Cognitive Enhancements to Support Dependability[1]

Partha Pal, Franklin Webber, Richard Schantz
BBN Technologies
{ppal,fwebber,schantz}@bbn.com

Abstract

The threat of cyber-attacks is not limited to the boundary of information systems any longer. Safety and reliability of almost any system can be compromised by exploiting the vulnerabilities in the information systems that connect with or control them. Agile and ongoing manipulation of (redundant and diverse) system components, defense mechanisms and system resources is essential for surviving attacks and continuing operation. Cyber-defense administration—dynamic management of components, defense mechanism and systems resources—is therefore a current topic of significant interest to the dependability community. In this paper, we present our ongoing work on automated support for intelligent cyber-defense administration.

1. Introduction

Intrusion tolerance and survivability focuses on design, implementation and verification of information systems that can tolerate cyber attacks—i.e., maintain the Confidentiality (C), Integrity (I) and Availability (A) attributes (of information and information services) despite an adversary's attempt to subvert or compromise them. Fault-tolerance techniques and principles (e.g., redundancy, quorum based consensus etc.) are utilized in defending availability and countering corruption-attacks, but intrusion tolerance is not exactly the same as fault tolerance. Runtime adaptive management is one key differentiator.

Failures induced by malicious actions of an intelligent adversary may not follow any statistical distribution; may come in multiple numbers simultaneously; may range from a simple crash to timing failures and sophisticated Byzantine failures; and can manifest faster or slower than many accidental failures (because the adversary controls some aspects of the system). All these reinforce the need for advanced runtime manipulation of system components, defense mechanisms and resource controllers—in other words, sophisticated cyber-defense administration.

Cyber-defense administration is not the same as network administration. Network administrators treat the network as an omnibus system, and typically their network view does not include any deep understanding of the information systems and applications that use the network. Survivable systems on the other hand view the network as one of the (shared) resources that various information systems and applications need.

We have been developing a survivability approach that combines aspects of protection, detection and adaptive response (instead of focusing on fault detection, fault avoidance, or repair in isolation of each other) and involves dynamic manipulation of not just defense mechanisms or fault-tolerant protocols, but also the system's resources. Work to date has achieved the initial survivability objectives of containing the attacker's access, containing the spread of attack effects, isolating the compromised parts of the system and degrading the system's behavior gracefully (as opposed to sudden and complete disruption). We recently demonstrated a high-water mark survivable system (called the DPASA survivable JBI [1]) in multiple rounds of adversarial red team testing.

The survivable system achieved significant technical success (75% successful mission completion within stipulated time) against the intelligent and highly privileged adversary (pre-positioned attack code started as part of the system was run under direct control of the adversary). However, cyber-defense administration still significantly depends on human experts.

Cyber-defense that can only be administered by highly trained experts with deep designer and implementer level knowledge about the system is too expensive to be practical. In addition, significant

[1] This work is supported by DARPA in parts under AFRL Contract No. F30602-02-0134 and Navy Contract No. N00178-07-C-2003.

reliance on human factors can be a risk—over time and under the continuous vigilance required for cyber-defense administration, operator fatigue may impede expert behavior. As a consequence, insertion of the underlying cyber-defense technology into cost-conscious and widely deployed information systems where intrusions must be responded quickly and effectively (such as network centric military systems) becomes less attractive. A natural next step therefore, is to reduce the dependency on human expertise and increase the level of automation in cyber-defense. This implies that the defended system must perform much more of the knowledge intensive analysis and rapid decision making automatically. In our current research in the CSISM (Cognitive Support for Intelligent Survivability Management) project, we are developing an approach to do just that.

3. Challenges

Although typical survivability architectures include system management functionality where a reasoning mechanism for cyber-defense decision making could be placed, there are a number of challenges to overcome. First, the space in which the reasoning takes place is full of uncertainty. The initial knowledgebase representing potential attack paths, even some aspects of the system and its defenses may be incomplete. The events and observations that are reported from the system may be imperfect and even corrupt. Furthermore, the cyber-defense domain is more complex and multidimensional than other contexts (e.g., games and economics) that handle reasoning in the presence of uncertainty. Second, the window of opportunity for effective defensive response is typically small—this requires the decision-making mechanism to converge on a conclusion rapidly, while new events and observations are being reported continuously. To complicate the matter further, cyber-defense environment is much less forgiving than other fields where automated reasoning is used to govern physical systems (e.g., robotics, where the robot can see ahead of time that it needs to take a turn and a few missteps or bumps is not a problem). Third, the decision making strategies need to adjust to widely varying operating conditions (e.g., ranging from no alerts to 1000s of alerts per second) quickly (e.g., within a minute). The strategies also need to evolve as new symptoms emerge and as the adversary changes his strategy. Finally, there is the issue of striking a balance between automated support and human involvement. Clearly, expert developers cannot man every instance of an operational system, but on the other hand, total automation of cyber-defense administration seems out of reach. Regardless, relinquishing any degree of cyber-defense decision making will require a level of confidence in the automated mechanism, especially when decisions involve changing the state and configuration of the system (sometimes drastically). But available methodologies are utterly insufficient for certifying even simpler dynamic behavior in software systems.

In this project we are addressing the first three; the last one is being investigated in other ongoing work.

4. The CSISM approach

We have designed a reasoning approach that combines our past experiences as expert cyber-defense administrators as well as users of advanced cognitive tools and architectures. We are implementing this approach using the Soar [2] cognitive architecture and its underlying rule engine. We hypothesize that the resulting mechanism will be capable of expert-level cyber-defense decision-making. We will be calibrating its effectiveness in simulated as well as live-fire experiments over the next 2 years.

Key elements of this approach are summarized below.

1. Parallel exploration of multiple explanations from a number of complementary perspectives to deal with uncertain, imperfect and incomplete information.
2. Embedded risk-benefit evaluation from both the defense and adversarial points of view to handle uncertainty and changing attacker strategy.
3. Policy-based rapid containment response as an additional tier complementing situational short-cuts in reasoning for near real time response.
4. Treatment of events and observations as a continuous stream of inputs with external flow control to accommodate wide swings in operating conditions.
5. Machine learning based dynamic modification of defense parameters and decision-making strategies to facilitate continued currency and improvement.
6. An escape mechanism to draw operators' attention to a focused area when the reasoning mechanism is at an impasse or is churning.

To remain within the page limit, the remainder of this section will focus on the first bullet only.

As an illustration of multi-perspective exploration of multiple possibilities consider an event report sent by an application in host A that it cannot communicate with its peer in host B. Possible explanations of the reported symptom include: 1) the report originator on host A is lying, 2) something in the application at host B or the host B itself may have failed, or 3) some element(s) in between A and B have failed. Each of these high-level possibilities can be explored further, often intermixing with specific facts about the system and newer events eliminating existing possibilities or introducing new ones, ultimately leading to a quiescent state with a number of hypotheses that, with a level of confidence, explain what is reported so far.

To tame the complexity of the exploration space, we adopted a divide-and-conquer scheme that considers reported events in terms of four independent perspectives. The rules-space for each such perspective is smaller than the aggregated rule-space, and is further subdivided such that reasoning with generic rules precedes reasoning with context-specific rules. This is depicted in the left side of Figure 1. The perspective-based reasoning structure resulted from our understanding of how cyber-defense administrators behave and provides us a good starting point for implementation and evaluation of the new capability.

The first perspective considers bad behavior—is the report accusing some component, whether the accuser has any possible basis for such an accusation (i.e., any physical or logical connection between the two) etc. The second perspective uses an information flow point of view to deduce additional facts and inferences about the reported events (e.g., the sources or influencer of the purported bad behavior or corruption). The third perspective uses predefined attack trees trying to deduce the attacker's intention that match the observed symptoms and suspected corruptions. The fourth perspective considers the specific system and workflow context (e.g., an application is logging in, or publishing critical information) to specialize the inference further.

Rules involved in the reasoning process may embody common sense (e.g., if A cannot talk to B, either B or something in between is bad), insight gained from prior cyber-defense experience (e.g., the possibility that the reporter or something in the reporting host may be bad), generic knowledge about information flows in computer systems (e.g., how information flows impact each other and can propagate corruption or failure) as well as intimate knowledge about the system and its survivability architecture (e.g., what elements connects A and B, their vulnerabilities and failure profiles). This knowledge can be obtained from the system and its implementers, some may even be obtained in an automated manner (e.g., what are the constituent entities in the system, how are they connected, what are their properties and profiles etc.).

The result of multi-perspective exploration is a set of hypotheses. A hypothesis may point to a set of system components believed to be in some specific state. In the cyber-defense context, the suspected states may not always be verifiable (i.e. trace or log unavailable or incomplete). However, the reasoning process can select actions that can mitigate the suspected conditions (e.g., restore a file, isolate or block a component, etc.) based on encoded knowledge about the available defense mechanisms. If the condition persists despite the response, the reasoning process continues to explore other responses first, and then other hypotheses.

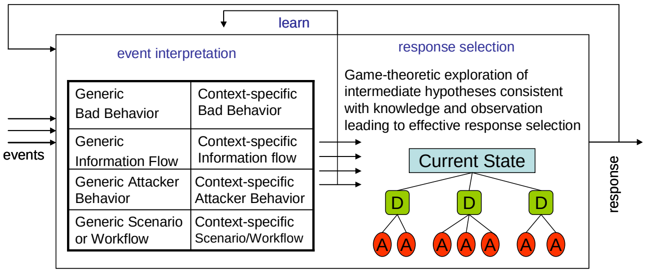
learn
event interpretation
response selection
Generic Bad Behavior
Context-specific Bad Behavior
Generic Information Flow
Context-specific Information flow
Generic Attacker Behavior
Context-specific Attacker Behavior
Generic Scenario or Workflow
Context-specific Scenario/Workflow
events
Game-theoretic exploration of intermediate hypotheses consistent with knowledge and observation leading to effective response selection
Current State
D
D
D
A
A
A
A
A
A
A
response

Figure 1: Reasoning from multiple perspectives

We fully anticipate situations when a response must be mounted before the reasoning mechanism exhausts all possibilities. Toward that end, our reasoning mechanism provides situation-dependant shortcuts to control processing time spent in each perspective. In addition, our approach includes a fast-acting local (to each host) mechanism for reversible containment responses (in case, the fast response turns out to be inappropriate).

Finally, there is a machine learning based mechanism observing all inputs and outputs of the reasoning process that attempts to continually improve the defense (e.g., adding a new rule about an attacker objective or modifying the utility of a response or instructing to cut through certain perspectives).

3.3. Key CSISM differentiators

Previous approaches have used matching attack signatures (see Cortex in [3]) or deviations from a specified system model (see AWDRAT in [3]) as response triggers. Models and signatures, “learned” (e.g., Immune Systems [4], Polygraph [5] or Learning & Repair Techniques in Self-healing Systems [3]) or otherwise developed are not new in cyber-defense. Their shortcomings are also well known. For example, a fixed signature based approach cannot detect novel attacks, learned signatures can be gamed, the complexity of building a high fidelity system model etc. In contrast, our reasoning is triggered by symptoms, and response is triggered by hypotheses that localize system malfunctions linked to these symptoms. An individual symptom, by itself, may not indicate an attack, but it is linked to some malfunction(s) in the system. We are able to localize the parts and type of malfunctions manifested in the symptoms without knowing how the attacker might have caused them because of the design and implementation details encoded within the reasoning engine. We anticipate encoding possible explanations of symptoms, both at a generic level as well as in a system specific level will be easier than building a comparable system model.

Another distinguishing factor of our approach involves spoofing of observations. Attack that corrupts some of a system's components can lead to reports that are false (e.g., accusing good components of being corrupt). Prior approaches ignored this issue and assumed all reported events to be true. Our approach explicitly considers the possibility that the observer may be lying while searching for hypotheses. Impact of spoofing is further reduced by combining data from multiple reports, by observing whether spoofing has already been used in the current attack and then learning to ignore it, and by look-ahead techniques to reduce the opportunity for future spoofing.

Responses supported in previous approaches (e.g., blocking system calls, or reversing the deviation from the model by repairing the data structure invariant, inserting a new filter etc.) are tightly coupled with the signatures or models used. In contrast, our response space is mapped to defense mechanisms and resource controllers present in typical survivability architectures.

Response selection based only on reactive knowledge (i.e., this action causes this effect) is not sufficient because there might be multiple candidates, the response may impact the ongoing operation of the system adversely, and may also open new opportunities for the adversary. Prior approaches to address this issue included using fixed utility functions, or “trial and refine/learn”. In contrast, we evaluate the utility of the candidate responses against potential counter-responses from the adversary for a (tunable) number of steps.

There are a number of other improvements over existing approaches. First, the provision to escape to operators and machine learning based dynamic modification of defense strategies provides a way to manage the impasses that may arise in this process. Second, to address the problem of an adversary attempting to game the reasoning process, we employ non-deterministic selection from equivalent classes (of hypotheses and responses) that are determined by predefined rules and could potentially be modified dynamically. Finally, we include a redundancy and consensus-based mechanism specifically to address the threat of direct attacks against the reasoning engine attempting to crash or corrupt it. This threat to the controlling reconfiguration engine was not considered in earlier approaches.

4. Conclusion

This work is at the confluence of cyber-defense and applied cognitive technology and pushes the limits in each. Whether the need for administrators with “developer-level” expertise can be alleviated by an automated system working in conjunction with regular operators with “operator-level” expertise is a critical question that must be resolved before the vision of the next generation of survivable systems—systems that can self-reconfigure and self-govern.—can be realized.

Our current work hypothesizes that it will be possible and that we have the basis for a workable design for such a capability; we will be testing this hypothesis through experimental evaluation of the mechanisms that we are currently developing.

5. Acknowledgement

The authors would like to thank DARPA, AFRL and NSWC for their continued support and involvement.

6. References

[1] J. Chong, P. Pal, M. Atighetchi, P. Rubel, and F. Webber, “Survivability Architecture of a Mission Critical System: The DPASA Example”, *Proc. 21st Annual Computer Security Applications Conference* (Dec. 2005), 495-504.

[2] J. Laird, A. Newell, and P. Rosenbloom, “Soar: Architecture for General Intelligence”, *Artificial Intelligence*: 33 (1987), 1-64.

[3] DARPA SRS Phase 1 Program Information (http://www.tolerantsystems.org/SRSProgram/srs_phase1.php) and SRS Phase 1 Program Abstracts (http://www.tolerantsystems.org/SRSProgram/20041124SRSsummaryApprovedforRelease.doc)

[4] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, “A Sense of Self for Unix Processes”, *Proc. IEEE Symposium on Security and Privacy (May 1996)*, 120-128

[5] J. Newsome, B. Karp, and D. Song, “Polygraph: Automatically Generating Signatures for Polymorphic Worms”, *Proc. IEEE Symposium on Security and Privacy* (May 2005), 226-241.