

SYSTEM AND METHOD OF RENDERING DATA-SETS FOR DIFFERENTIATING A PLURALITY OF POTENTIAL THREAT ACTORS

FIELD OF INVENTION

5

[0001] The present invention relates to system and method for differentiating insider threat actors from external threat actors, in particular to system and method of rendering data-sets for differentiation of insider threat actors and external threat actors.

10

BACKGROUND

[0002] Typically, the computer security technology has developed significant capabilities for protecting against external threat actors (e.g., firewalls, intrusion detection systems, etc.); however, technology for protecting the computer infrastructure from insider threat actors is significantly weaker. The organizations utilize various security systems and mechanisms to identify the type of threat actors. The existing security system captures the processes activities of the computer system which lead to yield noisy data. The noisy retrieved data is difficult and expensive to process and store for further analysis. Additionally, the conventional raw log data requires deep knowledge and expertise to interpret and sometimes yields little or inconclusive information as to answer if malicious acts performed on the computer system are performed by the insider user or by abusing a compromised account by external parties. If it were an externally delivered attack to end-users, it is difficult and time-consuming to forensically trace the first victim, sometimes termed as "Patient-Zero" in a large network of users.

15

20

25

30

[0003] Further, existing sophisticated methods like User-Entity-Behavioural Analytics, using the existing raw log data is inconclusive or lead to false-positives as the detection algorithms are only as good as the inputs provided. Many of the sophisticated detection-sensors are still dependent on the underlying logging mechanisms which can be disabled upon escalation of privilege by threat actors, and most platforms are unable to detect such tampering. By the time the incidents are reported, the relevant logs and meta-data within the computer system may have already been wiped or tampered to obstruct any further investigations.

[0004] US patent number 8,769,684 B2 filed by Stolfo et al. discloses a system and method for detecting masquerade attacks. The method disclosed is Stolfo et al. monitors user actions and access to the decoy information in a computing environment. Then the method generates the user intent model for a category that includes first user actions and monitors the second user actions. Further, the method compares the second user actions with the user intent model by determining deviation from the generated user intent model. The method then identifies whether the second user actions is a masquerade attack based on the comparison. However, the method for detecting masquerade attacks disclosed in the Stolfo et al. reference does not differentiate the insider threat actors and external threat actors by monitoring the user actions and generating the user intent model.

[0005] Another US patent number 8,793,790 B2 filed by Khurana et al. discloses a system and method to obtain data related to access the cyber and physical assets from a cyber and physical access control system. The method compares the obtained data to predefined patterns of expected behaviour and identifies the patterns of the suspicious behaviour as a function of comparing the data to the predefined patterns of the expected behaviour. Then the method utilizes the comparison to identify the potentially malicious insider behaviour toward the cyber and physical assets. However, the method for detecting inside threats disclosed in the Khurana et al. reference is not able to accurately differentiate the insider threat actors and external threat actors by comparing the obtained data with the predefined patterns of the expected behaviour.

SUMMARY

[0006] There is a need for a reliable, efficient and cost-effective system and method of rendering data-sets for differentiation of insider threat actors and external threat actors. Further, there is a need for a system and method that provides data-sets of the applications were in use during a "Hands-on" session. Furthermore, there is a need for a system and method to provide a reliable differentiation of foreground processes and background processes for system base-lining and user-profiling.

[0007] The present invention mainly solves the technical problems existing in the prior art. in response to these problems, the present invention provides a system and method of rendering data-sets for differentiating a plurality of potential threat actors.

[0008] An aspect of the present disclosure relates to a method of rendering data-sets for differentiating a plurality of potential threat actors. The method includes a step of integrating a user action tracking (UAT) module into a plurality of global keystroke and mouse hooks of an operating system (OS) through an integration module.

5 **[0009]** The method then includes the step of detecting a plurality of user action events initiated by a user through a first detection module to generate a first detection data. The plurality of user action events, in one embodiment, comprise a mouse clicking event, a mouse movement event, and a key pressing event. Then the method includes the step of capturing a foreground application process identifier (ID) on detection of the user action
10 events through the user action tracking (UAT) module to generate a software signal based on the first detection data. The method includes the step of recording the plurality of user action events and the application process identifier (ID) into an operating system (OS) logging facility through the user action tracking (UAT) module. Further, the method includes the step of detecting a plurality of hardware inputs pertaining to a keyboard and a
15 mouse through a second detection module to generate a second detection data. The second detection module is configured within a computing device.

[0010] The method then includes the step of recording the second detected data and an operating system (OS) instrumentation service through the computing device to generate a hardware signal. The method then comprises a step of determining a physical interaction
20 of the user based on the software signal received from the user action tracking (UAT) module and the hardware signal received from the computing device through a central module. The central module detects a plurality of potential threat actors on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof. The potential threat actors comprise a network security threat actor, a
25 communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.

[0011] In an aspect, the plurality of hardware inputs comprises a mouse input, a keyboard input, a trackpad input, a touch input, a video input, audio input and/or a combination thereof. The plurality of hardware inputs lead to create the process or to terminate the
30 process to initiate a plurality of active foreground transitions.

[0012] In an aspect, the computing device records the plurality of active foreground transitions.

[0013] In an aspect, the foreground application process identifier (ID) is a numeric or a test-string value to identify an active foreground process (AFP).

5 [0014] In an aspect, the recorded plurality of user action events and the application process identifier (ID) are transferable to a plurality of centralized log collectors and a plurality of detection units.

[0015] In an aspect, the hardware signals generated by the computing device utilizes an out-of-band channel to transmit the second detected data to a centralized management
10 service unit for recording. Further, the centralized management service unit transmits the recorded second detected data to the detection units.

[0016] An aspect of the present disclosure relates to a device in a network. The device includes a non-transitory storage device having embodied therein one or more routines operable to render data-sets for differentiating a plurality of potential threat actors. The one
15 or more routines include an integration module, a user action tracking (UAT) module, a first detection module, a second detection module, and a central module.

[0017] The integration module is configured to integrate a user action tracking (UAT) module into a plurality of global keystroke and mouse hooks of an operating system (OS). The first detection module is configured to detect a plurality of user action events initiated
20 by a user to generate a first detection data. The plurality of user action events comprises a mouse clicking event, a mouse movement event, and a key pressing event. The user action tracking (UAT) module is configured to capture a foreground application process identifier (ID) on detection of the user action events to generate a software signal based on the first detection data. The user action tracking (UAT) module records the plurality of user action
25 events and the application process identifier (ID) into an operating system (OS) logging facility. The second detection module to detect a plurality of hardware inputs pertaining to a keyboard and a mouse to generate a second detection data. The second detection module is configured within a computing device.

[0018] The computing device records the second detected data and an operating system
30 (OS) instrumentation service to generate a hardware signal. The central module is

configured to determine a physical interaction of the user based on the software signal received from the user action tracking (UAT) module and the hardware signal received from the computing device. The central module detects a plurality of potential threat actors on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof. The potential threat actors comprise a network security threat actor, a communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.

[0019] An aspect of the present disclosure relates to a system to render data-sets for differentiating a plurality of potential threat actors. The system includes a processor and a memory. The memory stores machine-readable instructions that when executed by the processor cause the processor to integrate a user action tracking (UAT) module into a plurality of global keystroke and mouse hooks of an operating system (OS) through an integration module.

[0020] The processor is further configured to detect a plurality of user action events initiated by a user through a first detection module to generate a first detection data. The plurality of user action events comprises a mouse clicking event, a mouse movement event, and a key pressing event. Then the processor is configured to capture a foreground application process identifier (ID) on detection of the user action events through the user action tracking (UAT) module to generate a software signal based on the first detection data. The processor is configured to record the plurality of user action events and the application process identifier (ID) into an operating system (OS) logging facility through the user action tracking (UAT) module.

[0021] Further, the processor is configured to detect a plurality of hardware inputs pertaining to a keyboard and a mouse through a second detection module to generate a second detection data. The second detection module is configured within a computing device. The processor is then configured to record the second detected data and an operating system (OS) instrumentation service through the computing device to generate a hardware signal. Then the processor is configured to determine a physical interaction of the user based on the software signal received from the user action tracking (UAT) module and the hardware signal received from the computing device through a central module. The central module detects a plurality of potential threat actors on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof. The potential

threat actors comprise a network security threat actor, a communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.

5 [0022] Accordingly, one advantage of the present invention is that it associates the foreground application process with any user-actions and compares with the hardware signals. When both the software signals and the hardware signals are captured together, the system determines that a user was physically present in the computer system. With such data-sets, the present invention provides a reliable proof that it was a "Hands-on" session as opposed to a remotely controlled session as with using remote desktop protocol or equivalent such as Teamviewer, remote admin tools etc.

10 [0023] Generally, external threat actors operate in the background without the user's knowledge. With the ability to quickly know that it is a foreground process in which the user was using, the present invention can reliably differentiate the background processes that are running under the user's context/rights. The present invention uses the refined data-sets generated by the present system to accurately profile users into technical versus non-
15 technical groups for outlier detection methods.

[0024] Accordingly, one advantage of the present invention is that it quickly attributes any alerts of unusual code-execution to specific user actions and circumstances. The present invention generates events into graphs in real-time as the events are collected at the backend. Any alerts from security products or sensors can be used to retrieve the built
20 graphs for quick analysis. With such data-sets, the present invention traces the "Patient-Zero" and attack-vector such as browser drive-by, email with URL and/or attachments and removable storage in parallel.

[0025] Other features of embodiments of the present disclosure will be apparent from accompanying drawings and from the detailed description that follows.

BRIEF DESCRIPTION OF THE DRAWINGS

[0026] In the figures, similar components and/or features may have the same reference label. Further, various components of the same type may be distinguished by following the reference label with a second label that distinguishes among the similar components. If
5 only the first reference label is used in the specification, the description applies to any one of the similar components having the same first reference label irrespective of the second reference label.

[0027] FIG. 1 illustrates a network implementation of the present system and method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance
10 with an embodiment of the present invention.

[0028] FIG. 2 illustrates the present system to render data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention.

[0029] FIG. 3 illustrates a flowchart of the method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present
15 invention.

[0030] FIG. 4 illustrates an operational flowchart of the present method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention.

[0031] FIG. 5 illustrates an exemplary flowchart of the detection of the software signals and the hardware signals, in accordance with an embodiment of the present invention.
20

DETAILED DESCRIPTION

[0032] Systems and methods are disclosed for differentiating insider threat actors from
25 external threat actors. Embodiments of the present disclosure include various steps, which will be described below. The steps may be performed by hardware components or may be embodied in machine-executable instructions, which may be used to cause a general-purpose or special-purpose processor programmed with the instructions to perform the

steps. Alternatively, steps may be performed by a combination of hardware, software, firmware, and/or by human operators.

[0033] Embodiments of the present disclosure may be provided as a computer program product, which may include a machine-readable storage medium tangibly embodying thereon instructions, which may be used to program a computer (or other electronic devices) to perform a process. The machine-readable medium may include, but is not limited to, fixed (hard) drives, magnetic tape, floppy diskettes, optical disks, compact disc read-only memories (CD-ROMs), and magneto-optical disks, semiconductor memories, such as ROMs, PROMs, random access memories (RAMs), programmable read-only memories (PROMs), erasable PROMs (EPROMs), electrically erasable PROMs (EEPROMs), flash memory, magnetic or optical cards, or other type of media/machine-readable medium suitable for storing electronic instructions (e.g., computer programming code, such as software or firmware).

[0034] Various methods described herein may be practised by combining one or more machine-readable storage media containing the code according to the present disclosure with appropriate standard computer hardware to execute the code contained therein. An apparatus for practicing various embodiments of the present disclosure may involve one or more computers (or one or more processors within a single computer) and storage systems containing or having network access to computer program(s) coded in accordance with various methods described herein, and the method steps of the disclosure could be accomplished by modules, routines, subroutines, or subparts of a computer program product.

[0035] The present invention discloses a system and method whereby the OS log tampering are quickly detected. The system and method have two channels of event-collection to detect both software and hardware tampering. For example, if the machine is not known to be a "jump-point", then the user activities without hardware signals are considered anomalies. Conversely, in an event of missing hardware signals but OS events collected could imply either a hardware fault if it affects the entire fleet or it could likely be some hardware tampering if only one or few sets of computing devices (Black PCs) are affected.

[0036] Although the present disclosure has been described with the purpose of differentiating insider threat actors from external threat actors, it should be appreciated that

the same has been done merely to illustrate the invention in an exemplary manner and any other purpose or function for which explained structures or configurations could be used, is covered within the scope of the present disclosure.

[0037] Exemplary embodiments will now be described more fully hereinafter with reference to the accompanying drawings, in which exemplary embodiments are shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein. These embodiments are provided so that this disclosure will be thorough and complete and will fully convey the scope of the invention to those of ordinary skill in the art. Moreover, all statements herein reciting embodiments of the invention, as well as specific examples thereof, are intended to encompass both structural and functional equivalents thereof. Additionally, it is intended that such equivalents include both currently known equivalents as well as equivalents developed in the future (i.e., any elements developed that perform the same function, regardless of structure).

[0038] Thus, for example, it will be appreciated by those of ordinary skill in the art that the diagrams, schematics, illustrations, and the like represent conceptual views or processes illustrating systems and methods embodying this invention. The functions of the various elements shown in the figures may be provided through the use of dedicated hardware as well as hardware capable of executing associated software. Similarly, any switches shown in the figures are conceptual only. Their function may be carried out through the operation of program logic, through dedicated logic, through the interaction of program control and dedicated logic, or even manually, the particular technique being selectable by the entity implementing this invention. Those of ordinary skill in the art further understand that the exemplary hardware, software, processes, methods, and/or operating systems described herein are for illustrative purposes and, thus, are not intended to be limited to any particular name.

[0039] Specific details are given in the following description to provide a thorough understanding of the embodiments. However, it will be understood by one of ordinary skill in the art that the embodiments may be practised without these specific details. For example, circuits, systems, networks, processes, and other components may be shown as components in block diagram form in order not to obscure the embodiments in unnecessary

detail. In other instances, well-known circuits, processes, algorithms, structures, and techniques may be shown without unnecessary detail to avoid obscuring the embodiments.

[0040] The term “machine-readable storage medium” or “computer-readable storage medium” includes, but is not limited to, portable or non-portable storage devices, optical storage devices, and various other mediums capable of storing, containing, or carrying instruction(s) and/or data. A machine-readable medium may include a non-transitory medium in which data can be stored, and that does not include carrier waves and/or transitory electronic signals propagating wirelessly or over wired connections. Examples of a non-transitory medium may include but are not limited to, a magnetic disk or tape, optical storage media such as compact disk (CD) or versatile digital disk (DVD), flash memory, memory or memory devices.

[0041] FIG. 1 illustrates a network implementation **100** of the present system **102** and method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention. Although the present invention is explained considering that the present system **102** is implemented on a server, it may be understood that the present system **102** may also be implemented in a variety of computing systems, such as a laptop computer, a desktop computer, a notebook, a workstation, a mainframe computer, a server, a network server, and the like. It will be understood that the present system **102** may be accessed by multiple users through one or more computing devices **104-1, 104-2...104-N**, collectively referred to as computing unit **104** hereinafter, or applications residing on the computing unit **104**. Examples of the computing unit **104** may include but are not limited to, a portable computer, a personal digital assistant, a handheld or mobile device, smart devices, and a workstation. The computing units **104** are communicatively accessible to the present system **102** through a network **106**.

[0042] In one implementation, the network **106** may be a wireless network, a wired network or a combination thereof. The network **106** can be implemented as one of the different types of networks, such as an intranet, local area network (LAN), wide area network (WAN), the internet, and the like. The network **106** may either be a dedicated network or a shared network. The shared network represents an association of the different types of networks that use a variety of protocols, for example, Hypertext Transfer Protocol (HTTP), Transmission Control Protocol/Internet Protocol (TCP/IP), Wireless Application Protocol (WAP), and the like, to communicate with one another. Further, the network **106**

may include a variety of network devices, including routers, bridges, servers, computing devices, storage devices, and the like.

[0043] FIG. 2 illustrates the present system **102** to render data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention. The system **102** may include at least one processor **202**, an input/output (I/O) interface **204**, and a memory **206**.

[0044] The processor **202** may be implemented as one or more microprocessors, microcomputers, microcontrollers, digital signal processors, central processing units, state machines, logic circuitries, and/or any devices that manipulate signals based on operational instructions. Among other capabilities, the at least one processor **202** is configured to fetch and execute computer-readable instructions stored in the memory **206**.

[0045] The I/O interface **204** may include a variety of software and hardware interfaces allow the user to interact with the system **102** locally through hardware.

[0046] The memory **206** may include any computer-readable medium known in the art including, for example, volatile memory, such as static random access memory (SRAM) and dynamic random access memory (DRAM), and/or non-volatile memory, such as read-only memory (ROM), erasable programmable ROM, flash memories, hard disks, optical disks, and magnetic tapes. The memory **206** may include modules **208** and data **210**.

[0047] The modules **208** include routines, programs, objects, components, data structures, etc., which perform particular tasks or implement particular abstract data types. In one implementation, the modules **208** may include an integration module **212**, a user action tracking (UAT) module **216**, a central module **220**, and other module **222**. The other modules **222** may include programs or coded instructions that supplement applications and functions of the system **102**.

[0048] The data **210**, amongst other things, serves as a repository for storing data processed, received, and generated by one or more of the modules **208**. The data **210** may also include an integration data **224**, a user action tracking (UAT) data **228**, a local input detection data **230**, a central data **232** and other data **234**. The other data **234** may include data generated as a result of the execution of one or more modules in the other module **222**.

[0049] The integration module **212** is configured to integrate a user action tracking (UAT) module **216** into a plurality of global input hooks of an operating system (OS). The first detection module **214** is configured to detect a plurality of user action events initiated by a user to generate a first detection data. The plurality of user action events may include a mouse clicking event, a mouse movement event, and a key pressing event. It is well understood to a skilled person that the illustrated means for triggering user action event are exemplified for illustrations, not limitations. In other embodiments, the user action events may be any inputs to control the system **102**. Such inputs can be triggered through any input interfaces, such as keyboard, mouse, touch screen, audio or video control, etc. It is to be noted that the user action events may include input through a remote access, such as remote desktop connection, whereby a remote user gain access to control the system remotely. It may also include any commands or coding injected into the system to operationally mimic user input controls.

[0050] The user action tracking (UAT) module **216** is configured to capture a foreground application process identifier (ID) on detection of the user action events to generate the first detection data **226**. In an embodiment, the foreground application process identifier (ID) is a numeric or a test-string value to identify an active foreground process (AFP). The recorded plurality of user action events and the application process identifier (ID) are transferable to a plurality of centralized log collectors and a plurality of detection units.

[0051] The user action tracking (UAT) module **216** further records the plurality of user action events and the application process identifier (ID) into an operating system (OS) logging facility.

[0052] A second detection module **218** deployed on computing device external to the system **102** is adapted to detect a plurality of local hardware inputs pertaining to a keyboard and a mouse to generate a second detection data to store on the system **102**. In an embodiment, the plurality of hardware inputs comprises a mouse input, a keyboard input, a trackpad input, a touch input, a video input, and/or a combination thereof. The plurality of hardware inputs lead to create the process or to terminate the process to initiate a plurality of active foreground transitions.

[0053] The computing device **108** is a hardware implementation to record the second detected data and an operating system (OS) instrumentation service to generate a hardware

signal. In an embodiment, all the external input devices may be connected to the system **102** via the computing device **108**, such that all input through the input devices can be logged by the second detection module **218**. In an additional embodiment, the computing device **108** and/or the system **102** can be a specialized computer (such as "BlackPC", or any other equivalent computing and recording device) that records events and activities triggered by the Hardware Input Devices such as mouse, trackpad, keyboard, video etc. together with the Operating System (OS) instrumentation service that performs real-time associations of AFP with user-actions into machine-readable data-sets.

[0054] These local input interfaces can be connected to the system **102** via any applicable human-computer interaction interfaces or I/O interfaces, wired or wirelessly. The human-computer interaction or I/O interfaces may include but not limited to, USB interface, Thunderbolt, PCI Express and others. The connections can be of wired connection or wireless connection.

[0055] The central module **220** is configured to determine a physical interaction ("hands-on" session) of the user based on the input signal received from the user action tracking (UAT) module **216** and the local input signal received from the computing device **108**. The central module **220** detects a plurality of potential threat actors on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof. The potential threat actors comprise a network security threat actor, a communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.

[0056] In an embodiment, the local input signals captured by the computing device **108** utilizes an out-of-band channel to transmit the second detected data **230** to a centralized management service unit for recording. Further, the centralized management service unit transmits the recorded second detected data to the detection units. In an exemplary embodiment, the detection units are selected from at least one of a Syslog, SFTP, commercial off-the-shelf threat detection sensor products or in-house analytics methods. By combining the local input signals and the AFP data and compiling them into one dataset at a prescribed format, the dataset can be used as a meaningful resource for determining if the system in used is under threat.

[0057] The present invention offers a dataset which associates the user-actions with Active Foreground Process (AFP). The AFP is the application that the user is working with at any

given instance. In an embodiment, the user-actions include but not limited to mouse clicks and key presses. The present invention may use a dedicated computing device **108** to track and record local Hardware Input Devices such as a mouse, trackpad, keyboard, audio, video, touch sensing etc. together with the Operating System (OS) instrumentation service that performs real-time associations of AFP with user-actions into machine-readable data-sets. The user-actions may lead to process creation or termination, which in turn cause AFP transitions, are also recorded. The OS instrumentation service records any mounting of logical-drives, which is a common vector to deliver malicious contents and documents. The OS instrumentation service records any changes in IPv4 and IPv6 address and associates it with the corresponding network-hardware MAC addresses. These data-sets can be used with any anomaly detection sensors to provide threat warning and response at the point in time when a victim/user falls for a malicious payload as with an externally delivered attack, and/or capturing concrete evidence of insider performing offensive acts.

[0058] FIG. 3 illustrates a flowchart **300** of the method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention. The method initiates with a step **302** of integrating a user action tracking (UAT) module into a plurality of global keystroke and mouse hooks of an operating system (OS) through an integration module. The method then includes the step of **304** detecting a plurality of user action events initiated by a user through a first detection module to generate a first detection data. The plurality of user action events comprises a mouse clicking event, a mouse movement event, and a key pressing event. Then the method includes the step **306** of capturing a foreground application process identifier (ID) on detection of the user action events through the user action tracking (UAT) module to generate a software signal based on the first detection data. The method includes the step **308** of recording the plurality of user action events and the application process identifier (ID) into an operating system (OS) logging facility through the user action tracking (UAT) module. Further, the method includes the step **310** of detecting a plurality of hardware inputs pertaining to a keyboard and a mouse through a second detection module to generate a second detection data. The second detection module is configured within a computing device.

[0059] The method then includes the step **312** of recording the second detected data and an operating system (OS) instrumentation service through the computing device to generate a

hardware signal. The method then comprises a step **314** of determining a physical interaction of the user based on the software signal received from the user action tracking (UAT) module and the hardware signal received from the computing device through a central module. The central module detects a potential threat actor on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof. The potential threat actor may include a network security threat actor, a communication threat actor, a software threat actor or any of the combination thereof.

[0060] FIG. 4 illustrates an operational flowchart **400** of the present method of rendering data-sets for differentiating a plurality of potential threat actors, in accordance with an embodiment of the present invention. FIG. 4 is explained in conjunction with FIG. 3. The Operating System (OS) is installed with the User Action Tracking (UAT) module. At step **402**, upon start up, the OS launches the UAT module (e.g. via a Windows or Linux service or other mechanisms). At step **404**, the UAT module hooks into OS global keystroke, mouse and all input events. By global, it means all process running under the user's control will be monitored. In an embodiment, the present invention utilizes *User32.dll* native functions to achieve this "global" hook into at least key-press, mouse click and other input activities. At step **406**, the UAT module is an event-driven program to respond whenever there are input events. After registering the event handlers with the OS with the global hook functions. Any input activities via the OS will trigger the event handler functions to perform the steps **410** and **412**. At step **408**, in case there are no events the UAT module becomes idle because the OS does not trigger the functions within the UAT module.

[0061] At step **410**, upon any input events, the UAT module responds by capturing the Foreground Application Process ID. The Foreground Application Process ID change with every new application launched and is managed by the OS. In an embodiment, the present invention uses *User32.dll* *GetForegroundWindow()* and *GetWindowThreadProcessId()* to obtain Process ID of the application process at the instance of key-press or mouse event. At step **412**, the UAT module records the type of user action event (either mouse or key inputs or others) together with the Process ID into the OS logging facility. The UAT module does not generate Process Activities events; they are either provided by the OS logging or additional software (e.g. Microsoft Sysmon) that taps into the OS to capture process related activities. The "link" here is the Process ID within the OS, or equivalent logs for foreground application will bear the same value as our UAT events. At step **414**, during an event of an

OS shut-down, the UAT module terminates along with the rest of the processes into the system (Step 416). The OS layer method can link the user actions (such as mouse and keyboard input) to the application process. A system can be compromised and remotely controlled by a backdoor tool, such that the backdoor process injects virtual/fraud user input actions into the OS layer.

[0062] The present invention uses the computing device that records local input activities independently from the OS to identify a “hands-on” session whereby a physical user is using the active OS session. Collectively, this data-set is “hardware signals” generated from the second detection module of the computing device, which contains details of all actual local inputs were triggered. The data-set of the UAT module are “software signals” which are anonymised to record only: { left/right-mouse-click, left/right-mouse-double-click, keypress, foreground-application-transition }. Anonymised in this case means the present system removed the absolute mouse positions and key values. Following are examples of the UAT log events.

[0063] Mouse Moving:

```
{ "EventTime": "2018-04-27 12:03:09", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 17074, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Mouse Moving\", \"ProcessId\": 6488 }", "Opcode": "Info", "EventReceivedTime": "2018-04-27 12:03:09", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

[0064] Mouse Stopped:

```
{ "EventTime": "2018-04-25 15:42:52", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 16179, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Mouse Stopped\", \"ProcessId\": 6488 }", "Opcode": "Info", "EventReceivedTime": "2018-04-25 15:42:52", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

[0065] Mouse (Single) Click:

```
{ "EventTime": "2018-04-25 15:42:52", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 16178, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Mouse Click Left\", \"ProcessId\": 6488 }", "Opcode": "Info", "EventReceivedTime": "2018-04-25 15:42:52", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

[0066] Mouse Double Click:

```
{ "EventTime": "2018-04-27 12:03:18", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 17085, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Mouse Double-Click Left\", \"ProcessId\": 13272 }", "Opcode": "Info", "EventReceivedTime": "2018-04-27 12:03:18", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

[0067] Key Press:

```
{ "EventTime": "2018-04-25 15:42:54", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 16180, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Key Press\", \"ProcessId\": 6488 }", "Opcode": "Info", "EventReceivedTime": "2018-04-25 15:42:54", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

[0068] Enter Key Press:

```
{ "EventTime": "2018-04-25 15:42:56", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 1, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 16188, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Enter Press\", \"ProcessId\": 6488 }", "Opcode": "Info", "EventReceivedTime": "2018-04-25 15:42:56", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }
```

15:42:56","SourceModuleName":"in","SourceModuleType":"im_msvistalog","DeviceVendor":"Microsoft","DeviceProduct":"EventLog"}

[0069] Foreground Application Transition:

{ "EventTime": "2018-04-27 12:03:11", "Hostname": "DESKTOP-PPDS4VR", "EventType": "AUDIT_SUCCESS", "SeverityValue": 2, "Severity": "INFO", "EventID": 2, "SourceName": "DataFuseUserActions", "Task": 0, "RecordNumber": 17076, "ProcessID": 0, "ThreadID": 0, "Channel": "Application", "Message": "{ \"Action\": \"Foreground Transition\", \"FromProcessId\": 6488, \"ToProcessId\": 13272 }", "Opcode": "Info", "EventReceivedTime": "2018-04-27 12:03:11", "SourceModuleName": "in", "SourceModuleType": "im_msvistalog", "DeviceVendor": "Microsoft", "DeviceProduct": "EventLog" }

[0070] FIG. 5 illustrates an exemplary flowchart **500** of the detection of the software signals and the hardware signals, in accordance with an embodiment of the present invention. FIG. 5 is explained in conjunction with FIG. 4. The system detects the presence of the two set of signals and reconciles the sequence of events to prove that the applications are triggered and used by a physical user, and was not controlled by a remote session or any automated process. It provides a decision process which can be implemented in most monitoring platforms without costly or complex methods. Given combined dataset of the local input log, APF and input log, following is the Table 1 to depict the three possibilities since both the present implies a hands-on session:

Possibilities	Software signals	Hardware signals
Remote controlled session or hardware tampering	Present	Missing
Operating System (OS) log disabled	Missing	Present
Computer system is turned off, or both software and hardware logging are disarmed	Missing	Missing

[0071] With UAT data-set, the centralize backend (e.g. a Security Incident and Events Management (SIEM) system) receives the data to detect and automate checks with the

central backend of the computing device to check for the status of the computer system. For the first possibility, if there are software signals and yet no hardware signals, it means the hardware has tampered. Otherwise, it is a remotely controlled session where user inputs are injected into the OS.

5 **[0072]** For the second possibility, if the present system is not receiving the entire fleet of computer systems that are being monitored, it means either a central logging failure due to technical fault or attack. If it is one or a few computer systems, it means deliberate disabling of OS logging.

10 **[0073]** For the third possibility of both types of logging disabled require linking with physical access events (e.g. logging badge access systems into SIEM for linking and detection). In case only one type of signal is present, the present system detects there are some forms of an anomaly. For instance, if hardware signals are present but UAT data-set is missing, it is likely that the attacker has disabled the OS logging facility, which is a common offensive tactic to prevent cyber-security monitoring platform from detecting
15 attackers' presence and activities. Even though the computing device requires a higher attack-factor (effort) to disarm, it is likely to require physical access to disrupt the hardware recording. If the hardware signals are missing, especially for one or a few computer systems, the present invention can infer that there is active hardware tampering. The UAT module data-set is recorded into the OS logging facilities and thus transferable to various
20 centralised log collectors and other detection units (e.g. over Syslog, SFTP, consumed by commercial off-the-shelf threat detection sensor products or in-house analytics methods), thus fulfilling the requirement for interoperability with existing solutions.

25 **[0074]** Thus the present system and method use the weakness of the conventional system logging as an indicator because privilege escalation exploits, along with source-codes, are now readily available in public domain (e.g. GitHub) and integrated into malware and backdoors. Further, the present system and method generate data for reliable detection of OS logging disablement. Furthermore, the present system and method generate data for reliable differentiation between overt and covert attacks in which the former could be performed by an insider in person, or a remote actor using some form of backdoor process
30 that can: (a) inject mouse and keyboard events into the OS; and (b) obtained remote view of what is on the target (victim's) screen.

[0075] While embodiments of the present disclosure have been illustrated and described, it will be clear that the disclosure is not limited to these embodiments only. Numerous modifications, changes, variations, substitutions, and equivalents will be apparent to those skilled in the art, without departing from the scope of the disclosure, as described in the claims.

CLAIMS:

1. A computer-implemented method comprising:

integrating, by one or more processors (202), a user action tracking (UAT) module (216)
 5 into a plurality of global keystroke and mouse hooks of an operating system (OS) through
 an integration module (212);

detecting, by one or more processors (202), a plurality of user action events initiated by a
 user through a first detection module (214) to generate a first detection data, wherein the
 plurality of user action events comprises a mouse clicking event, a mouse movement event,
 10 and a key pressing event;

capturing, by one or more processors (202), a foreground application process identifier (ID)
 on detection of the user action events through the user action tracking (UAT) module (216)
 to generate a software signal based on the first detection data;

recording, by one or more processors (202), the plurality of user action events and the
 15 application process identifier (ID) into an operating system (OS) logging facility through
 the user action tracking (UAT) module (216);

detecting, by one or more processors (202), a plurality of hardware inputs pertaining to a
 keyboard and a mouse through a second detection module (218) to generate a second
 detection data, wherein the second detection module is configured within a computing
 20 device (108); and

recording, by one or more processors (202), the second detected data and an operating
 system (OS) instrumentation service through the computing device (108) to generate a
 hardware signal.

2. The method according to claim 1 further comprises a step of determining, by one or more
 25 processors (202), a physical interaction of the user based on the software signal received
 from the user action tracking (UAT) module (216) and the hardware signal received from
 the computing device (108) through a central module (220), wherein the central module
 (220) detects a plurality of potential threat actors on determining an absence of at least one
 of the software signal, the hardware signal, and/or a combination thereof, wherein the

potential threat actors comprise a network security threat actor, a communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.

3. The method according to claim 1, wherein the plurality of hardware inputs comprises a mouse input, a keyboard input, a trackpad input, a touch input, a video input, and/or a combination thereof, further the plurality of hardware inputs lead to create the process or to terminate the process to initiate a plurality of active foreground transitions, wherein the computing device records the plurality of active foreground transitions.
4. The method according to claim 1, wherein the foreground application process identifier (ID) is a numeric or a test-string value to identify an active foreground process (AFP), wherein the recorded plurality of user action events and the application process identifier (ID) are transferable to a plurality of centralized log collectors and a plurality of detection units.
5. The method according to claim 1, wherein the hardware signals generated by the computing device (108) utilizes an out-of-band channel to transmit the second detected data to a centralized management service unit for recording, further the centralized management service unit transmits the recorded second detected data to the detection units.
6. A system (102) comprising:
 - a memory (206) to store machine-readable instructions pertaining to rendering data-sets for differentiating a plurality of potential threat actors; and
 - a processor (202) coupled to the memory and operable to execute the machine-readable instructions stored in the memory, wherein the processor (202) comprises:
 - an integration module (212) to integrate a user action tracking (UAT) module (216) into a plurality of global keystroke and mouse hooks of an operating system (OS);
 - a first detection module (214) to detect a plurality of user action events initiated by a user to generate a first detection data, wherein the plurality of user action events comprises a mouse clicking event, a mouse movement event, and a key pressing event;
 - the user action tracking (UAT) module (216) captures a foreground application process identifier (ID) on detection of the user action events to generate a software signal based on the first detection data;

the user action tracking (UAT) module (216) records the plurality of user action events and the application process identifier (ID) into an operating system (OS) logging facility;

a second detection module (218) to detect a plurality of hardware inputs pertaining to a keyboard and a mouse to generate a second detection data, wherein the second detection module (218) is configured within a computing device (108); and the computing device (108) records the second detected data and an operating system (OS) instrumentation service to generate a hardware signal.

7. The system according to claim 6, wherein the processor (202) further comprises a central module (220) to determine a physical interaction of the user based on the software signal received from the user action tracking (UAT) module (216) and the hardware signal received from the computing device (108), wherein the central module (220) detects a plurality of potential threat actors on determining an absence of at least one of the software signal, the hardware signal, and/or a combination thereof, wherein the potential threat actors comprise a network security threat actor, a communication threat actor, a software threat actor, a human threat actor, and a physical security threat actor.
8. The system according to claim 6, wherein the plurality of hardware inputs comprises a mouse input, a keyboard input, a trackpad input, a touch input, a video input, and/or a combination thereof, further the plurality of hardware inputs lead to create the process or to terminate the process to initiate a plurality of active foreground transitions, wherein the computing device records the plurality of active foreground transitions.
9. The system according to claim 6, wherein the foreground application process identifier (ID) is a numeric or a test-string value to identify an active foreground process (AFP), wherein the recorded plurality of user action events and the application process identifier (ID) are transferable to a plurality of centralized log collectors and a plurality of detection units.
10. The system according to claim 6, wherein the hardware signals generated by the computing device utilizes (108) an out-of-band channel to transmit the second detected data to a centralized management service unit for recording, further the centralized management service unit transmits the recorded second detected data to the detection units.