IRTG Miniproposal Assignment

CSC 400: Problem Seminar
September 10, 2015
Ethan J. Johnson
Project Summary

As computer systems grow in scale and complexity, it has become increasingly difficult for programmers and end users alike to effectively grasp the concepts needed to understand the systems they are controlling. Faced with the impracticality of thoroughly understanding a system at the depth required to accurately predict and control its behavior, users operate from oversimplified models of how the system works. This problem is especially acute in the area of computer security, where it creates opportunities to exploit the inevitable security holes left open by trial-and-error problem solving. Data and system breaches have been steadily rising to epidemic proportions in recent years; the vast majority of these have been accomplished not with advanced attack techniques or zero-day vulnerabilities, but by exploiting simple, commonplace mistakes made by ordinary end users – the “human factor”.

There is a great need for tools, systems, and methods to enable end users to easily, practically, and reliably protect their computer systems from intrusion and misuse. Currently, one of the greatest challenges faced in computer security is enabling computers to programmatically detect and counteract attacks. Fundamentally, this is a difficult problem because attacks abuse the same functionality that was intended for legitimate use. Computers, by nature, do exactly what they have been programmed to do; malicious or benign uses are not distinguishable at this level. The best solutions thus far rely on static pattern matching and advanced heuristics to recognize common attacks. This strategy is increasingly ineffective at stopping dynamic human opponents, who can cause their attacks to diverge from the expected pattern with only a few lines of code. In order to counter a dynamic opponent, the computer’s response must also be dynamic.

To address the need for a tool for dynamic detection of attacks, we propose the development of a smart “security monitor” system based on machine learning that will observe and track actions within the system, using observation and training to learn to recognize benign behavior and “intuitively” sense dangerous conditions. Simple machine learning, which performs direct classification of sensory observations, is inadequate to detect complex attacks; the computer needs to apply multiple levels of machine learning to derive “big picture” conclusions. A great deal of work in this area has already been done within the field of cognitive computing, as has been most effectively demonstrated by IBM’s “Watson” system. Our research will investigate the currently untapped potential in applying these techniques to improve the detection of attacks.

Intellectual Merit: This research will explore the immense yet untapped potential of applying cognitive computing techniques to the field of computer security. New methods for effective detection of computer attacks will address a critical problem in computer security, as current attack trends shift from static exploit code to dynamic infiltration through misused legitimate system functions. Continued exploration of cognitive computing-based monitor systems will provide foundational insight into the further application of these principles to operating systems and computer architecture.

Broader Impact: This research will address an immediate need for a better tool to dynamically detect attacks on computer systems. Further developments in expanding the smart security monitor to enhance the analytical capabilities of the operating system at large will directly address the root problem of the manageability of computer interface complexity. By enabling users to more effectively communicate actions and intentions to the computer, computer systems will be made more effective and useful tools for users of all skill levels.
Project Description

1. Motivation and Expected Contribution

As computer systems grow in scale and complexity, it has become increasingly difficult for programmers and end users alike to effectively grasp the concepts needed to understand the systems they are controlling. Faced with the impracticality of thoroughly understanding a system at the depth required to accurately predict and control its behavior, users operate from oversimplified models of how the system works. When presented with challenges beyond the scope of these models, users resort to blind experimentation to push their way through tasks, finding “working” but suboptimal or accidental solutions that reinforce incorrect concepts and cause further complications down the road. This problem is especially acute in the area of computer security, where it creates opportunities to exploit the inevitable security holes left open by trial-and-error problem solving. Data and system breaches have been steadily rising to epidemic proportions in recent years; the vast majority of these have been accomplished not with advanced attack techniques or zero-day vulnerabilities, but by exploiting simple, commonplace mistakes made by ordinary end users – the “human factor”.

There is a great need for tools, systems, and methods to enable end users to easily, practically, and reliably protect their computer systems from intrusion and misuse. Currently, one of the greatest challenges faced in computer security is enabling computers to programmatically detect and counteract attacks. Fundamentally, this is a difficult problem because attacks abuse the same functionality that was intended for legitimate use. Computers, by nature, do exactly what they have been programmed to do; malicious or benign uses are not distinguishable at this level. The best solutions thus far rely on static pattern matching and advanced heuristics to recognize common attacks. This strategy is increasingly ineffective at stopping dynamic human opponents, who can cause their attacks to diverge from the expected pattern with only a few lines of code. In order to counter a dynamic opponent, the computer’s response must also be dynamic.

To address the need for a tool for dynamic detection of attacks, we propose the development of a smart “security monitor” system based on machine learning that will observe and track actions within the system, using observation and training to learn to recognize benign behavior and “intuitively” sense dangerous conditions. Simple machine learning, which performs direct classification of sensory observations, is inadequate to detect complex attacks; the computer needs to apply multiple levels of machine learning to derive “big picture” conclusions. A great deal of work in this area has already been done within the field of cognitive computing, as has been most effectively demonstrated by IBM’s “Watson” system. Our research will investigate the currently untapped potential in applying these techniques to improve the detection of attacks.

Using cognitive computing principles to build security monitoring platforms will better equip users, programmers, and administrators to build systems resilient to dynamic human attack. Because cognitive computing systems process and analyze information at a higher logical level than traditional computers, users can specify security policies more directly in natural language, eliminating the large conceptual gap introduced by having to encode policies in a computer-oriented language. As an example, many present-day network intrusion detection systems require users to specify policies in terms of regular expressions to recognize application protocols and attack signatures; this process is highly technical, and overwhelming to many network administrators attempting to deploy such systems. This results in misprogrammed intrusion detection systems that erroneously block benign network traffic (while doing nothing to block critical categories of malicious traffic), much to the frustration of end users attempting to
continue productivity as the “bugs” are worked out of the system by slow trial and error. A cognitive computing-based monitor system, by contrast, could be programmed with high-level specifications of application network protocols, and compare these to a natural-language security policy specified by the administrator, accepting or rejecting network traffic based on its conformance to that policy. Additionally, a cognitive computing-based system could passively monitor and analyze network traffic to produce a high-level report for the administrator summarizing activity, potentially informing network policy decisions, or to enable the administrator to spot a novel or creative attack in progress that might otherwise be “lost” in a deluge of low-level logs.

In the long term, we hope that the development and deployment of a cognitive computing-based security monitor system will provide key insights into how these techniques can be expanded to integrate more tightly with computer architecture. Specifically, integrating cognitive computing techniques into operating systems would allow users to more naturally and fluidly control their own computers, directly attacking the root problem of the manageability of computer complexity. Using its learning and natural language processing capabilities, the computer could draw upon a library of factual knowledge and algorithmic techniques to dynamically solve problems under the guidance of the human user. In present-day computer systems, a great deal of human effort is spent on the mechanics of tasks, not their essence.

As an example, consider the process of uploading and submitting a grant proposal via the NSF website. Currently, this process involves opening a Web browser, navigating to the site, logging in, finding the appropriate section of the site to receive this type of proposal, and finally uploading the proposal file through an HTML form, which involves browsing for the file on the local hard disk and clicking a button to trigger an upload action in the browser. Only the final steps in this process are essential the task of uploading the proposal; but, most likely, a great deal more time is spent manually “driving” the computer through the other, incidental tasks. Incidental tasks like these are just specialized and dynamic enough that they cannot be readily scripted in a programming language, yet they are repetitive enough to require no meaningful exercise of human intelligence. A “smarter” operating system, imbued with cognitive computing capabilities, would be able to perform these tasks with ease: if the computer had been previously programmed (“taught”) to understand the directions on the NSF website by consuming the site’s help documentation, it could apply standard operations (open program, browse to site, navigate to link, upload file) in an appropriate logical sequence to complete the upload. The user would, then, only need to ask the computer, in natural language, to (for instance) “upload the grant proposal file from last night to the NSF website”. The computer, unhampered by the delay of real-time human interaction “driving” the process, would be able to complete the task in seconds.

Continuing this example, if the NSF web server receiving the file were also equipped with an operating system utilizing cognitive computing techniques, the process could be further streamlined: the computer submitting the proposal could simply contact the server, and send it the proposal file accompanied by a succinct explanation (in natural language, or perhaps a more terse form suitable for inter-computer communication, yet retaining the high level of natural language) that it should be filed with a particular user’s grant proposal application. The server could interpret the instructions much as a human secretary would, and take action accordingly, or perhaps initiate a dialogue with the sending computer to clarify the request with further details if needed.

In summary, we believe that the application of cognitive computing techniques to the development of security monitors will provide a useful and much-needed tool for the detection
and prevention of attacks, and provide a foundation for further research into the more general application of these techniques to improve human-computer interaction with operating systems. Exploring these ideas will provide key input toward making the computer as a whole smarter, more secure, and more effectively usable for the general user.

2. Research Plan

To develop the proposed security monitor system, we propose to:

1. Investigate technological building blocks already available to the public for development of cognitive computing platforms. For instance, IBM provides a variety of public APIs and development kits for building applications on top of the Watson platform.

2. Use these identified publicly available technologies to develop a concept prototype of the security monitor system based on cognitive computing ideas. This prototype would focus on detection of a single class of attack that is realistic, yet nonetheless beyond the reliable detection capabilities of current intrusion detection and anti-exploit systems.

3. Building on the work done in steps #1 and #2, pursue the purchase of dedicated cognitive computing hardware and systems (such as the Watson platform), which are expected to be marketed in the near future by various manufacturers, and already available to well-connected research groups in the field.

4. Develop a practical and effective system for monitoring and protection of live computer systems and/or networks based on the systems acquired in step #3.

5. Explore the adaptation of the security monitor system into a tool that works in concert with a computer operating system, to allow the OS to process and execute natural-language commands based on an acquired library of factual knowledge and algorithmic techniques.

3. Intellectual Merit

If successful, the proposed work will harness the groundbreaking techniques of cognitive computing to solve real-world problems of critical importance in computer security. Current techniques for securing systems fall far short of real-world demands; there is no shortage of potential applications in the field for which a novel, dynamic technique such as that proposed here would be useful. Long-term exploration of these techniques in the context of operating systems provides interdisciplinary appeal in the field of human-computer interaction, which in turn would have ancillary benefits in computer security by allowing users to more effectively utilize computers as tools, ameliorating the misunderstandings and misconfigurations that are at the root of so many security weaknesses in the real world.
4. **Broader Impacts**

This research will address an immediate need for a better tool to dynamically detect attacks on computer systems, enabling end users to more easily, practically, and reliably protect their systems from intrusion and misuse. End users and system administrators in all capacities are increasingly concerned with the growing threat of cyberattack, as attackers’ capabilities eclipse defenders’; with current techniques, we are “losing” the battle to keep systems secure on the Internet, and large data breaches are increasingly seen as an unavoidable cost of modern Internet business. Geopolitical realities make it unlikely that the attackers behind these attacks will be meaningfully dealt with on legal fronts, making the need for better defensive technologies critical. The tools which we propose to create will thus provide strong benefits for society at large.

Long-term exploration of the benefits our techniques can provide in human-computer interaction will address a big-picture need for more usable computer systems. Even as computers have become more “user-friendly” and pervasive in society, their increased complexity often renders them a drudge to the end user rather than a valued tool. In many cases, it can be debatable whether the insertion of computer systems into a task workflow actually improves productivity, instead of hampering it. Frustration with the complexity of personal computers is, undoubtedly, a strong factor in the rise of mobile devices, which are valued as much for their relative simplicity as their portability; yet even these are only a marginal improvement on the core problem, and simplified interfaces carry a cost of reduced functionality. Harnessing the techniques proposed here to improve the ability of end users to manage the complexity of computer systems will allow them to better harness the often-untapped power of their computer devices for useful ends. Just as the introduction of end-user-oriented operating systems made the personal computer revolution possible, a significant increase in computer usability could unlock a new wave of discovery in the practical potential of computers as useful tools.