consists of one layer of visible units 152, one layer of hidden units 154 and a bias unit 156. Each visible unit is connected to all the hidden units (this connection is undirected, so each hidden unit is also connected to all the visible units), and the bias unit 156 is connected to all the visible units and all the hidden units. The bias unit 156 is used to allow other units to learn an appropriate threshold. No visible unit is connected to any other visible unit and no hidden unit is connected to any other hidden unit. After successful learning, an RBM provides a closed-form representation of the distribution underlying the training data.

In embodiments of the present invention the latent feature extractor 130 includes an RBM as a classifier where the RBM is trained to model a joint probability distribution of inputs (features set {X} of VM configuration features based on VM characteristics 114) and corresponding labels (features set {A} of attack features based on attack characteristics 116), both represented by the visible units of the RBM. The hidden units represent a reduced set of features {Y} that, after training, can constitute a set of latent factors. The RBM works by updating states of some units given the states of others. A unit can have a binary state: state 0 (false – not activated); or state 1 (true – activated). Hence the VM configuration features and attack features are preferably represented as a binary vector.

For example, a set of features {X} for VM configuration features can include binary indications of the following features:

- DNS allowed
- Email allowed

- Admin allowed to read file
- OS is Window 7.0
- HTTP allowed

For example, a set of detected attack features {A} for a VM can include binary indications of the following features:

- Malware running
- Connection to malicious sites detected
- Automatic redirection
- Change in system files

Prior to training the RBM a set of management features {X} and attack feature {A} for an entire training data set need to be determined. It is necessary to determine the aggregate set of VM configuration features and attack features for the plurality of VMs in the training data set in order to determine a size of a required binary vector and, accordingly, a number of visible units for the RBM. For example, training data can consist of configuration features for a plurality of VMs with confirmed attack features. This means that there will be different sets of VM configuration parameters and attack characteristics for different VMs. Some of the configuration parameters are shared among the VMs and some are not. The same also applies to the attack features. Therefore, when a complete set of features {X} is passed to an RBM's visible units for a single VM, some visible units will activate (indicating features that are present in the set {X}, such as by binary '1' indication) and some will not (features that are absent in the set {X}, such as by binary '0' indication).

Figure 9 illustrates the determination of an aggregate set of VM configuration features {X} and an aggregate set of attack features {A} in an exemplary embodiment of the present invention. While only two VMs are indicated in Figure 9 it will be appreciated by those skilled in the art that more training data will lead to an RBM having a better capability to identify classifications for input data. Thus, in Figure 9, a first VM VM1 has a set of configuration features that differs from that of a second VM VM2, and further VM1 exhibits different attack features to VM2. The aggregate set of all possible configuration features is indicated as set {X} and includes seven possible features, so set {X} includes binary vectors having seven elements each thus: [0,0,0,0,0,0,0]. Further, the aggregate set of all possible attack features is indicated as set {A} and includes four possible features, so set {A} includes binary vectors having four elements thus: [0,0,0,0]. The number of visible units in the RBM is the sum of the number of features {X} and the number of features {A} and binary feature vectors for training the RBM will each be constituted as eleven element vectors comprising {{X},{A}} thus: [0,0,0,0,0,0,0,0,0,0,0]. A number of hidden units can be determined during an RBM training

phase to achieve an acceptable level of accuracy – a greater number of hidden units offering a wider diversity of classifications but fewer discrete classes (i.e. a larger set {Y}) while a smaller number of hidden units focuses classification on fewer classes but can lose subtle latent factors (i.e. a smaller set {Y}). The selection of an appropriate number of hidden units is thus a matter of tuning to achieve a desirable classification.

Figure 10 illustrates exemplary input vectors 160a and 160b for an RBM based on the features of Figure 9. Figure 10 shows how the features of VM1 and VM2 can be prepared for input as visible units to train the RBM, each vector 160a and 160b constituting an item of training data and the collective of all vectors constituting the training data set.

Additionally, preferably the configuration features of VMs which are confirmed to not have suffered any attack or infection can optionally be provided as further training data by mapping into an input binary vector for visible units with the corresponding attack feature vector being set to all zeros or false (to indicate no attack). Such an approach provides non-attacked VM configurations to the RBM to support the RBM in learning how to classify potentially safely-configured VMs.

Thus the RBM is trained with example features from infected and non-infected VMs input as inputs to the visible units. The objective of the training process is for the RBM to learn connection weights between the units, i.e. visible, hidden and bias. The training can be performed using an algorithm known as “Contrastive Divergence Learning” such as is described in Geoffrey Hinton’s paper “A Practical Guide to Training Restricted Boltzmann Machines” (August 2, 2010; University of Toronto Department of Computer Science). In summary contrastive divergence involves performing a number of iterations to compute states of hidden units based on states of visible units and vice versa, wherethe states of visible units are reconstructed from the hidden units. A number of iterations increases with learning steps to achieve improved accuracy. A number of hidden units is estimated at the start of learning phase and may be adapted to achieve better accuracy.

The trained RBM constitutes a model for the joint probability distribution of all inputs consisting of features sets {X} and {A}. The model is mainly represented by the computed weights of the connections between visible (v) and hidden (h) units/neurons. The distribution function p(v,h) is determined by the activation energy function E(v,h) defined by the model. p(v,h) is close to 1 for large positive activation energies, and p(v,h) close to 0 for negative activation energies. Units that are positively connected to each other try to get each other to share the same state (i.e., be both on or off), while units that are negatively connected to each other are enemies that prefer to be in different states. This behaviour can also be used to determine a susceptibility to attack in embodiments of the present invention.

Following training of the RBM the data structure manager 140 subsequently generates the feature classification data structure 142 such as a matrix, table or the like such as the matrix illustrated in Figure 6. A classification process is employed using the features sets {X}, {A} and the reduced set {Y} (or hidden units) of the trained RBM. The feature classification data structure 142 can be generated through sampling of visible units in the RBM based on hidden having randomly defined activation states. Thus Figure 11 illustrates states of hidden and visible units of a restricted Boltzmann machine as part of a sampling process in an exemplary embodiment of the present invention. The process can be summarised as:

1. A random sequence 174 for states of the hidden units is generated.
2. The hidden units are input to the trained RBM hidden units.
3. The RBM generates a number of samples of visible units.
4. The sampled visible units are extracted to configuration features set {X''} and attack features set {A'}.
5. The new features sets {X''} and {A'} are then mapped to an *m x n* matrix (*m* and *n* are the lengths of features sets {X'} and {A'}, respectively). Preferably, only sampled visible units with one or more non-zero values of attack features set {A'} are considered for inclusion in the matrix.
6. The whole sampling process is repeated multiple times with new random sequences 174 at step 1 to build a comprehensive hotspot matrix.

The resulting data structure (matrix) can subsequently be employed for: reconstructing possible attack scenarios for compromising a VM; determining a susceptibility of a VM configuration to an attack scenario; and determining a VM configuration for mitigating or reducing a susceptibility to an attack scenario.

Figure 12 is a component diagram illustrating an arrangement including a susceptibility determiner 184 component for determining whether a target VM is susceptible to a security attack based on a pre-existing VM configuration 180 for the target VM in accordance with some embodiments of the present invention. The susceptibility determiner 184 is a hardware, software, firmware or combination component for determining susceptibility of the target VM to attack. The susceptibility determiner accesses a feature classification 142 generated according to the techniques hereinbefore described. For example, the feature classification 142 can comprise a matrix, table or other data structure such as the matrix of Figure 6. The susceptibility determiner 184 further accesses the pre-existing VM configuration 180 for the target VM to determine if the target VM is susceptible to a security attack. The attack can be a particular attack being associated with one or more attack characteristics on which bases the feature classification 142 is defined. Alternatively, the attack can be identified directly in terms of one more attack features in the classification 142. The susceptibility determiner 184

thus uses the VM configuration for the target VM to identify attack characteristics identified in the feature classification 142 to which the target VM is susceptible. In this way attack characteristic susceptibility of the target VM can be determined and remediation or protective measures can be employed.

For example, each attack characteristic can have associated one or more protective measures such, inter alia: a configuration parameter or change to a configuration parameter for a VM to protect against attacks exhibiting a particular characteristic, such as disabling DNS redirection, restricting access to certain resources such as files or directories, closing certain network ports, and the like; and/or an additional function, routine, facility, service or other resource suitable for detecting and/or protecting against attacks exhibiting a particular characteristic, such as antimalware software, intrusion detection facilities, proxies and firewalls and the like.

Thus, in this way embodiments of the present invention provide for the determination of susceptibility of a target VM to security attacks. The susceptibility can be quantified such as a degree of susceptibility and remediation or protective measures or deployment determinations for the target VM can be based on the determined degree of susceptibility.

Figure 13 is a component diagram illustrating an arrangement including a configuration generator 188 for determining a configuration 186 of a target VM to protect against a security attack exhibiting a particular attack characteristic in accordance with some embodiments of the present invention. The configuration generator 188 is a hardware, software, firmware or combination component for generating the VM configuration 186. The configuration generator 188 accesses a feature classification 142 generated according to the techniques hereinbefore described. For example, the feature classification 142 can comprise a matrix, table or other data structure such as the matrix of Figure 6. Furthermore, the configuration generator 188 preferably receives an identification of one or more attack characteristics to from which the target VM is intended to be protected. Alternatively, the configuration generator 188 can be configured to generate a VM configuration 186 that protects against substantially all, or a majority of, or a subset of attack characteristics indicated in the feature classification 132. Where protection is provided against a subset the subset may be determined based on, for example, a prioritisation of attach characteristics or an assessment of attack characteristics relevant to a particular VM based on one or more software components to be executed by the VM or use case definition for the VM. Thus, in use, the configuration generator 188 inspects the feature classification 142 to determine configuration parameters for the target VM that are not associated with attack characteristics that the VM is to be protected from. In this way a VM configuration can be generated that serves to reduce a susceptibility of the target VM to attacks having particular attack characteristics.

It will be appreciated by those skilled in the art that protection against attacks exhibiting a particular attack characteristic need not provide a guarantee of absolute avoidance or removal of attacks with such characteristics, rather protection seeks to reduce susceptibility, mitigate and/or avoid such attacks.

Figure 14 is a component diagram illustrating an arrangement including a configuration updater 189 for determining a configuration of a VM to protect against a security attack exhibiting a particular attack characteristic and updating a pre-existing VM configuration 180 for a target VM to protect against attacks having the attack characteristic based on the determined configuration in accordance with some embodiments of the present invention. The manner of operation of the updater 189 of Figure 14 is similar to that of the configuration generator 188 of Figure 13 except that the updater 189 is further adapted to access the pre-existing VM configuration 180 and update the configuration 180 in view configuration parameters determined to protect against certain attack characteristics based on the feature classification to generate an updated or replacement VM configuration 186 for the target VM.

Figure 15 is a flowchart of a method to generate a classification scheme for configuration parameters of VMs in accordance with some embodiments of the present invention. Initially, at step 190, a machine learning algorithm is trained as a classifier based on a plurality of training data items, each training data item corresponding to a training VM and including a representation of parameters for a configuration of the training VM and a representation of characteristics of security attacks for the training VM. Subsequently, at step 192, a data structure is generated for storing one or more relationships between VM configuration parameters and attack characteristics. The data structure is generated by sampling the trained machine learning algorithm to identify the relationships.

Figure 16 is a flowchart of a method to determine whether a target VM is susceptible to a security attack in accordance with some embodiments of the present invention. Steps 190 and 192 are substantially as described above with respect to Figure 15. Subsequently, at step 194, a set of configuration parameters for the target VM are determined. At step 195 attack characteristics in the data structure associated with configuration parameters of the target VM are identified as characteristics of attacks to which the target VM is susceptible.

Figure 17 is a flowchart of a method to determine a configuration of a target VM to protect against a security attack exhibiting a particular attack characteristics in accordance with some embodiments of the present invention. Steps 190 and 192 are substantially as described above with respect to Figure 15. Subsequently, at step 196, the particular attack characteristic in the data structure are identified to determine a set of VM configuration parameters indicated as associated with the particular attack characteristic. At step 198 a VM configuration is generated for the target VM wherein the configuration parameters in the

determined set of VM configuration parameters are absent in the generated VM configuration.

Figure 18 is a component diagram of an arrangement for attack mitigation in accordance with embodiments of the present invention. An attack mitigation component 204 is provided as a hardware, software, firmware or combination component for mitigating an attack against a target VM where the attack exhibits one or more particular attack characteristics. The attack mitigation component 204 thus accesses a VM configuration 200 for the target VM and a directed graph data structure 202. The directed graph data structure 202 is predefined based on the feature classification 142 generated by the attack analysis and assessment component 118. The directed graph includes vertices representing VM configuration parameters connected by directed edges to form sequences of VM configuration parameters involved in achieving a particular attack characteristic for an attack. In some embodiments the attack mitigation component 204 generates new or modified VM parameters 206 as described below. An exemplary arrangement in respect of an exemplary malware attack characteristic will now be described.

Figure 19 illustrates an exemplary entry in a feature classification data structure 142 for a malware attack characteristic in accordance with an exemplary embodiment of the present invention. The feature classification entry of Figure 19 is generated by the attack analysis and assessment component 118 following training of a latent feature extractor 130 based on a plurality of training data items as training examples. As can be seen in Figure 19 an attack characteristic corresponding to the execution of malware in a VM is characterised by a number of VM configuration parameters including: email being permitted; Windows 10 operating system being used; file transfer protocol (FTP) being permitted; hypertext transport protocol (HTTP) being permitted; write access to a file system directory being permitted; administrator-level login being permitted; and superuser privilege being permitted.

Figure 20 illustrates a data structure storing a directed graph representation of sequences of VM configuration parameters for the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention. The graph of Figure 19 can be generated by a systems analyst, user or VM administrator and reflects latent knowledge of how the VM configuration parameters identified for the malware attack characteristic in Figure 19 can be arranged in ordered sequence(s) in order for an attack having such a characteristic to take place. Thus it can be seen in Figure 20 that sequences start at the “start” vertex and follow sequences through the graph to a final vertex in which “malware executes” is indicated. All sequences start at vertex 1 based on the “email allowed” VM configuration parameter. One sequence proceeds through vertices 2, 4, 5 and 6 representing VM configuration parameters

"DNS redirection permitted", "FTP allowed", "directory write access permitted" and "admin login permitted". Alternative sequences through the graph also exist, such as the sequence through vertices 1, 3, 5, 7 corresponding to: "Email allowed", "directory write access permitted", and "super user privileges permitted". Other sequences also exist such as, inter alia: 1, 3, 4, 5, 6; 1, 3, 5, 6; and 1, 2, 3, 5, 6. Thus the directed graph of Figure 20 represents multiple sequences from the "start" vertex to the "malware executes" vertex with each sequence comprised of a list of VM configuration parameters for achieving the particular attack characteristic. Preferably the directed graph is stored as a data structure for access by an attack mitigation component 204, such as data structures well known to those skilled in the art.

Figure 21 illustrates states of an exemplary configuration of a VM in accordance with the VM configuration parameters of Figure 19 and in accordance with an exemplary embodiment of the present invention. Notably the configuration parameters indicated in Figure 21 are for one specific VM implementation (as opposed to an entire feature classification 142) though, in the exemplary embodiment, the parameters are defined by a vector of binaries in terms of all possible VM parameters of the feature classification 142 of Figure 19.

Thus the VM associated with the VM configuration of Figure 21 exhibits only a subset of the VM configuration parameters of Figure 19 (for example, not exhibiting "FTP allowed"). The directed graph of Figure 20 can be used to determine any subset of sequences corresponding to the VM configuration parameters of the VM of Figure 21. Thus Figure 22 illustrates a subset of sequences in the directed graph of Figure 20 corresponding to VM parameters of the VM of Figure 21 in accordance with an exemplary embodiment of the present invention. The subset of sequences is shown by the emphasised continuous arrows in Figure 22. It can be seen, therefore, that the VM configuration parameters associated with the VM of Figure 21 do indeed constitute a subset of the sequences indicated by the directed graph and accordingly it can be concluded that the VM is susceptible to an attack exhibiting a malware attack characteristic.

Figure 23 is a flowchart of a method to identify configuration parameters of a target VM used in a security attack against the target VM in accordance with embodiments of the present invention. Initially the method performs the steps 190 and 192 as previously described to generate the feature classification data structure 142. Subsequently, at step 210, the method receives a data structure storing a directed graph representation of sequences of VM configuration parameters for achieving an attack characteristic of the security attack. The directed graph is determined based on the feature classification data structure. At step 212 the method determines a subset of sequences in the directed graph

corresponding to VM parameters of the target VM to identify VM parameters of the target VM used in the security attack. Thus, in this way the method identifies parameters of a configuration of the target VM used in a security attack against the target VM.

Once such VM configuration parameters have been identified then mitigation measures against the security attack can be employed. Figure 24 illustrates exemplary security facilities that can be employed to mitigate the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention. Each VM configuration parameter in the directed graph of Figure 24 has associated one or more security facilities that may be employed to mitigate or protect the VM or to reduce the risk of attack or success of an attack. For example, the "email allowed" parameter can be supplemented by security facilities for: scanning email; scanning for malware in email; removing attachments to emails; and/or removing or replacing links in emails. The "DNS redirection permitted" parameter can be supplemented by security facilities for detecting DNS redirection. The "HTTP allowed" parameter can be supplemented by security facilities such as: a firewall; a proxy; an HTTP filter; a download detector; and a malware scanner. The "FTP allowed" parameter can be supplemented by security facilities for: detecting downloads; and malware scanning. The "directory write access permitted" parameter can be supplemented by security facilities for malware scanning. The "admin login permitted" and "super user privileges permitted" parameters can be supplemented by security facilities for: enhanced authentication; multi-factor such as 2-factor authentication; logging of authentication attempts; and monitoring of the behaviour of administrators logged-in.

Figure 25 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention. Initially the method performs the steps 190, 192, 210 and 212 as previously described. Subsequently, at step 214, the target VM configuration is supplemented by one or more security facilities associated with one or more of the VM parameters identified for the target VM. Thus, considering the VM parameters for the VM of Figure 21 any or all of the security facilities associated with the "email allowed", "DNS redirection permitted", "HTTP allowed", "directory write access permitted", and "super user privileges permitted" may be configured to be applied to the VM to mitigate the malware attack.

As an alternative to mitigating an attack by the inclusion of security features, modifications to VM configuration parameters themselves may be adopted. Figure 26 illustrates exemplary VM configuration parameter changes that can be employed to mitigate the malware attack of Figure 19 in accordance with an exemplary embodiment of the present invention. Thus Figure 26 illustrates how any of the VM configuration parameters of the VM of Figure 21 may

be changed to break the sequence through the directed graph and so mitigate the malware attack. Accordingly, Figure 27 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention. Initially the method performs the steps 190, 192, 210 and 212 as previously described. Subsequently, at step 216, the method reconfigures the target VM by changing one or more VM parameters identified by directed graph as being included in the sequence of parameters for the attack characteristic.

One challenge remaining with the approach of Figure 27 is the possibility that an attack with the malware attack characteristic can nonetheless be brought against a VM even when the sequence of parameters for the VM in the directed graph is broken. For example, mitigation of the attack characteristic of Figure 26 by setting “HTTP allowed = false” could lead to circumvention of the mitigation measure, such as to employ FTP or an alternative communication mechanism.

To illustrate this challenge clearly reference is made to Figure 28. Figure 28 illustrates a data structure storing a directed graph representation of sequences of VM configuration parameters for an attack characteristic in accordance with an exemplary embodiment of the present invention. The directed graph of Figure 28 is considerably larger and more complex than that previously considered and it is to be recognised that directed graphs modelling sequences of VM parameters for real deployed VMs can be large and complex with many sequences leading from a “start” vertex to an “attack” vertex corresponding to an attack characteristic. Notably the graph of Figure 28 shows many alternative sequences to achieve the attack characteristic, such as the initial selection between vertices 12, 3 and 6, and even then further selections such as from vertex 12 to any of vertices 22, 21 and 15. Thus it can be seen that there are many routes through the graph of Figure 28. However, there are notably commonalities in the graph of Figure 28 also. In particular, all sequences ultimately pass through one of vertices 11 or 1 and all sequences ultimately pass through vertex 4. Other commonalities can be found also, such as all sequences pass through one of vertex 22, 7 or 1, and others that can be identified. Thus it is possible to rationalise a particular sequence or sequences through the directed graph to common vertices and address mitigation measures to the VM parameters associated with those vertices. Such rationalisation will involve the selection of a subset of vertices through which all sequences pass. This selection can be driven by an objective, such as a predetermined criteria. For example, the predetermined criteria can require that the selection of vertices for mitigation is based on a minimum number of vertices to cover all sequences through the graph. Alternatively other criteria may be used, such as a proportion coverage of sequences or a guaranteed coverage of specific sequences.

In some cases mitigation of a particular VM parameter may not be possible or may be undesirable. For example, a security facility may not be available for a particular VM parameter and/or it may not be possible to reconfigure a VM parameter due to constraints on the VM. For example, a VM operating as a web server must communicate via HTTP networking ports and it may therefore not be possible to close those ports on such a server. Accordingly, it can be desirable to select mitigation measures and vertices in the graph as a basis for mitigation based on some ranking, prioritisation or preference mechanism such that more appropriate/preferred VM parameters are modified in favour of less appropriate/preferred parameters.

In one embodiment some or all vertices (and the VM parameters they represent) in the directed graph are each associated with a predetermined weight or score. In such an embodiment the predetermined criteria for selecting vertices for mitigation are defined based on such weights or scores. For example, individual vertices can be selected that meet a predetermined threshold weight or score. Alternatively, a collection of vertices can be selected that collectively meet a predetermined weight or score (i.e. a total of all weights or scores meets a predetermined condition). Such a condition can be, for example, a maximum or minimum weight or score. Such an approach is helpful where it is desirable to indicate an importance, relevance, appropriateness or preference of VM parameters such that, for example, a weight or score can indicate an importance of a VM parameter where parameters that are more important have more impact on an overall weight.

Thus Figure 29 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention. Initially the method performs the steps 190, 192, 210 and 212 as previously described. Subsequently, at step 220 the directed graph is analysed to select at least one vertex through which all sequences for the attack characteristic pass. This analysis can be achieved by various algorithms as will be apparent to those skilled in the art for directed graph analysis such as a method in which all possible sequences through the graph are identified to determine individual vertices common to all sequences or a set of vertices whereby each sequence through the graph includes at least one element from the set. Subsequently, at step 222, the method reconfigures the target VM based on the selected vertices to mitigate attacks exhibiting the attack characteristic.

All the above methods are effective for identifying and/or mitigating attacks exhibiting an attack characteristic. However, a challenge remains where an attack characteristic continues to be observed in a VM despite mitigation. For example, where all sequences through the directed graph are blocked and yet an attack persists. Such attack characteristics can arise

as a result of the attack adapting to employ other services and/or facilities of a VM not currently modelled in the directed graph. Such a situation can be addressed by causing the retraining of the RBM to provide for the regeneration of the feature classification data structure. In particular, the retraining of the RBM must be undertaken with at least some training examples (data items) corresponding to the attack having the attack characteristic that exists despite the mitigation measures. Accordingly, the retraining will generate a new feature classification data structure 142 on which bases a new directed graph can be generated. Such new directed graph can then be employed to model the VM parameters employed by the attack characteristic to implement mitigation measures as hereinbefore described.

Thus Figure 30 is a flowchart of a method to mitigate a security attack against a target virtual machine in accordance with embodiments of the present invention. Initially the method performs the steps 190, 192 and 210 as previously described. Subsequently, at step 230, the method identifies VM parameters of a target VM used in the security attack, such as by way of the techniques described above. At step 232 the method determines if the security parameters form a continuous sequence in the directed graph from a start vertex to an attack vertex. Where there is such a continuous sequence then a mitigation can be implemented at step 236 in accordance with the techniques described hereinbefore. However, where there is no such sequence then the method proceeds to step 234 in which new training data items are generated for one or more training VMs including VMs subject to the attack for which a sequence was not identified. Subsequently the method causes retraining of the RBM by returning to step 190 and the method repeats until a sequence through a regenerated directed graph is identified on which basis mitigation can be applied.

Insofar as embodiments of the invention described are implementable, at least in part, using a software-controlled programmable processing device, such as a microprocessor, digital signal processor or other processing device, data processing apparatus or system, it will be appreciated that a computer program for configuring a programmable device, apparatus or system to implement the foregoing described methods is envisaged as an aspect of the present invention. The computer program may be embodied as source code or undergo compilation for implementation on a processing device, apparatus or system or may be embodied as object code, for example.

Suitably, the computer program is stored on a carrier medium in machine or device readable form, for example in solid-state memory, magnetic memory such as disk or tape, optically or magneto-optically readable memory such as compact disk or digital versatile disk etc., and the processing device utilises the program or a part thereof to configure it for

operation. The computer program may be supplied from a remote source embodied in a communications medium such as an electronic signal, radio frequency carrier wave or optical carrier wave. Such carrier media are also envisaged as aspects of the present invention.

It will be understood by those skilled in the art that, although the present invention has been described in relation to the above described example embodiments, the invention is not limited thereto and that there are many possible variations and modifications which fall within the scope of the invention.

The scope of the present invention includes any novel features or combination of features disclosed herein. The applicant hereby gives notice that new claims may be formulated to such features or combination of features during prosecution of this application or of any such further applications derived therefrom. In particular, with reference to the appended claims, features from dependent claims may be combined with those of the independent claims and features from respective independent claims may be combined in any appropriate manner and not merely in the specific combinations enumerated in the claims.

CLAIMS

1. A computer implemented method to mitigate a security attack against a target virtual machine (VM) in a virtualised computing environment, the target VM having a target VM configuration including configuration parameters, and the security attack exhibiting a particular attack characteristic, the method comprising:

training a machine learning algorithm as a classifier based on a plurality of training data items, each training data item corresponding to a training VM and including a representation of parameters for a configuration of the training VM and a representation of characteristics of security attacks for the training VM;

generating a first data structure for storing one or more relationships between VM configuration parameters and attack characteristics, wherein the first data structure is generated by sampling the trained machine learning algorithm to identify the relationships;

receiving a second data structure storing a directed graph representation of one or more sequences of VM configuration parameters for achieving the particular attack characteristic of the security attack, the VM parameters in the directed graph being determined based on the first data structure;

identifying VM parameters of the target VM used in the security attack;

in response to a determination that the VM parameters of the target VM do not form a continuous sequence in the directed graph, triggering the steps of:

a) generating new training data items for one or more training VMs including at least one VM being subject to the attack;

b) repeating the training and generating steps so as to generate a new first data structure of relationships; and

c) receiving a new second data structure based on the new first data structure.

2. The method of claim 1 further comprising:

identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph of the new second data structure corresponding to VM parameters of the target VM; and

supplementing the target VM configuration with a security facility associated with at least one of the identified VM parameters so as to protect the target VM from the attack.

3. The method of claim 1 further comprising:

identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph of the new second data structure corresponding to VM parameters of the target VM; and

reconfiguring the target VM by changing at least one of the identified VM parameters so as to stop the attack.

4. The method of claim 1 further comprising:

identifying VM parameters of the target VM used in the security attack as a subset of sequences in the directed graph corresponding to VM parameters of the target VM;

analysing the second data structure to select one or more vertices of the directed graph each indicating a VM parameter, wherein all sequences of VM configuration parameters for achieving the attack pass through at least one of the vertices;

reconfiguring the target VM by changing VM parameters indicated in each of the identified vertices, wherein the vertices are selected to include VM parameters according to predetermined criteria.

5. The method of claim 4 wherein the predetermined criteria are defined to require a minimum number of VM parameters.

6. The method of any of claims 4 or 5 wherein each vertex in the directed graph has associated a predetermined weighting based on a VM parameter indicated by the vertex and wherein the predetermined criteria are defined to require that each selected vertex meets a predetermined condition in relation to their associated weighting.

7. The method of any of claims 4 to 6 wherein each vertex in the directed graph has associated a predetermined weighting based on a VM parameter indicated by the vertex and wherein the predetermined criteria are defined to require that a total of all weightings of all selected vertices meets a predetermined condition.

8. The method of any of claims 4 or 5 wherein the predetermined condition is a maximum weight.

9. The method of any of claims 4 to 8 wherein the weighting is an indication of importance of a VM parameter such that parameters that are more important have more impact on the overall weight.

10. A computer system including a processor and memory storing computer program code for performing the steps of any preceding claim.

11. A computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of a method as claimed in any of claims 1 to 9.

Intellectual Property Office

Application No:	GB1711880.3	**Examiner:**	Mr Tristan Ballard
Claims searched:	1-11	**Date of search:**	15 January 2018

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance
A	-	US 2016/0164894 A1 (GUARDICORE)
A	-	US 8479276 B1 (EMC)
A	-	US 2010/0199351 A1 (PROTAS)
A	-	US 2013/0055398 A1 (RAPID7)

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC^{X} :

Worldwide search of patent documents classified in the following areas of the IPC

G06F

The following online and other databases have been used in the preparation of this search report

EPODOC, WPI, Patent Fulltext

International Classification:

Subclass	Subgroup	Valid From
G06F	0021/56	01/01/2013
G06F	0021/50	01/01/2013