Exploring an Agent as an Economic Insider Threat Solution

Betina Tagle[1], Dr. Henry Felch[2]

[1]Colorado Technical University, Colorado Springs, USA

tina_tagle2000@yahoo.com

[2]University of Maine at Augusta, August, USA

henry.felch@maine.edu

Abstract. The insider threat is a security problem that is well-known and has a long history, yet it still remains an invisible enemy. Insiders know the security processes and have accesses that allow them to easily cover their tracks. In recent years the idea of monitoring separately for these threats has come into its own. However, the tools currently in use have disadvantages and one of the most effective techniques of human review is costly. This paper explores the development of an intelligent agent that uses already in-place computing material for inference as an inexpensive monitoring tool for insider threats. Design Science Research (DSR) is a methodology used to explore and develop an IT artifact, such as for this intelligent agent research. This methodology allows for a structure that can guide a deep search method for problems that may not be possible to solve or could add to a phenomenological instantiation.

Keywords: Insider Threat · Linear Genetic Programming · inference, software agent · learning.

1 Introduction

The insider threat continues to be a problem, but not due to a growth in occurrences. Ponemon Institute (2013) reports that the frequency of security breaches have had either "no change or decreased at 47%", but the severity has "increased by 52%". It is estimated that organizations typically experience a security breach yearly where more than half of them are caused by an employ for non-compliance [26]. The insider is internal to an organization behind the firewall as a trusted user [29]. Blackwell (2009) suggests that "The attacker uses a tool to perform an action that exploits a vulnerability on a target causing an unauthorized result that meets its objectives". Since insiders need a physical means to commit insider crime and logical security policies are technical settings that can be configured, then it is system monitoring that is the effective technique to determined non-compliant behavior [8].

The need is for a tool that learns on its own, as well as one that can promote cost friendly insider threat detection programs for organizations. This paper explores the intelligent agent as an effective and inexpensive tool for the detection of insider threats. First the issue of current tools available is discussed (Section 2). Next is a discussion on software agents (Section 3). Fourth is the presentation of the proposed intelligent agent (Section 4). Next is the research methodology to design and build the intelligent agent (Section 5). Lastly is the conclusion that includes future research intended (Section 6).

2 Issues with Hardware Tools

The hardware tools available that may be used for insider threat monitoring are costly to implement and manage both in funding and man hours. Typically different tools would have to be combined for complete effective monitoring for insider threats. There are other possible hardware solutions other than the examples in this section. However, the idea is that an effective insider tool has the ability to stand on its own and be specifically for internal threats so two examples of possibilities is used.

2.1 Intrusion Detection System (IDS)

The intrusion detection system (IDS) is to detect intrusion to a network or computer. The anomaly signature-based IDS is the traditional configuration, but requires constant updates with new signatures and these are increasing at a fast pace, for example, five years ago SNORT reported an increase from 1500 to 2800 over a two-year period [3, 5]. According to Axelsson (2000) high false-alarm rates plague the IDS. Configuring the audit policy within the IDS is complicated and would be especially so for insider threats [15]. This device may be able to be used for insider threat detection, but the performance continues to be an issue [11].

2.2 Expert Systems

Expert systems are considered the knowledgeable device and experts in problem solving. They make decisions based on the information it is given and inferences referred to as modus ponens, "given (p) q) and p we deduce q" [12]. Expert systems are not able to learn more than the knowledge base they are provided. The device is not configured for game-base programming or decision-theory computation [16]. Heuristics is available, however, this capability is useless unless it was programmed for such where the decision is based on the knowledge and rules it is given [2, 13].

3 A Software Approach

Software agents have the capability to be created and configured to learn, evolve, adapt, and have self-reliance in any environment. They are able to abide by them-

selves where constant intervention is not required [6, 22]. This is an important characteristic with so many users within an organization. Agents have the ability to understand their environment by learning as they complete their tasks. They are commonly known as micro software systems that interact in an environment to achieve a goal [28]. There are many types of software agents available that could be insider monitoring tools.

3.1 Agent Types

The adaptive agent is one that creates an explicit control plan for events that occur in dynamic and difficult computing infrastructure, it adapts to its environment on its own [10]. The intelligent agent has three attributes for its intelligence where they can perceive the changes in their environment, react and impact the conditions of the environment, and possess reasoning in order to infer, interpret, determine, and take action [10]. This type of agent has the capability to modify its own code since its code is part of the environment [20]. The artificial intelligent (AI) agent is where humanistic abilities are applied and at a high-level of cognition [19, 31]. With these different qualities available, the software agent is capable of insider threat detection.

4 Proposed Artifact

The Intelligent Agent Monitoring Inference Engine (IAMIE) is a one-tool solution to detect and respond to insider threats while making use of already in-place operating system (OS) components, in this research it is Windows. The software agent produces cognitive units of programming object classes (considered cognitive units hereafter) that monitor the actions and objects of an insider threat. These are translated from audit logs and user actions then assimilated into an instance of context within the context engine. Here the units are arranged with the logical elements of inferential mechanics to identify what has occurred. They are then sent to the inference engine without their knowledge factors to infer possible responses using Linear genetic programming (LGP). The LGP allows the intelligent agent to respond to actions within the context engine. It is a method that uses the I/O (inherent to the OS) to read and write ad hoc protocols into the agent's source file with the C# code. This provides the ability to execute all possible inferred responses for context input and output.

The inference engine processes input through predicate and fuzzy logic to associate knowledge to the input. The process of predicate logic is the loop where the starts and stops are coded into the loop, such as true and false. The loop checks for conditions over and over. The fuzzy logic is used to check for conditions based on reasoning from approximation and possess partial evidence of true where logic is between the range of 0 and 1, not completely true and false [17]. These processes of logic allow the structure for the agent to take into account Scruffy and NeuroEvolution of Augmenting Topologies (NEAT) computation. The Scruffy performs ad hoc inference processing, meaning math is not used; while, the NEAT option of inference does per-

form discrete measurements [14]. Code tells the inference platform how to process the structure.

The cognitive units are then sent to the cognitive engine to obtain their knowledge once the first phase of inference . The cognitive units obtain their knowledge from the cognitive engine through the engine's memory retrieval from Extensible Markup Language (XML). Once the new knowledge is obtained they will go back through the inference engine to manipulate the assessed knowledge into the cognitive units.

Finally the context engine converts the inferred, processed, and assimilated knowledge into new instances of the cognitive units for each implication made in the previous context. A response is displayed and an action is taken by the cognitive units. If it is determined that the action is a possible insider threat a simple alert message is sent to the PC of the user and the monitoring PC of the agent. Figure 1, Artifact Diagram, provides an overview of how the intelligent agent works.

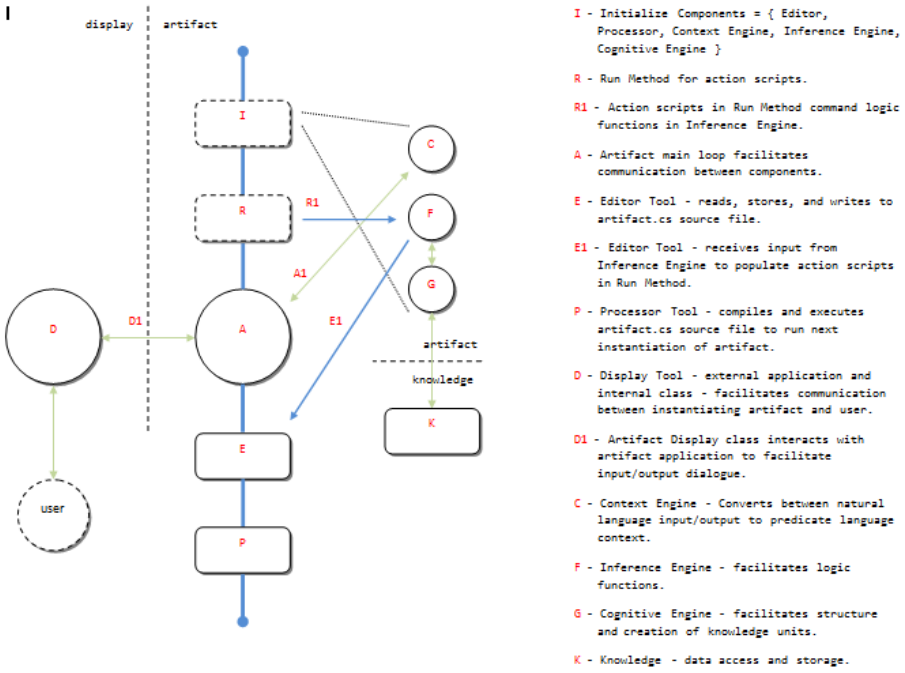

Fig. 1. Artifact Diagram

4.1 Artifact Example

The intelligent agent takes input that comes in the form of actual command line prompt inputs of natural language, it reads event log entries, or it watches for specific actions to occur that are established as rules in its programs. For example, the organization may have a specific folder that is restricted to only specific personnel since it contains the designs that gives a product its competitive advantages. The folder lists

of users that have permission to access, read, write, and execute the folder contents. The agent is a software program that resides on the computing device where the folder is located. The agent produces cognitive units that infer if accessing users are allowed to perform the actions they take with the folder. Flags set within the agent perform an alert message when certain activities in the folder are performed and if an unauthorized user is found it will send out an alert message. The key to the intelligent agent is that it watches the actions performed by trusted individuals.

5 Artifact Methodology

Design Science Research (DSR) is a method focused on the construction and improvement of an information Technology (IT) artifact or prototype and requires strong evaluation [21, 27]. Its use allows the whole of the research to add to the knowledge base [27]. For this agent research the DSR model chosen is the five-cycled General Design Cycle (GDC) model of awareness, problem, suggestion, development, and evaluation [25]. The frame used from DSR is the experiments and exploration to guide the research outcome to an operational prototype [24]. This research involves software development and the development life cycle (SDLC) chosen is an Agile approach of Rapid Application Development (RAD) using Iterative and Incremental phases (known as RADII within research documentation). Figure 2, Artifact Research Model, provides an overview of how DSR and a Software Development Life Cycle (SDLC) is used for developing the artifact.

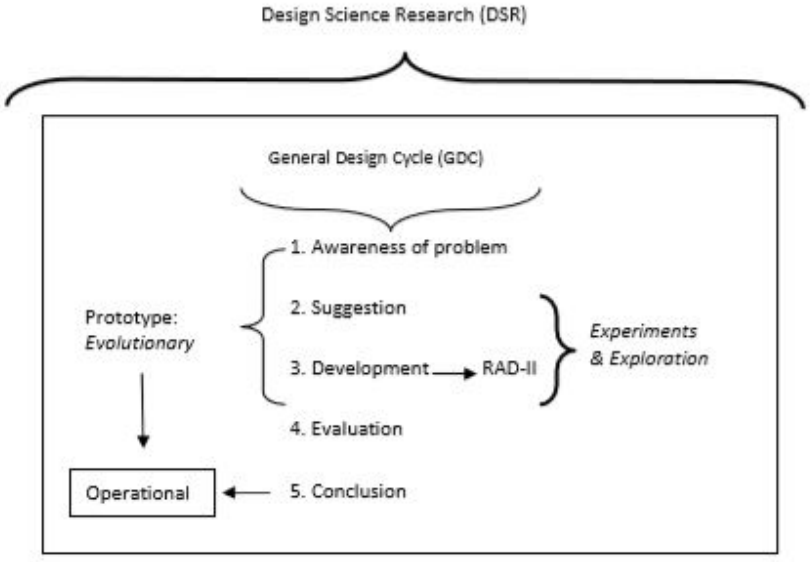

Fig. 2. Artifact Research Model

This intelligent agent research will include the dynamics of the suggested seven principles of DSR to address the rigor and processing, as follows: 1) Design as an Artifact; 2) Problem Relevance; 3) Design Evaluation; 4) Research Contributions; 5) Research Rigor; 6) Design as a Search Process; and, 7) Communication of the Research [26]. The same rigor to apply principles of the design of experimental research also must be applied to and by DSR [9].

5.1 Reason for DSR

DSR has the purpose of finding solutions to problems in technology that may not seem solvable or cannot be fixed with engineering alone [27]. The iterative and cycled approach of DSR is useful for practitioners and researchers who have some experience in using this approach [7]. The goal is to create a solution to a real-world problem and provide practical relevance. The key is the relevance of the problem to determine if research is needed beyond engineering. This is important since the activity of research is to contribute to the understanding of a phenomenon and add to the existing body of knowledge [1, 23]. The research goal is that the intelligent agent's cognitive engine can indeed write new instantiations of itself from LGP processed learning so it can better detect the invisibility of insider threats.

6 Conclusion

This research has defined the nature of the problem where the increase of insider threats is not the frequency of incidents, but the damage and its cost of one incident at one time. The the research is an intelligent software agent as an economical tool that effectively detects insider threats.

The DSR methodology is specifically to find solutions to problems within the field of technology. The ability for a research method that can go beyond engineering to answer to those problems that may not be solvable is invaluable. When a technology problem exists the first course of action is to decide if engineering can solve the issue or if DSR is applicable. Although DSR can go beyond creating a prototype or improvement by providing new knowledge, to force its use may not help the outcome.

In this specific agent research the use of DSR is valuable in order to take the IT artifact into operations. The intelligent agent processing information through the Context, Inference, and Cognitive engines makes possible emergent intelligent, where an agent's cognition grows as it performs its inferences [30]. When emergent intelligence is embraced in this specific research it exposes the availability of further implementation of the agent for different areas of technology. Research being performed in a parallel form is the knowledge that can be applied to the intelligent agent as an audit log reduction tool. The future research is the agent being structured for cloud application and open source processing.

References

1. Abecker, A., Bernardi, A., Hinkelmann, K., Kühn, O., & Sintek, M. (1997, March). Towards a well-founded technology for organizational memories. In Proceedings of the AAAI Spring Symposium on Artificial Intelligence in Knowledge Management (pp. 24-26).
2. Abraham, A. (2005). Rule-Based Expert Systems. Handbook of measuring system design.
3. Beg, S., Naru, U., Ashraf, M., & Mohsin, S. (2010). Feasibility of intrusion detection system with high performance computing: A survey. International Journal for Advances in Computer Science, 1(1).
4. Blackwell, C. (2009, April). Security architecture to protect against the insider threat from damage, fraud and theft. In Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies (p. 45). ACM.
5. Brox, A. (2002). Signature-based or anomaly-based intrusion detection: the practice and pitfalls. Multimedia information & technology, 28(1), 17-19.
6. Genesereth, M. R., & Ketchpel, S. P. (1994). Software agents. Commun. ACM, 37(7), 48-53.
7. Göbel, H., & Cronholm, S. (2012). Design science research in action-experiences from a process perspective.
8. Greitzer, F. L., & Frincke, D. A. (2010). Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation. In Insider Threats in Cyber Security (pp. 85-113). Springer US.
9. Hassan, N. R. (2014). Value of IS research: Is there a crisis?. Communications of the Association for Information Systems, 34(41), 801-816.
10. Hayes-Roth, B. (1990). Architectural foundations for real-time performance in intelligent agents. Real-Time Systems, 2(1-2), 99-125.
11. Jonnalagadda, S. K., & Reddy, R. P. (2013). A Literature Survey and Comprehensive Study of Intrusion Detection. *International Journal of Computer Applications*, *81*(16), 40-47.
12. Lindqvist, U., & Porras, P. A. (1999). Detecting computer and network misuse through the production-based expert system toolset (P-BEST). In Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on (pp. 146-161). IEEE.
13. Mauri, G., Filippini, M., & Gardin, F. (Eds.). (1990). Computational Intelligence, II (Vol. 2). Elsevier.
14. Minsky, M. L. (1991). Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI magazine*, *12*(2), 34.
15. Myers, J., Grimaila, M. R., & Mills, R. F. (2009, April). Towards insider threat detection using web server logs. In Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies (p. 54). ACM.
16. Parsons, S., & Wooldridge, M. (2002). Game theory and decision theory in multi-agent systems. Autonomous Agents and Multi-Agent Systems, 5 (3), 243-254.
17. Pearl, J. (1988). Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann.
18. Ponemon, L. (2013). Cost of Data Breach Study: Global Analysis. Poneomon Institute sponsored by Symantec.

19. Popplestone, R. J. (1970). An experiment in automatic induction. Machine Intelligence, 5, 203-215.
20. Ring, M., & Orseau, L. (2011). Delusion, survival, and intelligent agents. In Artificial General Intelligence (pp. 11-20). Springer Berlin Heidelberg.
21. Robey, D. (1996). Research commentary: diversity in information systems research: threat, promise, and responsibility. Information Systems Research, 7(4), 400-408.
22. Shoham, Y. (1993). Agent-oriented programming. Artificial intelligence, 60(1), 51-92.
23. Simon, H. (1996). The Sciences of the Artificial. MIT Press.
24. Tichy, W. F. (1998). Should computer scientists experiment more?. Computer, 31(5), 32-40.
25. Vaishnavi, V. K., & Kuechler Jr, W. (2007). Design science research methods and patterns: innovating information and communication technology. CRC Press.
26. Vance, A., Siponen, M., & Pahnila, S. (2012). Motivating IS security compliance: insights from habit and protection motivation theory. Information & Management, 49(3), 190-198.
27. von Alan, R. H., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS quarterly, 28(1), 75-105.
28. Wang, H., Liu, S., & Zhang, X. (2006, August). A prediction model of insider threat based on multi-agent. In Pervasive Computing and Applications, 2006 1st International Symposium on (pp. 273-278). IEEE.
29. Warkentin, M., & Willison, R. (2009). Behavioral and policy issues in information systems security: the insider threat. European Journal of Information Systems, 18(2), 101.
30. Wooldridge, M. (2009). An introduction to multi-agent systems. John Wiley & Sons.
31. Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. The knowledge engineering review, 10(02), 115-152.