Enhancing Cyber Defense with Autonomous Agents Managing Dynamic Cyber Deception (Position Paper)

Cho-Yu Jason Chiang, Alex Poylisher and Ritu Chadha, Vencore Labs
<jchiang, apoylisher, rchadha>@vencorelabs.com
Hasan Cam, Army Research Lab
hasan.cam.civ@mail.mil

1 Introduction

Today's cyber defenses do not prevent all malicious intrusions, which compromise enterprise network environments by exploiting both human errors and system vulnerabilities. This state of affairs is likely to persist in the foreseeable future. Cyber attack tactics such as phishing e-mail, SQL injections, SMB exploits, cross-site scripting, etc. enable adversaries to inject malware into enterprise networks. After establishing an initial foothold, malware typically conducts reconnaissance, followed by lateral movements by compromising other hosts/systems on the network. Although existing cyber defense tools can detect a wide range of potential intrusion activities, a certain share of the generated alarms are false-positives. Since administrators of enterprise networks generally lack the resources to investigate *every* alarm, they set threshold values for different types of alarms to investigate only those more likely caused by intrusions, according to the enterprise threat models. This approach allows stealthy malware, in particular Advanced Persistent Threats (APTs), to stay undetected if they manage to generate no alarm or very few alarms with potential to be considered false-positives in a very long period of time.

We are investigating novel use of dynamic cyber deception techniques to aid the detection of activities of stealthy APTs. Our long-term goal is to develop autonomous agents engaged in investigations of every alarm that could indicate malware activities, rather than taking actions only when preset alarm threshold values are crossed. To set up the foundation for achieving the above goal, our current research spans the following areas: (i) minimizing the number of devices/hosts that are observable/accessible from any given host in order to reduce both the number of different types of alarms and the total number of alarms that could be raised by cyber security sensors; (ii) enabling automated creation of deceptive network views for each host and allowing such views to be changed on demand; (iii) using both low-interaction and proactive honeypots to generate illusive augmented false attack interfaces to increase the chance of detecting hidden threats; and (iv) investigating novel approaches for developing autonomous agents that manage the use of cyber deception schemes on-the-fly.

The remainder of this paper is organized as follows. Section 2 provides a discussion about cyber deception along with tactics that we have developed and plan to leverage, including the generation of deceptive network views to minimize the number of genuine hosts accessible by a host and the insertion of honeypots as fake hosts in deceptive network views. Section 3 discusses the currently considered approach for developing autonomous agents managing cyber deception, game theory problem formulation, space search heuristics, and our strategy for training autonomous agents. In Section 4 we present our progress to date with some discussions. We conclude this position paper in Section 5.

2 Cyber Deception

Cyber deception is being considered as an approach to boost cyber defense [1]. In general, cyber deception approaches provide false information to (hidden) adversaries without significant effect on the normal cyber activities in enterprise networks. There are multiple research areas under cyber deception, such as camouflage, disinformation, decoy, etc. [2] With respect to building autonomous cyber deception agents, our current focus is on making use of the following tactics: Moving Target Defense [3] (camouflage) and honeypots [4] (decoy). We follow an SDN-based [6] deception approach that can generate network views for individual hosts, such that: (i) hosts may have very different network addresses, even though they are connected to the same physical switch, (ii) any network service, (e.g., DNS, e-mail, HTTP, etc.), is accessed by each host at a different IP address, and (iii) hosts appearing in a view may be true hosts or honeypots. Next, we describe the SDN-based MTD and honeypots we have been investigating.

2.1 SDN-based Cyber Deception (Camouflage)

Moving target defense has been a heavily researched topic as it provides camouflage for enterprise networks. In general, MTD techniques change network and device configuration settings on the fly, with an objective to both confuse (i.e., slow down/dissuade) adversaries and increase the chance of detecting further adversarial activities. In our research, we are building on top of ACyDS [10], an adaptive cyber deception system that provides a unique virtual network view to each host in an enterprise network. A host's view of its network, including the subnet topology and IP address assignments of reachable hosts and servers, generally does not reflect the actual network configuration and is different from the view of any other host in the network. For example, all hosts may send DNS queries to the same DNS server, but each host sends its requests to an IP address unique for that host, i.e., the address assigned to the DNS server is valid only in the network view that a host observes. ACyDS can change a node's network view with the desired properties on demand. It enforces dynamic network view changes to invalidate, to a desired degree, the intelligence collected by the adversary from prior reconnaissance activities, as subnet topology and IP address assignments can be changed in every view update. In a nutshell, ACyDS's deception approach (i) deters/delays/directs the adversary's reconnaissance activities, (ii) encumbers collusion if multiple hosts have been compromised, and (iii) increases the likelihood and confidence of detecting the presence of intruders. ACyDS leverages OpenFlow [7] switches and controllers (Open vSwitch [8] and RYU SDN controller [9] in the current prototype) to consistently handle the most common network layer protocols used in the enterprise networks, currently including ARP, DHCP, DNS, UDP, TCP, and ICMP—by dynamically modifying IP header fields at the SDN switch according to the installed flow rules that implement network views for individual hosts. ACyDS also allows masking of multicast/broadcast messages such as ARP queries that allow malware to map a network *passively*; meanwhile similar but fake multicast/broadcast messages could be sent to direct malware to take actions against honeypots.

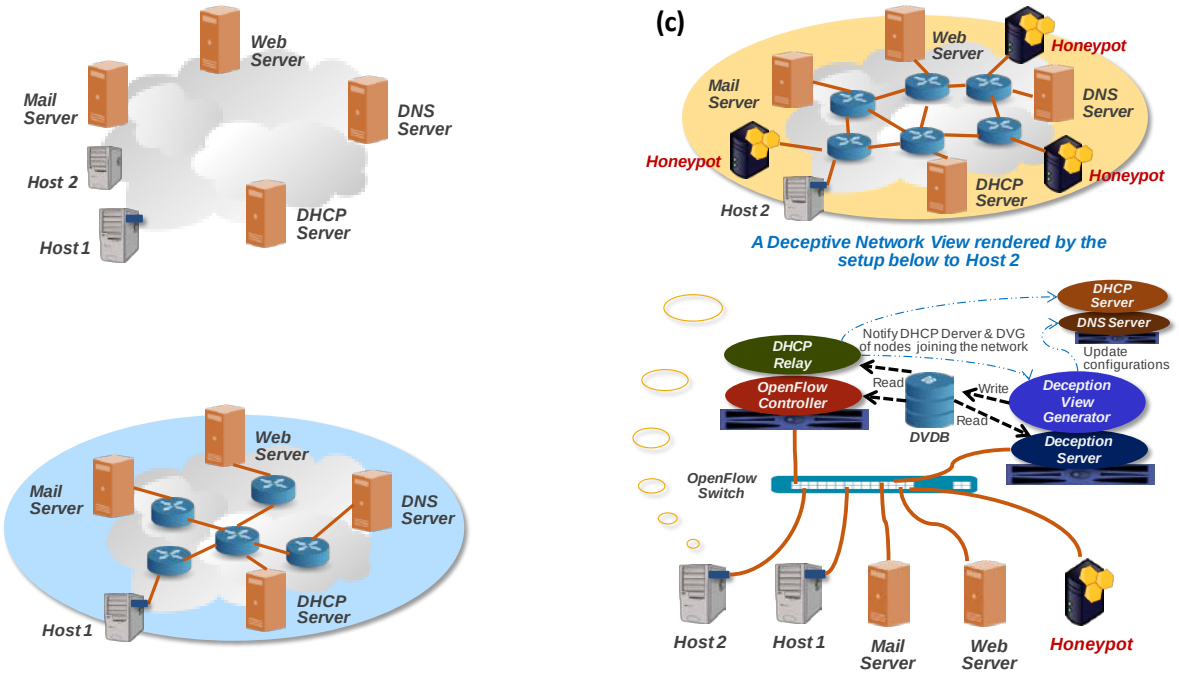

Figure 1. ACyDS enables rendering of different network views to hosts on the same physical network

We illustrate the concept of ACyDS using Figure 1. We assume that both Host 1 and Host 2 are on the same network, but are presented with two entirely different network views. Host 1's network topology is different than Host 2's, and the IP address of a given server (e.g., HTTP), is different in the two views. In the figure, Host 2's view includes 3 honeypots, while Host 1's view has none. ACyDS's capability is enabled by the various components shown in Figure 1(c). Readers interested in the functions of the components and ACyDS implementation are referred to [10]. We have successfully implemented a proof-of-concept prototype of ACyDS software and demonstrated its function in 2016.

2.2 Honeypots (Decoy)

Honeypots are fake hosts that are not used to perform any production task. Their primary purpose is to direct intruders in their lateral movement from already compromised devices to provide more information to network defenders for determining whether certain hosts have been compromised (low-interaction honeypots). A secondary purpose is to keep the adversary engaged in fruitless activity (high-interaction honeypots). There has been a wide array of research activities in this area, mostly are low-interaction honeypots. Open source (e.g. *honeyd* [5]) and commercial honeypot products facilitating honeypot deployment and customization are also available. In our work, we plan to make enhancements to *honeyd* such that it can (i) send and receive fake traffic flows with specific purposes such as informing others via broadcast messages that it is running a particular service and (ii) distinguish whether incoming packets are from other honeypots or the potential adversary, based on decoding of certain header fields of packets.

2.3 Combination of Camouflage and Decoy

We consider the combination of camouflage and decoy to enhance cyber security defense a promising research direction. Existing technologies allow static setup of camouflage and decoy; however, once the adversary recognizes the scheme and the usage pattern, both tactics become futile. However, providing

dynamic and variable camouflage and ever-changing decoy deployment/placement is a challenge because of the need to ensure consistent management of changing defense with minimal impact on normal network operations. Needless to say, humans are ill-fit for this task in any real network, but an autonomous agent that is able to manage various deception techniques on-the-fly has the potential of significantly raising the level of difficulty for adversarial reconnaissance.

3 Approach

To manage the combination of camouflage and decoy, an autonomous agent needs to assess changes to the current system state, determine the next moves, and then adjust the configuration of deception tactics. Our plan is to develop such agents by using the following approach, illustrated in Figure 2. Thanks to the isolated network environment that ACyDS provides for each host, the problem can be formulated as a two-player game between the dynamic deception management agent and a potential adversary on a host (or a set of hosts). As shown in the figure, the agent receives sensory input from hosts, honeypots, and SDN controller. Since the network view is controlled by the agent, the number of nodes and therefore the amount of sensory input is under its control, too. Based on the sensory input the agent receives, it may decide to keep the current network view for a given host intact, or it can produce a new view using deception tactics such as changing IP addresses of all the nodes in the network view of a host, inserting additional honeypots into a network view, configuring honeypots to change their behaviors such as becoming more interactive and running a database server containing false data, and so on. In particular, new tactics are used to further investigate potential intrusion events. For example, an alarm is raised by a real host about a failed connection from another host to a closed port. This could be accidental and hence a false alarm, or it could be an intentional probing attempt by malware on the probing host. In addition to collecting information from the probing host to find out which process attempts the connection, the agent may remove the probed node from the view of the probing host and replace it with a proxy or a honeypot, or engage another proactive honeypot to communicate with the probing node, which may get the attention of the malware, if present on the probing host, to probe the newly found honeypots. If further honeypot probing happens, it would be a strong indicator that malware is present on the probing node, and additional moves may be planned by the agent.

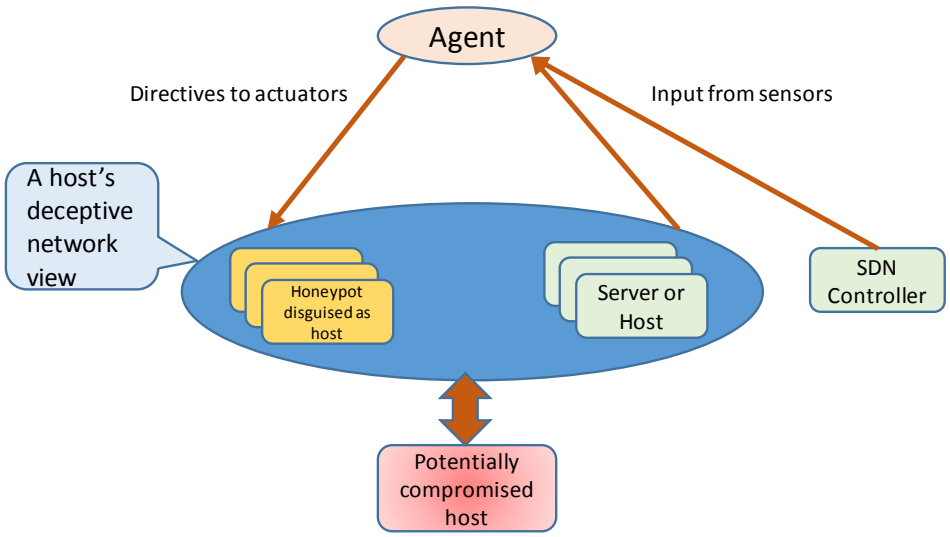

Figure 2. Agent interacting with a potentially compromised host through a deceptive network view

As the reader can easily extrapolate, the game has a large number of possible moves and states. Given the large state space, we plan to explore the following methodology to train the agent, allowing it to grow its capability over many simulated games. This will also help us achieve an understanding about the Nash equilibrium for different state spaces we consider. Our training approach works as follows.

First, with the help of human experts, we plan to build a semi-cognitive synthetic adversary that specializes in reconnaissance and lateral movement. We will start with the assumption that the adversary is a stealthy APT, and it makes prudent moves following a given adversary model. Under uncertainties this adversary may make probabilistic decisions. For the deceptive agent, we plan to explore the concept of deep learning [12] by starting with a set of deceptive tactics each with multiple configurable parameters and a basic operation model. The model is expected to evolve through the learning process, which is aided by simulation in an ACyDS network environment. Like many other reported deep-learning research, we also plan to review the generated model in the learning process and explore ways to insert experts' input to guide the learning process to a certain extent. The simulation will start with a Monto Carlo Tree Search to perform a search of the game tree that has a gigantic state space. Each game will end either when the agent successfully identifies the adversary or the adversary successfully compromises another host. This game, in a sense, can be made somewhat similar to the games of Chess and Go, in which two players exchange moves. Since the search tree state space is large and cannot be fully traversed, similar to the strategy adopted by Google's AlphaGo [13] we will develop evaluation functions to assess possible outcomes of subtrees that have only been partially traversed. The agent's learning process will be supported by the following three metrics, among possibly others: (i) the size of the true attack surface that is exposed to adversary (to be minimized), (ii) the likelihood that an intruding adversary identifies the true attack surface and successfully stages attacks (i.e., by making a lateral move to compromise another host), to be minimized, and (iii) the cost of moves. As different moves have different cost, cost may be evaluated by using multiple metrics, including CPU cycles, memory, number of IP addresses needed, number of rules to be used in the SDN switch, and time taken to switch to a different deception tactics.

4 Research Progress and Discussion

To develop an agent for managing cyber deception on-the-fly, we adopt a multi-phased approach. In Phase 1 we enumerate the following: (i) possible reconnaissance actions that could be used by adversary, with and without sending probes, (ii) sensor information to be collected from hosts by the agent and (iii) deception tactics that the agent may take, along with the parameters to configure and ranges of the parameters. In Phase 2 we plan to develop a stealthy semi-cognitive adversary, a deception-managing agent, and realistic scenarios first in simulation and then for the CyberVAN testbed [11] (a high-fidelity testing and evaluation environment) that allow an adversary (synthetic or human) and a deception agent to be engaged in many rounds of games. In Phase 3 we plan to tune the training environment for the agent to learn from the game, and investigate how to improve learning efficiency and train an effective agent making decisions no worse than a human cyber expert.

We are currently at the end of Phase 1, and part of the Phase 2 work is under way, including the development of a stealthy adversary.

5 Summary

In this paper, we outline the goals of managing dynamic cyber deception, its basic mechanisms and an approach to creating an autonomous agent to automate the task. Thanks to the use of ACyDS to limit the

scope of the problem space, we are able to significantly reduce the state space compared to a non-ACyDS network environment. Our plan is to continue this research under the U.S. Army Cyber Security Collaborative Research Alliance (CRA) program.

6 Acknowledgement

The authors want to thank the U.S. Army Research Laboratory Cyber Security Collaborative Research Alliance program for supporting this research.

7 References

1. K. L. Tan, “Confronting cyberterrorism with cyber deception”, Thesis, Naval Postgraduate School, 2003.
2. David Poarch, David O'Leary, Jason Nelson, Anne Grahn, ”Six ways to deceive cyber attackers,” http://focus.forsythe.com/articles/337/6-Ways-to-Deceive-Cyber-Attackers.
3. Sushil Jajodia, Anup K. Ghosh, Vipin Swarup, Cliff Wang, X. Sean Wang, ed., “Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats”, Springer Book, 2011.
4. Lance Spitzner, “Honeypots: Catching the Insider Threat”, Proceedings of Computer Security Application Conference, 2003.
5. Honeyd, http://www.honeyd.org”
6. SDN, Software Defined Networking, https://www.opennetworking.org/
7. OpenFlow. http://https://www.opennetworking.org/sdn-resources/openflow, retrieved on 4/9/16.
8. Open vSwitch. http://openvswitch.org/
9. RYU, https://osrg.github.io/ryu/
10. Cho-Yu J. Chiang, Yitzchak Gottlieb, Shridatt J. Sugrim, Ritu Chadha, Constantin Serban, Alex Poylisher, Lisa Marvel and Jon Santos, “*ACyDS, An Adaptive Cyber Deception System*” MILCOM 2016.
11. Ritu Chadha, Thomas Bowen, Cho-Yu J. Chiang, Yitzchak M. Gottlieb, Alex Poylisher, Angelo Sapello, Constantin Serban, Shridatt Sugrim, Gary Walther, Lisa Marvel, Allison Newcomb, and Jonathan Santos, “*CyberVAN: A Cyber Security Virtual Assured Network Testbed*”, MILCOM 2016.
12. Jurgen Schmidhuber, “Deep Learning in Neural Networks: An Overview”, Technical Report IDSIA-03-14, 2014.
13. D. Silver et al., “Mastering the game of Go with deep neural networks and tree search”, Nature 529, 484-489, 2016.
14. Tao Ye and Shivkumar Kalyanaraman, “A recursive random search algorithm for large-scale network parameter configuration”, Proceedings of 2003 ACM SIGMETRICS Conference, 2003.