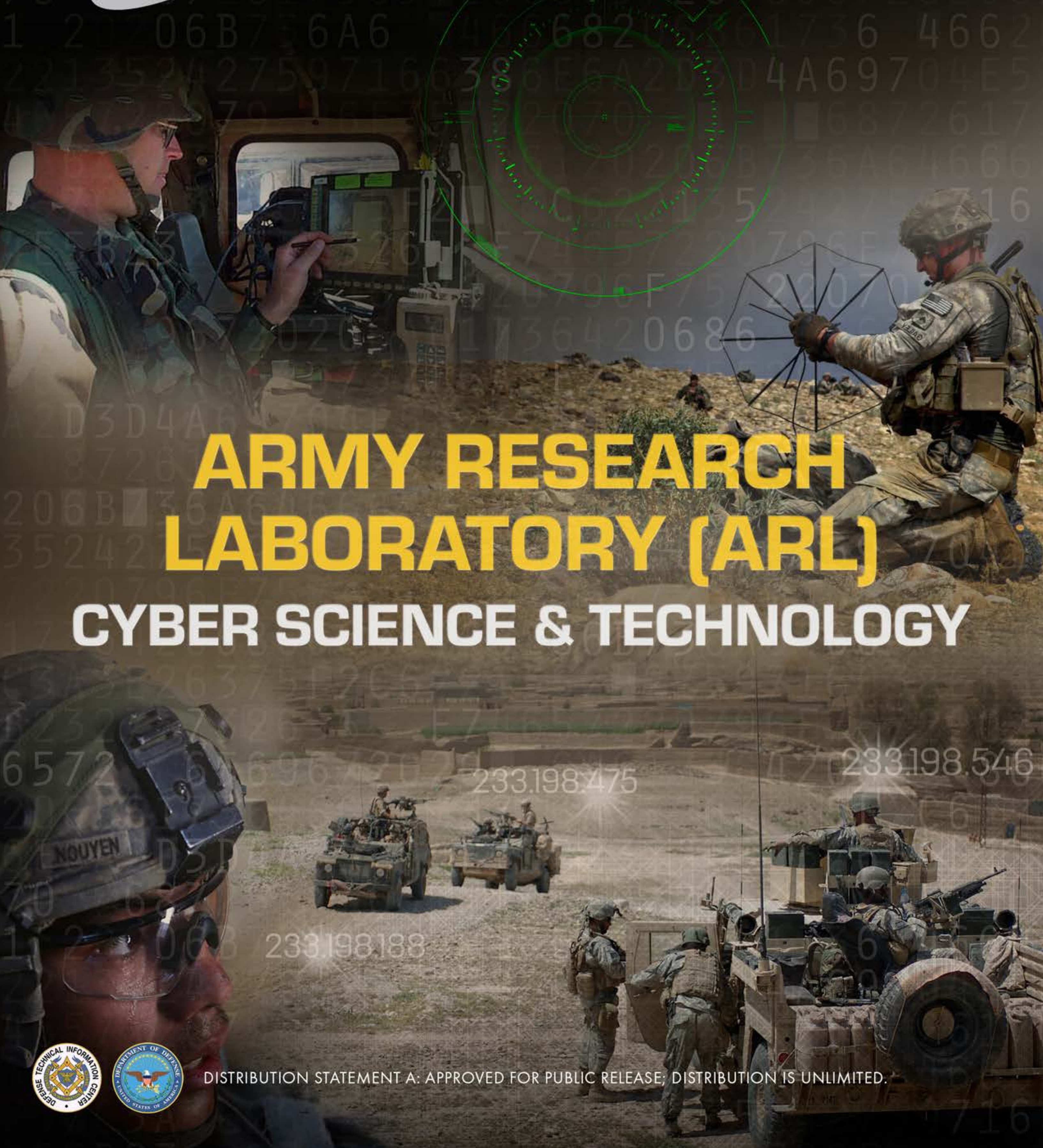
VOLUME 5 NUMBER 1
DECEMBER 2016

CSIAC JOURNAL

A Quarterly Publication of the Cyber Security & Information Systems Information Analysis Center



ARMY RESEARCH LABORATORY (ARL)

CYBER SCIENCE & TECHNOLOGY

CSIAC Cyber Security & Information Systems Information Analysis Center

ABOUT THE CSIAC

As one of three DoD Information Analysis Centers (IACs), sponsored by the Defense Technical Information Center (DTIC), CSIAC is the Center of Excellence in Cyber Security and Information Systems. CSIAC fulfills the Scientific and Technical Information (STI) needs of the Research and Development (R&D) and acquisition communities. This is accomplished by providing access to the vast knowledge repositories of existing STI as well as conducting novel core analysis tasks (CATs) to address current, customer focused technological shortfalls.

OUR MISSION

CSIAC is chartered to leverage the best practices and expertise from government, industry, and academia in order to promote technology domain awareness and solve the most critically challenging scientific and technical (S&T) problems in the following areas:

- Cybersecurity and Information Assurance
- Software Engineering
- Modeling and Simulation
- Knowledge Management/Information Sharing

The primary activities focus on the collection, analysis, synthesis, processing, production and dissemination of Scientific and Technical Information (STI).

OUR VISION

The goal of CSIAC is to facilitate the advancement of technological innovations and developments. This is achieved by conducting gap analyses and proactively performing research efforts to fill the voids in the knowledge bases that are vital to our nation. CSIAC provides access to a wealth of STI along with expert guidance in order to improve our strategic capabilities.

WHAT WE OFFER

We provide expert technical advice and assistance to our user community. CSIAC is a competitively procured, single award contract. The CSIAC contract vehicle has Indefinite Delivery/Indefinite Quantity (ID/IQ) provisions that allow us to rapidly respond to our users' most important needs and requirements.

Custom solutions are delivered by executing user defined and funded CAT projects.

CORE SERVICES

- Technical Inquiries: up to 4 hours free
- Extended Inquiries: 5 - 24 hours
- Search and Summary Inquiries
- STI Searches of DTIC and other repositories
- Workshops and Training Classes
- Subject Matter Expert (SME) Registry and Referrals
- Risk Management Framework (RMF) Assessment & Authorization (A&A) Assistance and Training
- Community of Interest (COI) and Practice Support
- Document Hosting and Blog Spaces
- Agile & Responsive Solutions to emerging trends/threats

PRODUCTS

- State-of-the-Art Reports (SOARs)
- Technical Journals (Quarterly)
- Cybersecurity Digest (Semimonthly)
- RMF A&A Information
- Critical Reviews and Technology Assessments (CR/TAs)
- Analytical Tools and Techniques
- Webinars & Podcasts
- Handbooks and Data Books
- DoD Cybersecurity Policy Chart

CORE ANALYSIS TASKS (CATS)

- Customer tailored R&D efforts performed to solve specific user defined problems
- Funded Studies - $500K ceiling
- Duration - 12 month maximum
- Lead time - on contract within as few as 6-8 weeks

CONTACT INFORMATION

100 Seymour Rd.
Suite C102
Utica, NY 13502

1 (800) 214-7921

info@csiac.org

/DoD_CSIAC

/CSIAC

/CSIAC

DoDIACs Department of Defense Information Analysis Centers

ABOUT THE JOURNAL OF CYBER SECURITY AND INFORMATION SYSTEMS

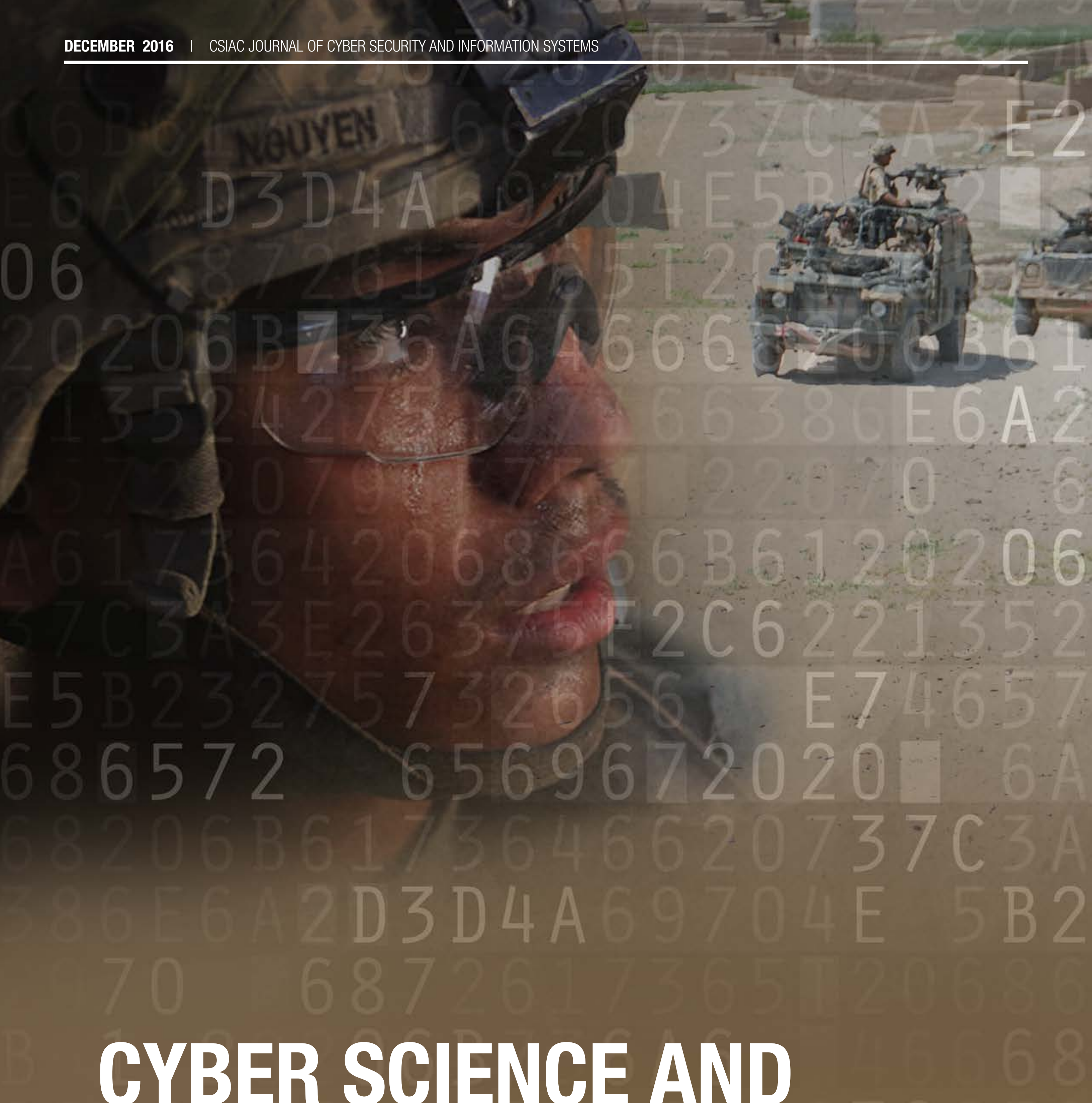
ABOUT THIS PUBLICATION

The Journal of Cyber Security and Information Systems is published quarterly by the Cyber Security and Information Systems Information Analysis Center (CSIAC). The CSIAC is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC). The CSIAC is technically managed by Air Force Research Laboratory in Rome, NY and operated by Quanterion Solutions Incorporated in Utica, NY.

ARTICLE REPRODUCTION

Journal of Cyber Security and Information Systems
Cyber Science and Technology at the ARL
December 2016

— IN THIS ISSUE —



CYBER SCIENCE AND TECHNOLOGY AT THE ARMY RESEARCH LABORATORY

BY: Alexander Kott, Army Research Laboratory (ARL)

THE U.S. ARMY RESEARCH LABORATORY (ARL) *received the first salvos in the battle for cybersecurity as early as three decades ago. In terms of technology history, it was an astonishingly long time ago. Before most people ever heard of the Internet. Before there were web browsers. Long before the smartphones. Back in 1986, the laboratory withstood attacks by Markus Hess, a Soviet-sponsored hacker who had successfully penetrated dozens of U.S. military computer sites. In his bestselling book, The Cuckoo's Egg, the pioneering U.S. cyber defender, Cliff Stoll, describes how he monitored the hacker's networks activities in the fall of 1986: "He then tried the Army's Ballistic Research Lab's computers in Aberdeen, Maryland. The Milnet took only a second to connect, but BRL's passwords defeated him: he couldn't get through" (Stoll 1989).*

Continued on next page ❯❯❯

Two years later, the laboratory faced the legendary Morris Worm. Around midnight on November 3, 1988, system managers at the Army's Ballistic Research Laboratory noticed their computers slowing down to a crawl as the worm stole precious computing processing time. Fearing a foreign attack, they pulled their computers off the nationwide network predating the Internet, called ARPAnet." (Hess, 2016)

The Army's Ballistic Research Lab, an ancestor of ARL, was the home of the ENIAC, the world's first electronic digital computer in 1946. It was also where the "ping" program was written in 1983, and where many other milestones of computing and networking took place. The encounters with the Soviet-sponsored hacker and with the Morris Worm were among such milestones.

Since those early beginnings, the history of ARL's efforts in cyber defense was exciting and challenging (Fig. 1). Although ARL is the Army's corporate laboratory that focuses on fundamental and early applied research (in the Department of Defense lingo – the research of 6.1 and early 6.2 types), the fundamental science endeavors are closely integrated with extensive operationally-oriented programs. These range from providing continuous cybersecurity defense services to multiple organizations, as well as cyber survivability and vulnerability analysis of Army systems.

Continuous cybersecurity defense services to multiple organizations

A remarkable feature of ARL's business model is the great degree of collaboration with the academic community. One example is the Cyber Collaborative Research Alliance (CRA) (see the article "Cyber Collaborative Research Alliance" in this issue) that brings together, in closely integrated collaborative projects, ARL scientists with academic researchers from dozens of U.S. universities. Cyber CRA aims to develop the fundamental science of cyber detection, risk, agility, as well as the overarching challenge of human factors in cyber security. Similarly, the Network Science Collaborative Technology Alliance (see **http://www.ns-cta.org/**) integrates ARL and academic research efforts towards a broad understanding of how multi-genre networks of humans and information and communications devices influence each other and undergo complex dynamic transformations.

ARL collaborations are not limited to U.S. universities. ARL is also actively engaged with international partners. ARL's Open Campus business model (**http://www.arl.army.mil/**) helps such wide-ranged

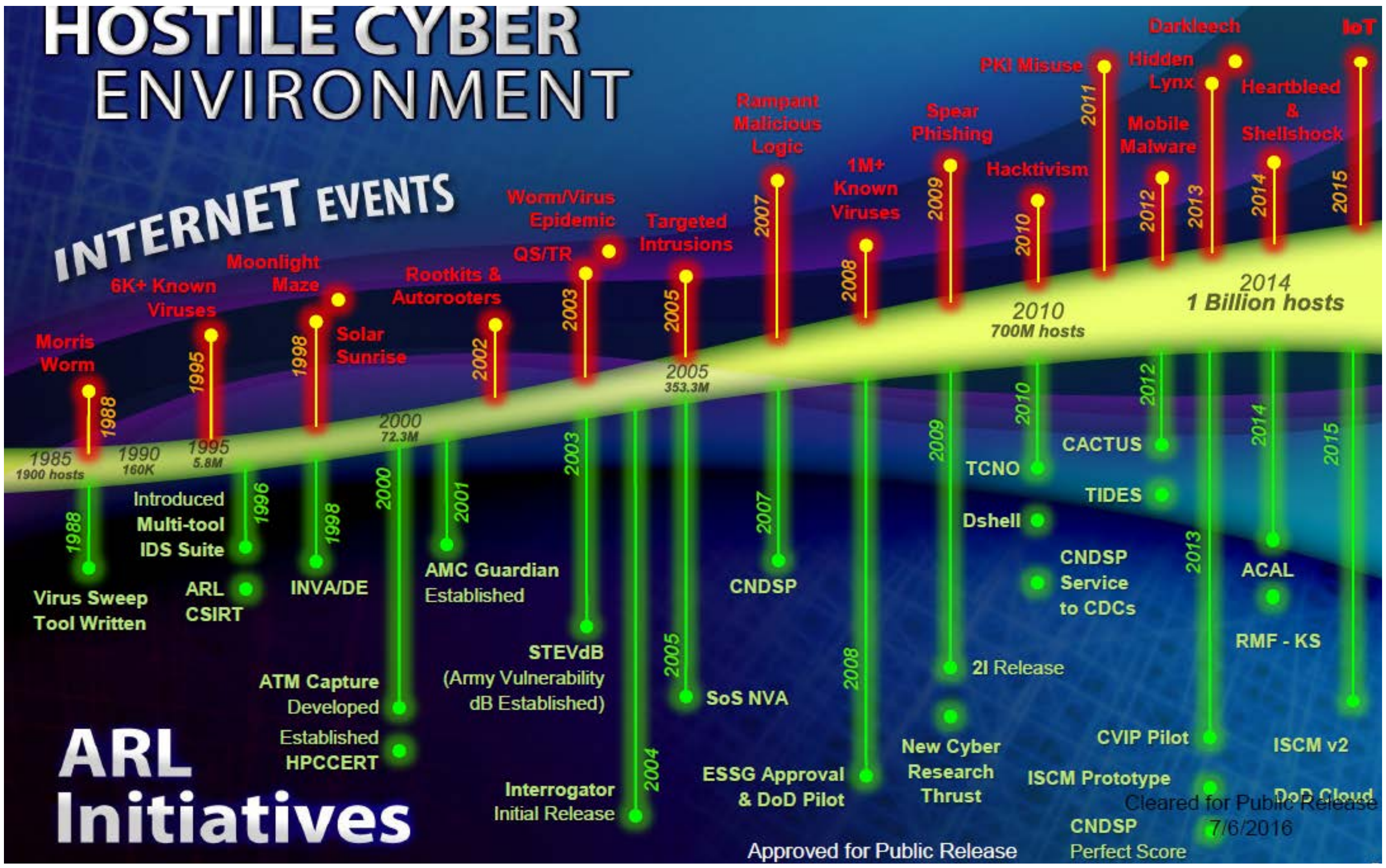


Fig. 1 ARL cyber research was always informed by real-world environment

collaborations by providing facilities and organizational support for enabling scientists and engineers from the U.S. and abroad to come to ARL for a period of time to work in partnership with ARL scientists.

Complementing its close ties with academic scientists, ARL research is also intertwined with practical, day-to-day operational responsibilities. Scientists are in direct communications with cyber analysts from the ARL Cybersecurity Service Provider (CSSP), a Tier II organization that defends networks of hundreds of customers belonging to all U.S. Military services, other government organizations, and even industrial entities (e.g., see the article "Information Security Continuous Monitoring (ISCM)" in this issue). ARL has a strong reputation in the area of threat analysis and forensics. The laboratory's experts in these fields are in high demand as they support cyber-related investigations conducted by law enforcement and counter-intelligence bodies. Vulnerability and survivability assessments of systems and networks that are either already deployed or are still in acquisition process, are another major area of ARL practical, hands-on contributions to Army cybersecurity. ARL's highly experienced teams of experts perform Cooperative Vulnerability and Penetration Assessments (blue team assessments) as well as Adversarial Assessments (red team assessments). Practical operational insights and needs obtained in operational activities are provided to scientists. They, in turn, utilize observations and data to develop new theories and models, and eventually to develop tools that transition into operational use.

Soldiers operate in a mobile environment, in complex terrain, with rapidly changing connectivity

Although participants of a broad cyber research community, ARL cyber scientists are largely driven by challenges unique to the ground operations of the Army. A key example is the exceptionally large attack surface of Army networks: the Army operates in environments within close proximity to allied and civilian assets and adversaries, comprising a complex cyber ecosystem. Forward-deployed network assets are vulnerable to cyber entry or physical capture and subversion of information and devices. Another distinct feature of Army cyber environments is the relatively disadvantaged assets, as the Soldiers' computing and communication devices are energy and weight constrained, with limited bandwidth and computational capacity. Large number of nodes and fast changes of Army cyber environments are also quite distinct. Soldiers operate in a mobile environment, in complex terrain, with rapidly changing connectivity. Lastly, these networks are often interspersed with civilian, allied, and adversarial networks.

These challenges inform and focus of ARL's research areas. One key area of research is the understanding the cyber threat. The topics in this area range from inferring influences and relations within a command and control organization from its encrypted communications, to novel uses of stylometry for identifying authors or origins of malware (Caliskan-Islam and Harang 2015), to tools and techniques for forensic analysis, and even to the study of cultural factors and personality that influence patterns of behaviors of cyber actors (Cho et al 2016).

Understanding the threats contributes to the characterization of risk experienced by a system or network. ARL's research in risk characterization includes such topics as statistical analysis of factors affecting the anticipated frequency of successful cyber attacks and theoretical approaches to network risk computation (see the article "Risk analysis with execution-based model generation" in this issue; also (Cam 2015). It also includes applied efforts to develop better procedures for risk inspection programs; tools for continuous monitoring of risk, cyber situational awareness (Kott et al 2014), and decision support systems for cyber risk assessments.

Knowing the risks helps focus the detection efforts (Kott and Arnold 2013). The comprehensive portfolio of ARL's research in detection of hostile cyber activities is based on close integration with practical network defense operations. It provides data and insights, and leads to the study of topics like the impact of packet loss in realistic cyber sensors on effectiveness of intrusion detection (Smith et al 2016). Other topics include special issues of detection in cyber physical systems; use of machine learning for detection methods suitable for mobile, resource-constrained devices (Harang et al 2015); cognitive models of human analyst's process of detection (Acosta et al 2016); and synergistic approaches to human-machine intrusion detection (see the article "Synergistic Architecture for Human-Machine Intrusion Detection" in this issue).

Ultimately, whether detected or not, the hostile cyber activities must be defeated. ARL explores approaches such as active cyber defense (Marvel et al 2014), post-intrusion triage for optimized recovery (Mell and Harang 2014), and cyber maneuvers that limit lateral propagation of hostile malware (Ben-Asher et al 2016).

These research projects are supported by a network of experimental facilities and laboratories dedicated to cyber research. For example, the ARL Cybersecurity Service Provider performs double duty: it supports large-scale operational cyber defense, but also acts as a laboratory for collection of real-world data for research, and a platform for insertion and testing of novel cyber defense tools continually invented and developed by ARL scientists.

Another example of a laboratory is the virtual laboratory called CyberVAN. It is an environment for design and execution of cyber experiments using virtual machines, real Army applications, and a network simulator capable of realistic portrayal of sizeable Army units in mobile operations in complex terrain. CyberVAN is particularly well suited for experimental validation of theoretical results by academic researchers, including international collaborators.

Additionally, the Army Cyber-research and Analytics Laboratory at ARL serves as an environment that supports various industrial and federally-funded partners of ARL. Its functions range: personnel training, product integration, systems engineering, and integrated testing using real-world data. A unique CHIMERA laboratory specializes in the study of human factors and human-information interactions in cyber defense; it helps to explore the human dimension of cybersecurity.

All this research yields results, many of which transitioned to practice as tools and systems. For example, Interrogator is an ARL-developed suite of network monitoring, intrusion detection and intrusion analysis tools. Used at ARL, as well as at a number of other organizations, its architecture is optimized for government cyber security operations, for defense against sophisticated threats, and for rapid insertion of research tools as plug-ins. Another example, Interrogator-in-a-Box, was developed for defense of mobile tactical networks. In addition, DShell is a framework for forensic analysis, popular with users at government agencies. ARL researchers attracted multiple, valuable international collaborators – and a good number of comments on social media – when they developed an open-source version of DShell and placed it on GitHub (see GitHub.com/USArmyResearchLab). Other examples of practical tools developed at ARL include COBWebS, a simulation tool that incorporate cyber warfare elements into training exercises, and a decision support tool for cybersecurity assessments, which helps perform assessments using public knowledge sources and custom data.

Future cyber capabilities will have to support continuous (real-time_ planning and execution

Looking further out, our long-term campaign of cyber research is guided by the vision of the future Army battlefield. In the year 2040, it will be a highly converged virtual-physical space, where cyber operations will be an integral part of the fight (Kott et al 2015). Cyber fires are the activities that will degrade, disrupt, deny, deceive and destroy not only informational, computational and communication resources of the adversary, but also the physical capabilities of its platforms, weapons, robots, munitions, and even of personnel. Cyber maneuver refers to activities that will rapidly move and transform the friendly informational-computational resources to deny the adversary an opportunity to attack, while imposing on him a new unsolvable problem (Fig. 2). Cyber fires and maneuver will rely on effective cyber intelligence collection capabilities.

Operating on multiple time scales, often far faster than human cognitive processes, in a highly dynamic, non-contiguous battlefield, these fires and maneuvers will join the conventional, kinetic fires and movements. Future cyber capabilities will have to support continuous (real-time, not just deliberate) planning and execution of highly agile, daring, aggressive cyber fires and maneuvers This will be performed in a way that is necessarily highly automated and reliant on machine intelligence, and yet responsive to human intent and guidance.

For these reasons, our cyber research efforts will increasingly focus on developing the models, methods, and understanding to overcome existing barriers to the realization of effective cyber fires and maneuvers in a tactical environment. The goals of this work are to

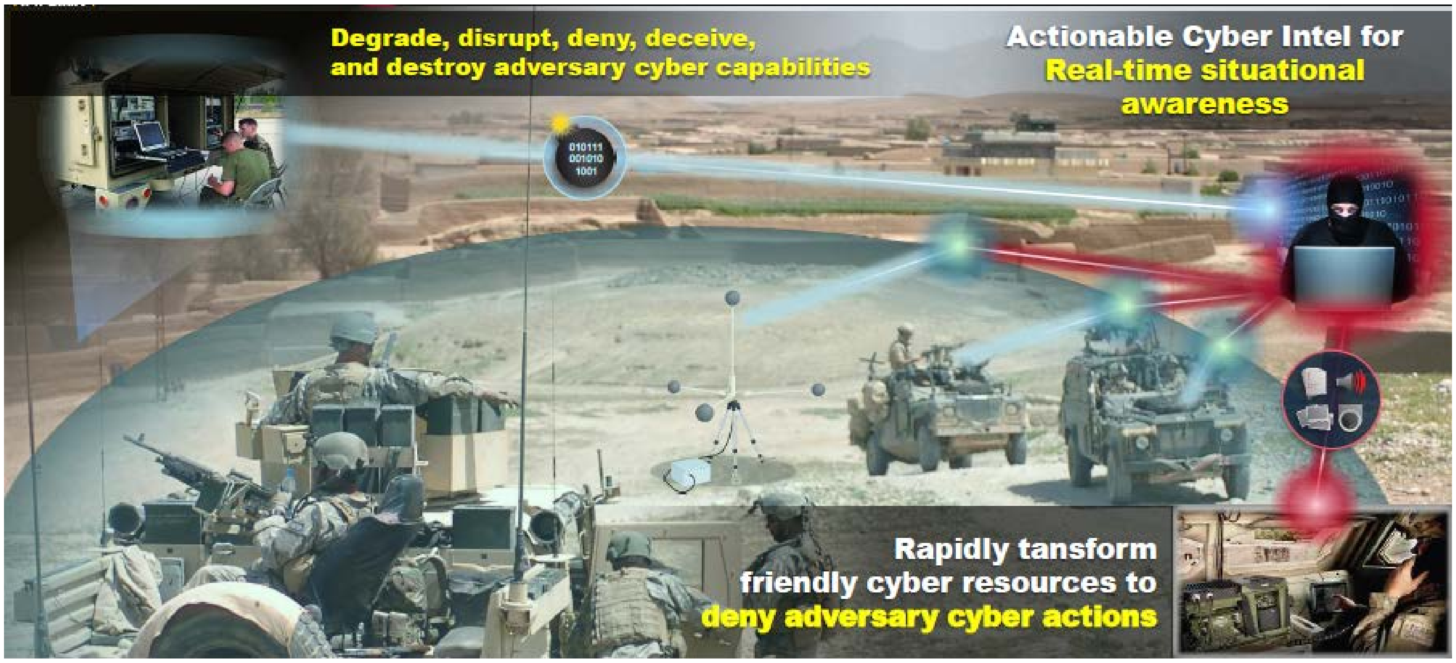

Degrade, disrupt, deny, deceive, and destroy adversary cyber capabilities
Actionable Cyber Intel for Real-time situational awareness
010111 001010 1001
Rapidly tansform friendly cyber resources to deny adversary cyber actions

Fig. 2 ARL cyber research is increasingly focused on cyber fires and maneuvers in tactical environments

pursue near-autonomous detection and identification of malicious activity directed at friendly networks; methods to rapidly respond to adversarial activities; predictive characterization of network vulnerabilities; and a robust framework to assess networks. Moreover, our research program will focus on the realization of methodologies for the reliable reconfiguration of friendly cyber assets to evade or recover from attack; covert means for collection and predictive analysis of enemy actions; and methodologies to degrade or destroy adversarial cyber assets with high certainty and predictable probabilities of kill. The articles assembled in this special issue reflect some of the steps ARL is taking towards this ambitious vision.

ACKNOWLEDGEMENTS: *Iris Saunders helped prepare the manuscript, Jerry Clarke built the presentation that served as the outline of this introduction, and Latasha Solomon orchestrated the development of all articles for this issue.*

ABOUT THE AUTHOR

Alexander Kott serves as the chief of the Network Science Division of the Army Research Laboratory headquartered in Adelphi, MD. In this position, he is responsible for fundamental research and applied development in performance and security of both tactical mobile and strategic networks. He oversees projects in network performance and security, intrusion detection, and network emulation. Between 2003 and 2008, Dr. Kott served as a Defense Advanced Research Projects Agency (DARPA) program manager responsible for a number of large-scale advanced technology research programs. His earlier positions included technical director of BBN Technologies, Cambridge, MA; director of R&D at Logica Carnegie Group, Pittsburgh, PA; and IT research department manager at AlliedSignal, Inc., Morristown, NJ. Dr. Kott received the Secretary of Defense Exceptional Public Service Award and accompanying Exceptional Public Service Medal, in October 2008. He earned his Ph.D. from the University of Pittsburgh, Pittsburgh, PA, in 1989, published over 70 technical papers, and co-authored and edited six technical books.

REFERENCES

[1] Acosta J, Edwards J, Shearer G, Parker T, Braun T, Marvel L. Modeling the decision processes of cybersecurity analysts to improve security assessments and defense strategies. Paper presented at: 23rd Annual National Fire Control Symposium (NFCS); 2016 Feb 8–11; Lake Buena Vista, FL.

[2] Ben-Asher N, Morris-King J, Thompson B, Glodek W. Attacker Skill, Defender Strategies, and the Effectiveness of Migration-Based Moving Target Defense in Cyber Systems. Paper presented at: 11th International Conference on Cyber Warfare and Security; 2016; Boston, MA.

[3] Caliskan-Islam A, Harang R, et al. De-anonymizing Programmers via Code Stylometry. SEC'15 Proceedings of the 24th USENIX Security Symposium; 2015; Washington, DC. Berkeley, CA: USENIX Association; c2015. p. 255-270.

[4] Cam H. Risk Assessment by Dynamic Representation of Vulnerability, Exploitation, and Impact. In: Ternovakiy, IV, Chin P. Proc. SPIE 9458 Cyber Sensing; 2015 April 20-24; Baltimore, MD. SPIE Proceedings Vol. 9548; c2015.

[5] Cho J, Cam H, Oltramari A. Effect of personality traits on trust and risk to phishing vulnerability: Modeling and analysis. 2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA). IEEE, 2016.

[6] Harang R, Marvel L, Parker T, Glodek W. Bandwidth Conserving DCO Signature Deployment with Signature Set Privacy. IEEE MILCOM 2015; Tampa, FL; October 2015.

[7] Hess M. The Worm that Changed the Internet. Everything CBTN. From https://blog.cbtnuggets.com/2016/02/the-worm-that-changed-the-internet/. [accessed 2016 Feb 3].

[8] Kott A, Alberts D A, Wang C. Will Cybersecurity Dictate the Outcome of Future Wars?. Computer 48.12 (2015):98-101.

[9] Kott A, Arnold C. The promises and challenges of continuous monitoring and risk scoring. IEEE Security & Privacy 11.1 (2013):90-93.

[10] Kott A, Wang C, Erbacher RF eds. 2014. Cyber Defense and Situational Awareness. New York: Springer.

[11] Marvel L, Harang RE, Glodek WJ , Parker TW, Ritchey RP. A Proposed Model for Active Computer Network Defense. IEEE MILCOM 2014; Baltimore, MD; 2014 October.

[12] Smith SC, Hammell RJ, Parker TW, and Marvel LM. A theoretical exploration of the impact of packet loss on network intrusion detection. International Journal of Networked and Distributed Computing, 4(1): 2016 Jan 1.

[13] Stoll C. The Cuckoo's Egg. New York, NY Simon & Schuster, 1989.

[14] Mell, P., & Harang, R. E. (2014, June). Using network tainting to bound the scope of network ingress attacks. In proceedings of the *Eighth International Conference on Software Security and Reliability* (pp. 206-215). IEEE.

THE CYBER SECURITY COLLABORATIVE RESEARCH ALLIANCE:

Unifying Detection, Agility, and Risk in Mission-Oriented Cyber Decision Making

BY: Patrick McDaniel
Institute for Networking and Security Research, Penn State University, University Park, PA

Ananthram Swami
Army Research Laboratory (ARL), Adelphi Maryland

ABSTRACT: *For military networks and systems, the cyber domain is ever-increasingly contested and congested space. Defenders of these systems must fight through adversary action in complex tactical and strategic environments. Just now completing its third year, the Cyber-Security Collaborative Research Alliance has sought to develop approaches for understanding and countering adversaries. The goal of this work is to develop a new science of cyber-decision making in military networks and systems. In this article we introduce the conceptual framework for this new science and consider its core research elements of detection (situational awareness), risk (measurement and assessment), and agility (adapting systems to evolving threats); overlaying this is the human dimension of users, defenders and attackers. We conclude by articulating a vision for future military cyber-operations.*

Cyber systems have changed the nature of warfare and the military. Real-time intelligence, autonomous and semi-autonomous systems, and improved command and control provide strategic advantages that save lives and make operations more effective, efficient and economical. Such systems are now essential to strategic networks supporting day-to-day military operations and tactical networks operating in hostile environments. The ever increasing reliance on cyber and cyber physical systems to conduct the Army's mission has in turn led to increasing number and sophistication of attacks on military cyber networks. Future Army networks will be a heterogeneous converged mix of wired networks, mobile cellular and mobile ad hoc networks. Nodes will consist of a variety of sensing, computing, actuating and communicating devices with diverse capabilities, and will be relatively disadvantaged. They may be embedded in backpacks, clothing, vehicles, weapon systems, munitions etc. Links on such networks will also be diverse, drawing upon multiple communication modalities. Soldiers and their assets will operate in a dynamic contested and congested environment, and must cope with advanced persistent threats. Army cyber security is further complicated, as it must often use and defend networks that it neither owns nor controls directly (e.g. mobile, fixed, and SCADA networks). The Army must often construct mission networks rapidly, with a variety of partners and allies. Thus Army networks face numerous challenges including a large attack surface, relatively disadvantaged assets, large scale, high dynamics, and advanced persistent threats (APT).

The military (and the entire computing world) have yet to develop the basic principles of how one identifies, understands, and counters adversaries in the digital domain. Indeed, providing such principles and the operational procedures to support them represents one of the grand challenges for military systems moving into the future [1].

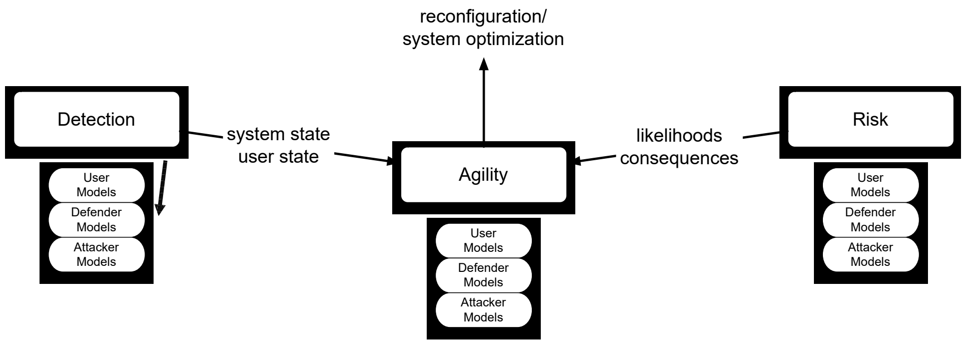
The Cyber Security Collaborative Research Alliance (CRA) is one aspect of the ARL Enterprise approach to Cyber Security for future Army networks [29][31]. The overall objective of the CRA is to develop a fundamental understanding of cyber phenomena, including aspects of human attackers, cyber defenders, and end users, so that fundamental laws, theories, and theoretically grounded and empirically validated models can be applied to a broad range of Army domains, applications, and environments. Entering its 4th year, the goal of the "Models for Enabling Continuous Reconfigurability of Secure Missions (MACRO)" Cyber CRA program [9][31] is to understand and model the risks, human behaviors, and maneuvers within Army cyber-operations. More practically, the goal of the Cyber CRA is to provide models for making decisions that "optimally" support the mission-oriented goals. Such models will enable defenders to detect and thwart attacks as well as allow operation progress in the face of ongoing and evolving threats, e.g., "fighting through" in contested and congested digital domains. From a pure research perspective, the overarching scientific goal of this effort is to develop a rigorous *science of cyber-decision making* that enables military networks to (a) detect the risks and attacks present in the environment, (b) understand and predict the motivations and actions of users, defenders, and attackers, (c) alter the environment to securely achieve maximal operation success rates at the lowest resource cost.

The Cyber CRA consortium is led by Penn State University, with Applied Communication Sciences Inc. (ACS), Carnegie Mellon University, Indiana University, the University of California Davis and the University of California Riverside as consortium members. The Alliance is a collaborative partnership between the consortium, the Army Research Laboratory (ARL) and the Communications-Electronics Research, Development and Engineering Center (CERDEC).

Develop a fundamental understanding of cyber phenomena, including aspects of human attackers, cyber defenders, and end users

The conceptual approach is to focus on cyber-defender decision-making. Winning or losing in the cyber-battlefield is dependent on defender action—but any defender action should be based on a careful analysis of the totality of the relevant environment, risks, and potential future states. Practically speaking, the goal is to enable defenders to answer the subtly complex question, "*Given a security and environmental state, what cyber-maneuvers best mitigate attacker actions and enable operation success?*" Note that this is not a discrete and momentary analysis, but one that must be continuous and adaptive within evolving situational awareness and mission goals. Like its physical counterpart in traditional kinetic warfare, the waging of cyber-warfare requires constant reevaluation of threats via reconnaissance, interpretation of adversarial intent and capability, and adjustments to strategy and resource use.

The Cyber CRA envisions future operational environments in which models of operations, users, defenders, and attackers guide the reconfiguration of highly diverse security and network infrastructure on a continuous basis. Operation survivability is achieved by altering the security configuration and network capabilities in response to detected adversarial operations and situational needs of users and resources and tools available to defenders. Cost and risk metrics are used to select optimal strategies and configurations that maximize operation success probabilities while mitigating adversarial actions. Models of user, defender, and attacker actions and needs are used to derive the operation state, as well as to identify those configurations that increase the probability of operation success. A simplified view of this conceptual framework is as follows:



In pursuing such a vision, it is important to remember that decisions cannot be made solely based on an understanding of cyber systems—one must factor in the needs and motivations of the people, i.e., users, defenders, and adversaries. Here, efforts in the Cyber CRA Cross-Cutting Research Area (CCRI) reasons about situational factors that may substantially alter user, defender, or attacker performance. For example, we have explored how to best present environmentally relevant information to defenders under times of stress to elicit the best outcome, i.e., what are the best details and formats to present a defender about potentially multiple simultaneous attacks. We are developing models of attackers to gauge intent and to identify countermeasures that will mitigate their impact on operation outcomes. By understanding how an attacker, user or defender is acting (or will act in response to a stimulus), we can predict the actions they will take. The vision is to estimate the type of attack, the goals of an attack, and the expected response taken by a user or defender, and thus estimate risk and predict future behavior. Ultimately, we will use these predictions to influence and control cyber-operation evolution and adversarial action.

This article describes the vision and current results of the Cyber CRA research focusing on elements of this vision including the operational model, risk, detection, and agility research areas. We begin in the next section by presenting a motivating cyber-mission used throughout the article.

Example Mission

Military networks present unique challenges. The diversity of strategic and tactical networks and the wired and wireless media over which operations are performed introduce diverse requirements on a science of security. The resources, configuration, and attack surface of tactical networks change from moment to moment, requiring agility to be responsive to shifting trends and goals. Such tactical networks may afford more restrictive security policy and configurations. Conversely, strategic "enterprise" military networks tend to be like traditional non-military wired networks in that they are static but heterogeneous in terms of size, topology, services and devices.

Consider an operation in a tactical network environment in which data are collected from vehicle-mounted cameras. This data is captured at regular intervals (every few seconds) as enabled by the driver and sent back to an intelligence-gathering facility several continents away (in the U.S.). The data is transmitted from the vehicle across a battlefield, to a regional operations center, and finally to the United States. The requirements of the operation are that it (a) reliably and (b) securely deliver the data from the battlefield to the intelligence facility in (c) a timely matter. In this setting, securely means that data confidentiality and integrity are retained (an adversary cannot obtain the image, nor can they alter it).

Operational Model

Operations with the Cyber CRA are modeled as state progression structures for reasoning about cyber-maneuvers and security goals and strategies [7]. Operations are broken into subtasks that progress temporally to a final operation success end-state. Each subtask is defined by a set of security requirements, security outcomes (the change in security state consequent of the subtask completion), risks, costs, and payouts. At the technical level, we formalize the operational model as discrete-time, finite-horizon Markov Decision Processes (MDP) [3][4]. This model enables us to obtain multiple *maneuver sequences*, to evaluate the cost associated with each sequence, and to make optimal choice of a maneuver sequence that accomplishes the operation under various attacks. In this, we adapt established control theory systems to Cyber-decision making. The operation model represents a formal specification of a cyber-scenario, e.g., the actions needed to complete an online task as described in the preceding section. We model an operation as a directed graph where the nodes are the states of the operation, and the edges are the state transitions needed to complete the mission. Each transition can represent atomic actions, abstractions for sub-operations or discrete time intervals [32]. However, based on the scenario, a choice of several maneuvers is possible. One cannot predict with certainty the consequence of these maneuvers given the current system state, but may model it as a stochastic event with many possible outcomes.

Consider a vastly simplified view of the example vehicle image transmission operation outlined in the preceding section. A model of that Cyber-mission (called an operation in this context) would proceed to (a) establish communication over local and global communication links, (b) transmit that data over the network and (c) terminate the communication. Note that this mission appears superficially to be a linear progression of states, but actually each step can be carried out in many different ways and may require mitigating attacks, using alternate methods when media is either not secure or too costly. Hence each of these steps can be represented by a complex flow of alternative approaches to implementing that higher level goal. The operational model is used to navigate these states and continuously select the approach with the highest probability of reaching a successful end state within a cost budget (e.g., the optimal control result). In this way, the Cyber system is adaptive to changing environmental states, resources, and resilient in the presence of adversary action.

Highlighted throughout, the operational model is built upon three interdependent inputs; detection state, risk metrics, and agility maneuvers. The detection state is the collection of all inputs from sensors and inferred states of the system (i.e., the situational awareness derived from the environment). The risk metrics is an assessment of the set of possible outcomes, their impacts on the operation and the environment, both as weighted by the probability of their occurrence. The set of maneuvers is the set of actions that will enable mission progress. The output is the best maneuver that has the best probability of moving the state towards the best mission goal (end-state).

Note that the end-state of the operation or operation strategy must change as the environment changes. For example, if it is found that an attacker has launched an attack that may prevent completion of the operation, other means to ensure success must be found. If success is no longer possible, other operation goals (with lower payout) and subtasks (with different security apparatus and configuration) may be defined or the operation aborted entirely. For example, in the event that an attacker successfully prevents the delivery of the high-resolution image (e.g. via DoS attack), the system may choose to send an image to another location (such as a local HQ) or may send a lower resolution image. In both cases a favorable outcome is achieved whereas the original goal could not be. This ability to find alternate strategies and outcomes is the key to fighting through adversarial action.

Cyber system is adaptive to changing environmental states, resources, and resilient in the presence of adversary action

Detection

The goal of the detection thrust is to develop theories and models that relate properties and capabilities of cyber threat detection and recognition processes/mechanisms to properties of a malicious activity, and of properties of Army networks. More concretely, the goal is to determine whether there is an ongoing cyber-threat that can negatively affect the operation and provide assessments on: (i) what is the most likely threat; (ii) what impact will it have on the operation (e.g., leakage of data, system breakdown, etc.) in terms of increase in cost or decrease in payout; and (iii) the confidence in the process (based on evidence collected). Detection is influenced by (i) the actions of the attacker, and (ii) the dynamics of the environment (which can itself influence the attacker to behave in certain ways). The CRA's efforts in detection to date have been focused on three areas addressing the needs of operations: (a) advancing traditional intrusion detection, (b) understanding defender's decision processes, and (c) developing a science of evidence collection.

Note that collection of data and transmission to a central fusion center can place demands on already constrained communications media in tactical networks. Even collecting and local processing may pose challenges in the often energy and computing constrained environment. Strategic networks may have greater resources, but support a larger diversity of operations. One of the key investigations within the CRA is the calibration of detection apparatus based on the resource cost for the target (tactical or strategic) as based on an understanding of operational requirements.

The operational model uses inputs from intrusion detection systems to infer the model state. However, current systems are limited in their accuracy and false-positive rates [25]. The team is looking at several alternate models and scientific challenges to traditional detection. One alternate model developed within the CRA is *diagnosis-enabling intrusion detection* (DEID) [20]. Departing substantially from traditional signature and anomaly-based detection, DEID infers high level attacks and effects using correlations, automated reasoning, and forensic techniques. In DEID: (*i*) A large volume of data that encompasses all levels of operation at each node (human actions, sensors, applications, OS, network behaviors) and across a multitude of monitors is collected. (*ii*) The observed, correlated evidence are examined and an attempt is made to map them onto expected correlated behaviors derived from the models of both the system and human actors; the mappings allow the determination of normal/attack behaviors with high accuracy (diagnosis). (*iii*) If the system is unable to map the observed correlated behaviors to known attacks (e.g., may be a zero-day attack) appropriate information is exported to the human defenders.

Another effort seeks to expand and formalize the science of detection by exploring the vulnerabilities and countermeasures inherent to the underlying machine learning algorithms upon which most detection systems are based. In particular, we are developing intrusion detection techniques that will be robust in the face of adversaries, work with limited information, and greatly reduce the attack surface that adversaries may leverage undetected. For example, we have developed novel algorithms and defenses for adversarial samples—adversarially crafted detection sensor inputs that use model error to bypass detection [39][30]. In defining the new science of this area, we introduced a taxonomy formalizing the space of adversaries targeting deep neural networks used for classification tasks [35]. We then investigated the case of source-target misclassification: forcing the targeted classifier to misclassify samples from any source class into any chosen adversarial target class. Our algorithms exploit a precise understanding of the sensitivity of the mapping between inputs and outputs using the forward derivative---the Jacobian of the model learned by the DNN classifier. Adversarial saliency maps build on the forward derivative to compute a score indicating the likelihood that each input component contributes to the adversarial goal of source-target misclassification. Perturbations are iteratively selected using adversarial saliency maps and added to the sample until it becomes adversarial---misclassified by the deep neural network. We extend that work to create defense against such attacks [36]. The intuition is to make the models learned by deep neural networks smoother to increase the average minimal perturbation magnitude an attacker needs to introduce to craft adversarial samples. This minimal perturbation characterizes a neighborhood around points in which the model's decision is constant, which in turn defines a robustness metric for detection models. We proposed the use of defensive distillation to increase model robustness. Thus, distilled models are harder to attack: adversarial samples need to be more perturbed in order for the model to misclassify them. We are applying similar approaches and validation techniques to other ML techniques and measuring the resilience of detection systems again these attacks.

A challenge for intrusion detection systems is the integration of hard-earned information relevant to an attack, but that is not measurable at run time. Recent advances by CRA PIs in Learning Using Privileged Information (LUPI) [40] provide some insights.

CRA is exploring the 'evidence' collection processes and systems

Beyond the detection algorithms, the CRA is exploring the "evidence" collection processes and systems—the quality of any detection system is critically defined by the completeness and accuracy of its sensor inputs. There are several challenges addressed by the CRA in configuring evidence collection in military systems. First, monitor placement is often ad hoc and accidental. Large, complex environments can contain thousands of devices and services with subtle interactions and behaviors. How one places sensors in these environments is key to getting an accurate vision of the environmental state. The team is studying measures of coverage and developing algorithms for sensor placement (minimal number, optimal locations), e.g., [26]. The team is exploring several strategies including max-coverage, min-resource, and game-theoretic strategies for placement algorithms. Such algorithms are being developed both as static and dynamic placements —the latter of which is a form of system agility discussed below.

Risk

The accepted definition of risk in Cyber-systems is the probability of some negative outcome times the "cost" of its impact. Decision making under risk takes into account these probabilities and impacts when forming the optimal maneuver, e.g., maximizing payout while minimizing impact costs. For example, if the use of one kind of transmission medium for the image transfer in our example mission would introduce a high risk of failure or compromise, then another must be selected. Identifying these risks and making decisions based upon them are key to achieving successful outcomes (and avoiding negative side effects). Within the CRA, we are developing theories and models that relate fundamental properties and features of dynamic risk assessment algorithms to the fundamental properties of dynamic cyber threats, Army's networks, and defensive mechanism. These risk models and metrics will then be integrated into risk calculations in the operational model. Here we combine traditional system and network risk metrics with human oriented risk metrics. In the latter, individuals (users, defenders, and attackers) and human-resource interfaces are directly integrated as a component of risk valuation. Attackers create risk; defenders mitigate risk; and users both create and mitigate risk. In the operation-based framework, each operation will include users, defenders, the user/defender interacting team, and attackers. Based on the probability of being attacked and that attack being detected, each combination of operation/user/defender/resources must select an appropriate mitigation path within the operation model. Thus, the risk related to an operation state transition is a vector of outcomes with consequences that may impact not only the task itself, but also the infrastructure, users, and other operation activities. This evaluation of risk requires us to model and verify not only individual risks, but also the interplay of risk at multiple layers and sources, and under different contexts.

CRA research has identified risk metrics for system level, human factors, and software vulnerabilities [42]. We have used human factors frameworks to identify defender trust metrics and attacker culture metrics [18][33][5] . Expertise surveys and extensive data collected during the National Guard CyberShield exercises (2015, 2016) are being used to develop defender models [19]. We have developed a Bayesian network analysis approach for risk quantification and decision-making, and demonstrated that it can capture the dynamic change in risk magnitude due to state change [17].

Having identified early candidate models, the CRA team is developing experiments for validating user metrics, systems and network metrics, risk quantification, effective representations of risk, and optimality of risk assessment vectors. For human related models, each model and sub-model are evaluated for predictability of the outcomes derived from test subjects in controlled and operational environments. These subjects will be tested as individuals and as teams (e.g., during Cyber-training events). The team is also experimenting with risk metrics in physical networked environments to measure their accuracy in multiple tactical and strategic networks and in the presence of attacks.

Agility

Agility refers to the context and operation-aware reconfiguration of the system or the operation autonomously or by the defender with respect to a potential attack or perceived risk. Such reconfigurations of environment or operation strategies are referred to as *cyber-maneuvers*. Maneuvers, often called moving target defenses, seek to continually alter the attack surface as perceived by an adversary. Within the CRA, the research effort focuses on developing models and algorithms that reason about the current state, the universe of potential security-compliant cyber maneuvers (i.e., "maneuver" in the space of hardware, software, network and system characteristics and topologies) and end-states, and how these maneuvers are affected by and impacts human users, defenders, and attackers. Building on recent advances in moving target defenses [21][22], we are exploring game-theoretic models that select maneuvers that mitigate adversarial actions on operation outcomes. Note that some maneuvers may be offensive (such as deception techniques) in that they launch counter-measures that impact would-be attackers.

Broadly speaking, in an agile operation environment the system state needs to be continuously analyzed based on detected threats, assessed risks and human feeds on operation evolution. Subsequently, the system must be reconfigured towards: (i) preventing and mitigating attacks, thereby maximizing outcome utility in our operation model; (ii) completing the operation in a secure and resource-optimal way given the current state and the dynamics of the end state; (iii) minimizing risk and accounting for deception; and (iv) integrating the human factors that impact the cyber-security operations. An adversary's perception of the attack surface can be altered by maneuvers in different layers, e.g., software, network, and system layers. Not surprisingly, one can also formulate agility problems in a game-theoretic setting.

Combine traditional system and network risk metrics with human oriented risk metrics

Our study of *software maneuvers* seeks to develop the science of software agility. The objective of software agility is to pick the optimal tasks to execute, and the optimal software configuration in which these will execute, given desired security outcomes, risks, and current state of the system (e.g., attacks, defenses, including psychosocial factors). We achieve this objective by (1) using a proactive approach to software agility to withstand and thwart attacks, and (2) continuously analyzing the systems software state and if/when needed, performing software reconfiguration, based on detected threats, assessed risks and human feeds on operation evolution. Our early efforts were focused on reactive approaches for reconfiguring a key-value server and mobile apps, and moved on to study of proactive reconfiguration, cost/payout metrics, and approaches beyond smartphones and key-value servers. The cost/benefit analysis balances security, capability, availability and resource consumption. The Agility team has made advances in several directions, such as the quantification of the cost of reconfigurations [28], theory and practice of cyber-maneuvering [38], and characterizing root-provider attacks [41]. Over the next two years we will generalize to more powerful models of maneuver in a formal quasimetric space, reconfiguration, and cost; and formal guarantees of attack resistance. Agility mechanisms are one form of deception, and a formal approach to this, including psychosocial metrics of deception, warrant study. We proposed software "wrappers" as a flexible mechanism for dynamically changing programs and runtime environments and have used it for changing data structures on-the-fly in server-side processing [27], changing the OS state [38], and bytecode rewriting to survive faults [1]. Recent work in the CRA is developing a unified approach to encoding configurations, and formal mechanisms for controlling transitions. A related validation study is analyzing existing and new side channels (e.g., TCP stacks, [6]) to understand the limitations of software randomization strategies.

A key issue in *game-theoretic approaches* is to determine the appropriate models of interactions between the defender and attacker. While it is conceivable that the two may choose their strategies simultaneously, it is more likely that each of them will choose their strategies in response to the "observable actions" by the other. The tradeoff between leading/following depends on the specific payoff functions as well as the penalty of delaying a player's action (e.g., missing an attack opportunity). In this, we are exploring various dynamic game formulations, with different leader/follower roles for the attacker and defender. For example, the defender may lead the game by invoking his/her proactive security measures. The attacker will then respond with his/her own actions. The roles can be dynamically switched, depending on each player's payoff (e.g., shortly after taking an action, the defender may decide to take a subsequent action without waiting for the attacker action; this decision may be triggered by more updated statistical analysis of adversarial responses). Such dynamism enable us to capture the bounded regimes of rationality of human adversaries. Other recent work on game theoretic approaches include models for stealthy attacks, involving two-player differential games, and asymmetric versions of the FlipIT game where the feedback may be delayed. We have characterized best response strategies [13][14]. Our three-player game models build on this, including now a third player – the insider – who may be helpful or harmful. We have characterized Nash equilibria in this three-player sequential game.

Recent research on the psychology of decision making seeks to understand how humans make decisions from experience (DFE) rather than descriptions. Such an approach enables one to relax assumptions of rationality. PIs in the team have championed the development and use of Instance-Based Learning (IBL) models that do not need predefined implementations of interaction strategies [16], [15]. IBL can be integrated with automated tools and models of risk assessment in cyber security, e.g., [8], as recently demonstrated in [2]. Our current work addresses key challenges related to scalability with multiple players, and cognitive biases and judgment impairments (such as due to memory and recall limitations), and how attack and defense strategies evolve in repeated games, across multiple attack patterns. Central to most game theoretic assumptions are assumptions of information certainty and human rationality (which includes, for example, ability to perfectly recall all relevant information). Assumptions of rational behavior on part of the attacker may lead to poorly performing strategies against a myopic attacker; and assumptions of rational defenders may lead to defense mechanisms that are never realized in practice. We are augmenting our game theoretic approaches with IBL to model humans with bounded rationality. Psychological research suggests that risk variability in humans may be explained and predicted by cultural and other cognitive factors; and such factors have been observed in the cyber domain to gain insights into attackers [23][37]. Our current focus is on incorporating such personality and cultural factors, for individuals and groups, into behavioral models such as IBL and game theoretical approaches to account for individual variability and biases [13]. We are further incorporating tools to enable cutting edge analysis of individual decision-making [24] and exploring how system prompts and presentation effect security outcomes [40].

Discussion & Conclusions

We have introduced a conceptual framework and research agenda for reasoning about cyber-maneuvers in military environments. This model jointly reasons about situational awareness, risk assessment, and software, system and network agility to support ongoing cyber-operations. These factors are integrated into a unified operational model that defenders and automated systems can use to make "optimal" decisions about how to achieve mission goals and mitigate the activities of adversaries.

The inter-dependencies between the elements of risk, detection and agility are obvious. Resources spent on detection (how many monitors, how many samples, choices of algorithms) are dictated by assessment of threat and risk, and in turn feed into risk assessments. The outputs of detection (including our confidence in such detection outputs) provide inputs for agility maneuvers; in turn, agility decisions feed information about network configurations to detection strategies. Agility algorithms depend on the detection of potential attacks, the risks associated with the perceived attacks, the desired responses by the defenders and attackers, the perceived risk in transitioning from the current to a desired state; and accounting for human dynamics. Risk feeds both detection and agility; for both, it shapes the goals and focus of the algorithms. Thus, as is evident from our operation model, the goals of the models and algorithms for agility are integrally dependent on the risk, detection, and human dynamics.

Experimental verification and validation have been and continue to be key components of CRA research. While all algorithms are typically tested on synthetic / simulated date, we make extensive use of the cyber experimentation testbed called Cyber Virtual Ad hoc Network (CyberVAN) [10].

The research towards reaching this vision is just beginning its fourth year, but we have already made great strides in analyzing target environments and developing preliminary models. Our current focus is to bring together these disparate but complementary models into a comprehensive framework, and to measure its effectiveness in realistic military contexts. These experiments will assess the accuracy and sensitivities of decision making process and provide guidance into its refinement. ✪

REFERENCES

[1] K.B. Alexander. Warfighting in cyberspace. Joint Force Quarterly, Issue 46, July 2007. .

[2] T. Azim, I. Neamtiu, and L. Marvel. Towards self-healing smartphone software via automated patching. Proc. 29th IEEE/ACM International Conference on Automated Software Engineering (New ideas track), ASE 2014, September 2014

[3] N. Ben-Asher, A. Oltramari, R. Erbacher, C. Gonzalez. Ontology-based Adaptive Systems of Cyber Defense. *The 10th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS) 201*

[4] D.P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, 2005.

[5] D.P. Bertsekas and S.E Shreve. *Stochastic optimal control: The discrete time case*. Academic Press, 2007.

[6] M. Cains, D. Henshel, B. Hoffman, C. Sample. Integrating Cultural Factors into Human Factors Framework for Cyber Attackers. Proc. 7th Intl. Conf. Applied Human Factors and Ergonomics (AHFE), 2016

[7] Y. Cao, Z. Qian, Z. Wang, T. Dao, S.V. Krishnamurthy, L. M. Marvel. Off-Path TCP Exploits: Global Rate Limit Considered Dangerous (CVE-2016-5696). *Proc. USENIX SECURITY 2016*, Austin, TX, 2016

[8] Z. B. Celik, N. Hu, Y. Li, N. Papernot, P. McDaniel, J. Rowe, R. Walls, K. Levitt, N. Bartolini, T.F. La Porta, and R. Chadha. Mapping Sample Scenarios to Operational Models. *Proceedings of the IEEE Military Communications Conference (MILCOM)*, Nov 2016, Baltimore, MD.

[9] J.-H. Cho, H. Cam, A. Oltramari. Effect of personality traits on trust and risk to phishing vulnerability: Modeling and analysis. Proc. *IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA'2016)*, March 21-25, 2016, San Diego

[10] Cyber-Security Collaborative Research Alliance, Webpage, August 2016, http://cra.psu.edu/

[11] Cyber virtual ad hoc network (CyberVan). http://www.appcomsci.com/research/tools/cybervan [Online; accessed 5-September-2016].

[12] G. Deckard, L.J. Camp. Measuring efficacy of a classroom training week for a military cybersecurity training exercise. Proc. IEEE International Conference on Technologies for Homeland Security, (Waltham, MA) 10-16 May 2016.

[13] K. Durkota, V. Lisy, B. Bosansky, and C. Kiekintveld. Optimal network security hardening using attack graph games. Proc. IJCAI, 2015.

[14] X, Feng, Z. Zheng, D. Cansever, A. Swami, and P. Mohapatra. Stealthy Attacks with Insider Information: A Game Theoretic Model with Asymmetric Feedback. Proc. *IEEE MILCOM 2016*, Baltimore, MD, Nov 2016.

[15] X, Feng, Z. Zheng, P. Hu, D. Cansever, and P. Mohapatra. Stealthy Attacks Meets Insider Threats: A Three-Player Game Model. Proc. *IEEE MILCOM 2015*, Tampa, FL, Oct 2015.

[16] C. Gonzalez, N. Ben-Asher, J. Martin, V. Dutt. A cognitive model of dynamic cooperation with varied interdependency information. *Cognitive Science*, 39, 457-495, 2015.

[17] C. Gonzalez, N. Ben-Asher, A. Oltramari, C. Lebiere. Cognition and technology. In *Cyber Defense and Situational Awareness*, pp. 93-117. Springer International Publishing, 2014

[18] D. Henshel, A. Alexeev, M. Cains, B. Hoffman, I. Neamtiu, J. Rowe. Modeling cybersecurity risks: Proof of concept of a holistic approach for integrated risk quantification. Proc. IEEE Intl. Symp. Technologies for Homeland Security (HST), 2016

[19] D. Henshel, M. Cains, B. Hoffman, T. Kelley. Trust as a human factor in cyber security risk assessment. Proc. 6th Intl. Conf. Applied Human Factors and Ergonomics (AHFE), July 2015

[20] D. Henshel, G. Deckard, B. Lufkin, N. Buchler, B. Hoffman, L. Marvel, S. Cannello, and P. Rajivan. Predicting Proficiency in Cyber Defense Team Exercises. Submitted to Military Communications Conference, MILCOM 2016-2017 IEEE, IEEE 2016

[21] C. Jackson, R. Erbacher, S. Krishnamurthy, K. Levitt, L. Marvel, J. Rowe, A. Swami. A Diagnosis-based Approach to Intrusion Detection. *20th European Symposium on Research in Computer Security (ESORICS 2015)*, Vienna, Austria.

[22] S. Jajodia, A.K. Ghosh, V. Swarup, C. Wang, and X.S Wang, Eds. *Moving Target Defense: creating asymmetric uncertainty for cyber threats*, volume 54. Springer Science & Business Media, 2011

[23] S. Jajodia, A.K. Ghosh, V.S Subrahmanian, V. Swarup, C. Wang, and X.S. Wang, Eds. Moving Target Defense II: *Application of Game Theory and Adversarial Modeling*. Springer Science & Business Media, 2013.

[24] D.N. Jones and D.L. Paulhus. Introducing the short dark triad (sd3): A brief measure of dark personality traits. Assessment, 21(1):28{41, 2014.

[25] T. Kelley, B. Bertenthal. Attention and past behavior, not security knowledge, modulate users' decisions to login to insecure websites. *Information and Computer Security*, 24(2), 2016

[26] R.A. Kemmerer, and G. Vigna. Intrusion Detection: A Brief History and Overview. I*EEE Security & Privacy*, 2002.

[27] K. Khalil, Z. Qian, P. Yu, S. Krishnamurthy, A. Swami, Optimal Monitor Placement for Detection of Persistent Threats. IEEE Globecom, Washington DC. 4-8 Dec 2016.

[28] A. Kusum, I. Neamtiu, and R. Gupta. Adapting graph application performance via alternate data structure representation. *Proc. 5th International Workshop on Adaptive Self-tuning Computing Systems*, 2015.

[29] L. Marvel, S. Brown, I. Neamtiu, R. Harang, D. Harman, and B. Henz. A framework to evaluate cyber agility. Proc. *IEEE MILCOM 2015*, Tampa, FL, Oct 2015.

[30] P. McDaniel, T. Jaeger, T.F. La Porta, N. Papernot, R. Walls, A. Kott, I. Neamtiu, L. Marvel, A. Swami, P. Mohapatra, S. Krishnamurthy. Security and Science of Agility. *Proceedings of the First ACM Workshop on Moving Target Defense*, 2014

[31] P. McDaniel, N. Papernot, and Z.B. Celik. Machine learning in adversarial settings. *IEEE Security & Privacy*, 2016.

[32] P. McDaniel, B. Rivera, and A. Swami, Toward a Science of Secure Environments. *IEEE Security & Privacy Magazine*, 12(5), July-August, 2014

[33] A. Oltramari, L.F. Cranor, R. Walls, and P. McDaniel. Computational Ontology of Network Operations. Proceedings of the *IEEE Military Communications Conference (MILCOM)*, October 2015. Tampa, FL.

[34] A. Oltramari, D. Henshel, M. Cains, B. Hoffman. Towards a Human Factors Ontology for Cyber Security. *Proc. Semantic Technology for Intelligence, Defense, and Security (STIDS)*, 2015.

[35] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z.B. Celik, and A. Swami. Practical black-box attacks against deep learning systems using adversarial examples. arXiv preprint arXiv:1602.02697, 2016.

[36] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. *IEEE European Security and Privacy Symposium*, Mar 2016.

[37] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. *IEEE Security and Privacy Symposium*, May 2016.

[38] C. Sample. Cyber + Culture Early Warning Study. CMU/SEI-2015–SR-025, 2015, Online at: http://resources.sei.cmu.edu/asset_files/SpecialReport/2015_003_001_449739.pdf

[39] Z. Shan, I. Neamtiu, Z. Qian, and D. Torrieri. Proactive restart as cyber maneuver for android. Proc. *IEEE MILCOM 2015*, Tampa, FL, Oct 2015.

[40] R. Shay, L. Bauer, N. Christin, L.F. Cranor, A. Forget, S. Komanduri, M.L. Mazurek, W. Melicher, S.M. Segreti, and B. Ur. A Spoonful of Sugar?: The Impact of Guidance and Feedback on Password-Creation Behavior. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 2903-2912. ACM, 2015.

[41] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *Proceedings of the 2014 International Conference on Learning Representations*. Computational and Biological Learning Society, 2014.

[42] V. Vapnik and R. Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, pp. 2023-2049, 2015

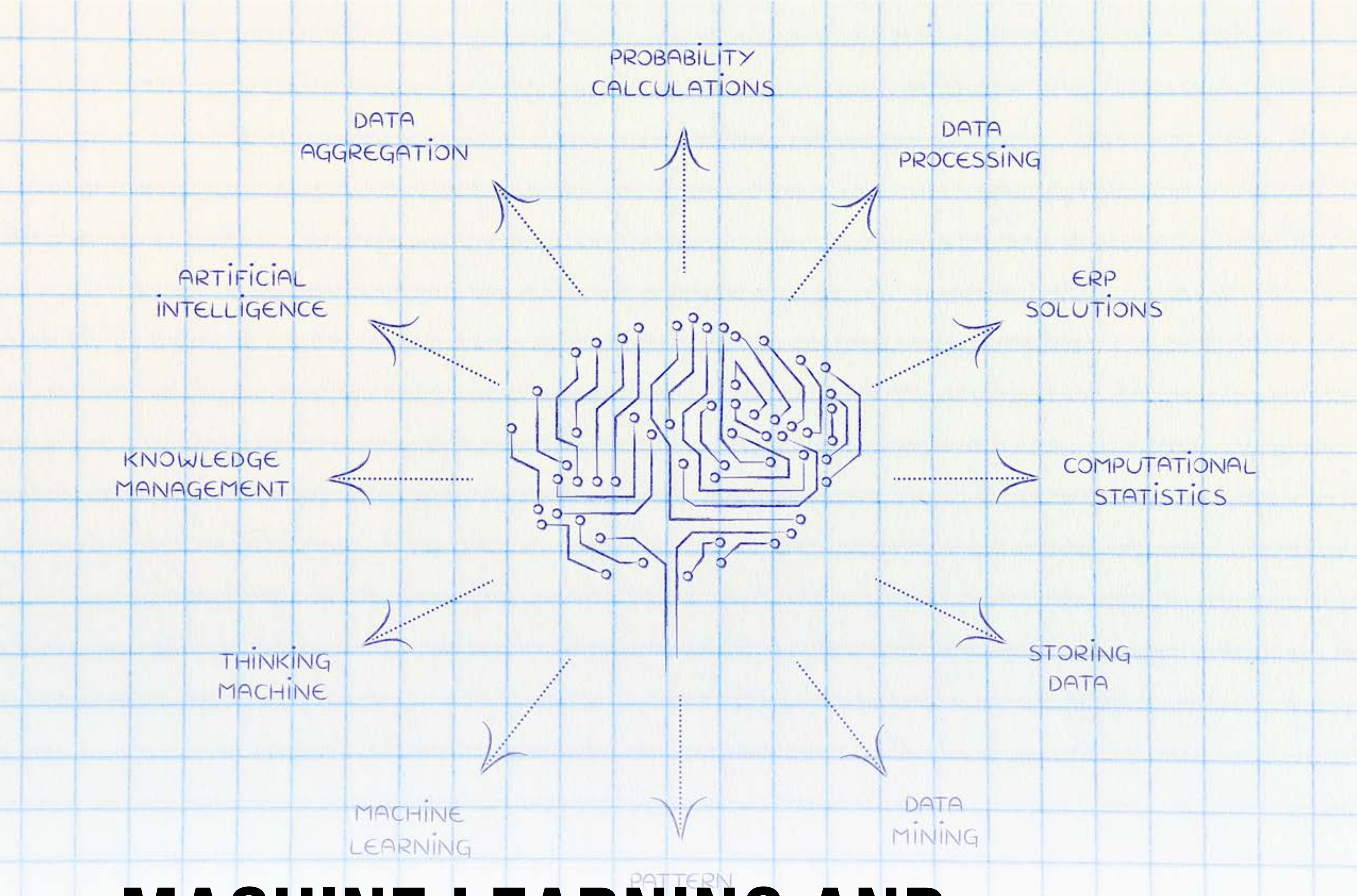
[43] H. Zhang, D. She, and Z. Qian. Android root and its providers: A double-edged sword. *Proc. 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pp 1093–1104, 2015

[44] B. Zhou, I. Neamtiu, R. Gupta. How Do Bug Characteristics Differ Across Severity Classes: A Multi-platform Study. Proc.26th IEEE International Symposium on Software Reliability Engineering, November 2015

ABOUT THE AUTHORS

Patrick McDaniel is a Distinguished Professor in the School of Electrical Engineering and Computer Science at Pennsylvania State University, Fellow of the IEEE and ACM, and Director of the Institute for Networking and Security Research. Professor McDaniel is also the program manager and lead scientist for the Army Research Laboratory's Cyber-Security Collaborative Research Alliance. Patrick's research focuses on a wide range of topics in computer and network security and technical public policy. Prior to joining Penn State in 2004, he was a senior research staff member at AT&T Labs-Research.

Ananthram Swami is with the US Army Research Laboratory and is the Army's Senior Research Scientist (ST) for Network Science. Prior to joining ARL, he held positions with Unocal Corporation, USC, CS-3 and Malgudi Systems. He was a Statistical Consultant to the California Lottery, developed a MATLAB-based toolbox for non-Gaussian signal processing. He has held visiting faculty positions at INP, Toulouse, and currently at Imperial College, London. Swami's work is in the broad area of network science, including network security. He is an ARL Fellow and a Fellow of the IEEE.



MACHINE LEARNING AND NETWORK INTRUSION DETECTION:

Results from Grammatical Inference

BY: Dr. Richard Harang, Army Research Laboratory (ARL)

INTRODUCTION: *Machine learning for network intrusion detection is an area of ongoing and active research (see references in [1] for a representative selection), however nearly all results in this area are empirical in nature, and despite the significant amount of work that has been performed in this area, very few such systems have received nearly the widespread support or adoption that manually configured systems such as Bro [2] or Snort [3] have. As discussed in [1], there are several differences between more conventional applications of machine learning and machine learning for network intrusion detection that make intrusion detection a challenging domain; these include the overwhelming class imbalance (see [4] for a detailed discussion of this issue), the high asymmetry in misclassification costs, the difficulty in evaluating the performance of an intrusion detection system, and the constantly changing nature of network attacks.*

However; these arguments, which largely stem from empirical observations about the distribution and impact of attacks, do not address a more fundamental question: whether or not it is possible from a theoretical perspective to use machine learning to construct a general and effective intrusion detection system. This is the sort of fundamental question that we seek to answer with a "fundamental science of cybersecurity" [5], which has so far proved to be somewhat elusive. In this article, we collect results from the field of grammatical inference to show that the use of machine learning for intrusion detection via deep packet inspection is inherently limited, and cannot be expected to generalize effectively even if the proposed algorithm can be shown to perform well on some particular task.

Network inputs, file formats, and other automatically generated input structures (henceforth: 'messages') are by construction formal grammars, in which the message is both produced and interpreted according to a fixed set of rules. If we wish to identify a particular class of message as either "safe" or "malicious", we are then implicitly attempting to learn a recognizer for the (perhaps merely implicit) formal grammar underlying that class of message format or protocol. This places the problem firmly within the realm of grammatical inference, a field with a deep theoretical literature. And indeed, we can find strong parallels, if not outright reductions, from many commonly encountered problems in machine learning for intrusion detection and grammatical inference.

Explore some common approaches to network intrusion detection through the lens of computational"

In the following, we present brief background on grammatical inference, explore some common approaches to network intrusion detection through the lens of computational and show how these results illuminate several common approaches to machine learning for network intrusion detection. In particular, these results suggest that general purpose machine learning approaches to network intrusion detection are – from first principles – not computationally tractable, and in fact may be attempting to solve problems that range from cryptographically hard to NP-complete.

Key Concepts from Formal Language Theory and Grammatical Inference

The key concepts from formal language theory that we will require are "languages", "grammars", and "recognizers".

Grammars are defined as tuples: $G=(N,\Sigma,R,i)$ where N is a set of nonterminal symbols, Σ is a set of all possible symbols (terminal and nonterminal), R is a set of production rules indicating which symbols in the string can be replaced with other symbols (the 'left' and 'right' side of the rule, respectively), and i is an initial symbol. A "production" of a grammar is a sequence of rule applications to a sequence until it consists of only terminal symbols; the (typically infinite) set of all productions of a grammar is the language L produced by the grammar, often denoted $L(G)$. A recognizer for a grammar is some algorithm which takes some input sequence as input, and either returns "accept" if $s \in L(G)$, or "do not accept" if $S \notin L(G)$.

The various restrictions that are placed on the production rules place the grammar into one class or another. The most commonly encountered classification of grammars – the Chomsky hierarchy [6] – classifies grammars in increasing complexity as regular, context-free, context-sensitive, and unrestricted; each class being contained in the following one, having successively fewer restrictions on the production rules, and requiring a successively more powerful model of computation in order to recognize. Regular grammars, for instance, may only have rules in which the left side contains a single nonterminal symbol, and the right side contains an empty string, a terminal symbol, or a terminal symbol followed by a nonterminal symbol. The Chomsky hierarchy is of particular interest because each class within the Chomsky hierarchy corresponds precisely to a particular model of computation which is required to build a recognizer for a language in that class. Regular grammars, for instance, can be recognized by finite automata, while context-free grammars require the addition of a stack in order to be able to construct a recognizer for them. Context-free grammars are of particular importance as the Backus-Naur notation, which is frequently used to describe network protocols and occasionally file formats are itself a notation for context-free grammars.

Grammars and recognizers then fix a set of production/recognition rules, and produce decisions about strings. Grammatical inference (see [7] for a broad review) tackles the inverse problem: given some set of data relating to an unknown grammar, can we produce decisions about properties of that grammar?

Example data available for training may include only strings $\{s^{+}: s \in \Sigma^{*} \cap L\}$ in the language generated by a grammar – a "positive presentation" – or a set of strings $s=\{s^{+}: s \in \Sigma^{*} \cap L\} \cup \{s^{-}: s \in \Sigma^{*} \setminus L\}$ combined with labels $n:\{0,1\}$ which indicate if $s \in s^{+}$ or $s \in s^{-}$ (a "complete presentation"). The available data can in some cases also be obtained dynamically: the learning algorithm may have access to additional information, such as an oracle that when given a conjectured grammar G' will either confirm its accuracy if $G'=G$, or provide a counterexample $s \in L(G) \setminus L(G')$ otherwise ("learning from a teacher").

One of the earliest formalizations for what it means to "learn" a grammar is "learning in the limit" [8], in which (informally) the learning algorithm requires only a finite amount of data to produce a correct answer from which it never then deviates. Note that this is not a particularly strong model of learning: the amount of data required can be arbitrarily large so long as it is finite, and it places no restriction on the size of the learned automata; it need not be minimal or even bounded. More realistic models of learning include the "Probably Approximately Correct" (PAC) framework [9], rather than requiring $L(G') = L(G)$ always, we might require only that over some distribution of strings and some distribution of training data T, we require that $P_T(P_D(s \notin L(G') \cap L(G)) < \epsilon) \geq 1 - \delta$; in other words, the two grammars disagree on at most some bounded proportion of strings (i.e. our proposed grammar is "approximately correct") and that this "probably" occurs, learning such an approximately correct grammar on at least $1-\delta$ of all possible training sets.

Regardless of the family, general results considering these various models are not encouraging. One of the earliest considered cases was examined by Gold [8]: that of learning in the limit from "presentations" as described above, where the learner is provided with data, cannot form its own queries, and must eventually converge to an equivalent grammar (it may only make finitely many mistakes). In this area of learning by examples, Gold's Theorem [8] demonstrates learning even regular languages is not possible from only positive examples (indeed, any class that contains all finite languages and at least one infinite language cannot be inferred from a positive presentation). Given a finite set of both positive and negative examples, the work of [10] shows that the decision question of whether there is an n-state DFA that agrees with the finite data set is NP-complete. Through a reduction to Boolean formulas, the work of [11] shows that PAC learning of DFAs from both positive and negative examples – the standard "supervised learning" context perhaps more familiar to those with a background in machine learning – is at least as hard as factoring Blum integers, and so under current cryptographic assumptions does not appear to be polynomial-time tractable. As DFAs occupy one of the lowest levels of the Chomsky hierarchy, it immediately follows that any grammar that requires more power than that offered by a DFA must be at least as hard as a DFA to learn[1].

Active learning – in which the learning algorithm can form some kind of query to narrow its search – provides slightly more positive results

Active learning – in which the learning algorithm can form some kind of query to narrow its search – provides slightly more positive results. The closest counterpart to the notion of active learning encountered in (non-grammatical inference oriented) machine learning is the model of an "informant," in which case the algorithm is allowed to construct query strings, and has access to an oracle which responds to membership queries for proposed strings. When learning with an informant, context-sensitive grammars may be learned in the limit [8]. The more powerful model of a "teacher" as described above (an informant who will provide counterexamples to proposed grammars) allows for yet more powerful inference. Under the model of a teacher, some context-free grammars may be learned in polynomial time with respect to the number of states in the underlying automata [12]. Finally, the use of "simple examples" as training data allows for learning minimal DFA in polynomial time [13], however simple examples – informally – are training items carefully chosen to guide the learning algorithm to the correct answer, and so in addition to being specific to the learning algorithm and the grammar, do not correspond to a more conventional learning approach where the distribution of the input data cannot be controlled.

In addition to variations in available input to the algorithm, various modifications of the grammatical classes to which the unknown language may belong have also been considered. The class of pattern languages [14] can be shown to be learnable in the limit from positive data due to the property of "finite thickness" [15] – loosely, that there is no string that is found in infinitely many members of the class and therefore for any given subset of grammars in the class there must be some production which is unique to it – which is shown to be sufficient for identification in the limit. While pattern languages themselves are difficult to apply to many empirical problems in intrusion detection [16], the notion of finite thickness may in many cases be applicable. A related notion is that of elasticity [17]. A class of languages has "infinite elasticity" if and only if one can find an infinite sequence of productions such that for all n, the first n productions are found in some grammar in the class while production $n+1$ is not; if a class does not have infinite elasticity, then it is said to have finite elasticity and it can again be inferred in the limit from positive data. Finally, if a class of grammars has a family of "characteristic sets" ('condition 1' in [15]) such that each language in the grammar has an associated finite characteristic set, and showing that the characteristic set for language i exists within language j is sufficient to show that language i is a subset of language j, then languages from that class are learnable in the limit.

Taken in whole, these results do not paint a particularly positive picture of grammatical inference. Even the simplest classes of grammars we might encounter have extremely poor "learnability," and rendering them easier to learn often requires some degree of interactivity or side information. Significant restrictions on the class of grammars which we may wish to perform inference on allows for substantial reductions in complexity, however we have little guarantee for novel protocols that we may rely on such restrictions. In the following section, we briefly examine major classes of machine learning that have been proposed for intrusion detection, and relate the results in this section to the practical problems posed by such learning systems.

1 Although see also [14], in which a class of languages that is polynomial-time learnable from examples is provided that is orthogonal to the Chomsky hierarchy.

Machine learning and Intrusion detection

The literature on machine learning and intrusion detection is vast (see references in [1] for a partial overview; also, short reviews by [18] and [19] which contain more details about specific machine learning methods that have been attempted); however, it divides broadly into the two main categories of "anomaly detection" and "signature inspired" [20] (in machine learning terminology, "supervised learning"). Anomaly detection, including such systems as Anagram [21] and McPAD [22], focuses on constructing a "normal" model of traffic and producing alerts when traffic that does not fit this model is observed. Supervised learning systems (see [23], [24], and [25] for representative examples) are provided with both malicious and benign traffic, and attempt to learn rules to distinguish them. While less common in the domain of intrusion detection, active learning (i.e. interactive) approaches for outlier detection have been presented as well, as in [26].

In all cases, the general formulation of the problem is approximately the same. Network messages are – by construction – designed to be parsed and interpreted by a machine, and hence can be characterized as formal grammars which accept and reject on specific strings. Within the space of all possible strings M received by the network service, there is the subset $A \subseteq M$ of messages that are accepted by the network endpoint; this subset itself contains $S \subseteq A$ of "safe" messages that represent normal use of the endpoint. In learning a function that can identify which subset of a given message lies within – in particular for some string s whether or not $s \in S$ – we are constructing a recognizer for S, thus placing the problem exactly within the domain of grammatical inference.

Consider a more realistic scenario in which several protocols may be present in the same set of network traffic

Consider, for example the case of a standard machine learning approach in which we are presented with samples of normal traffic from S, and hostile traffic from the set $A\backslash S$, each appropriately labeled, and we wish to train a function to determine whether a candidate string lies in one set or the other. This is precisely the problem of learning a grammar from a complete presentation, and as such we may readily apply existing results. Even if we make the simplifying assumption that the protocol under consideration (or the union of protocols if deployed on a multi-protocol endpoint) is regular, we are still attempting to learn a Discrete Finite Automata from a complete presentation. If we wish to learn it precisely (in the limit) then we have that the problem is NP-complete [10]. If we wish to learn it in a more practical sense (i.e. PAC), then we have that the identification problem is "merely" cryptographically hard [11]. This forces us to accept the conclusion that even if we obtain empirically good performance for a particular algorithm in a particular setting, we cannot be sure that it will generalize to a new domain.

If we consider (as in McPAD [22]) that only positive ('normal') data S is available, and continue to assume that we are observing then we are attempting to learn a grammar from its positive presentation, with all associated complexities. While specific examples of grammars are clearly at least PAC-learnable in this setting, as shown by the results of [22], it follows immediately from the difficulty of learning from a positive presentation that McPAD must fail to generalize to at least some classes of grammars; whether or not those grammars are of practical relevance to intrusion detection cannot be decided in any fashion other than empirically. We are thus left with no foundational guarantee of correctness; simply empirical observations.

Clearly, when we consider a more realistic scenario in which several protocols may be present in the same set of network traffic, the problem often becomes significantly more difficult; the problem of learning the mixture of grammars is at a minimum at least as hard as learning the most complex one, and depending upon the closure properties of that class, may in fact be more difficult. While most languages in the Chomsky hierarchy are in fact closed under unions, it is often not clear whether or not restricted classes (such as those that have finite thickness) may be.

While somewhat outside the realm of network intrusion detection, more powerful inference models such as learning from an informant have been shown to generate positive results in areas such as developing usable models of program execution (stateful typestates) [27]. This approach obtains a direct generative model of program outputs which can be examined for various security properties. Standard fuzzing techniques [28] are perhaps a more direct application within the security domain, in which a subset $C \subseteq A$ of inputs that lead to crashes is learned from interaction with a program, frequently combined with additional information about the execution of code paths [29], however these methods do not typically produce formal descriptions or generative models of incorrect inputs, and rather seek to enumerate a useful subset of them, typically (in defensive settings) attempting to reduce the size of the set A. The work of [30] explores methods to leverage attacker knowledge in constructing fuzzing inputs via a descriptive language, which could be used in an iterative fashion to eventually describe a subset of the target grammar.

In many cases, we do not even require the results of grammatical inference to show that a particular classifier cannot (provably) learn a sharp distinction between malicious and benign traffic. A key step in any machine learning process is that of 'feature extraction' in which the raw data that is to be classified is converted into some numerical representation that can then be operated on by the learning algorithm. N-grams (and minor variations on the concept such as skip-grams) are the core feature representation used in a number of anomaly-based intrusion detection systems, including Anagram [21] and McPAD [22], in which every n-byte substring within the payload is tabulated (for instance, the string "learning" would have 3-grams of "lea", "ear", "arn", "rni", and so on).

However such representations can be shown to be insufficiently powerful to distinguish between many members of the class of regular languages. For example, the rather trivial regular languages $(ab)^x(ba)^y(ab)^z$ and $(ba)^x(ba)^y(ba)^z$ cannot be distinguished from each other on the basis of 2-grams (note that the 2-grams bb and aa both appear exactly once in each, with variable numbers of ab and ba tokens), while constructing a recognizing DFA for each is trivial. Similar counterexamples can be constructed for n-grams of arbitrary length. This immediately implies that any learning algorithm that first reduces a sequence to n-gram counts is *a priori* incapable of learning large subsets of the class of regular grammars, and as a consequence we may expect that – even if empirically good performance is shown on some data sets – this performance cannot be relied upon to generalize.

Other feature representations are also used. Perhaps most widely known are those of the (now severely out-of-date but still regularly studied) KDD'99 data set [31], which parses the network traffic to extract a number of features suggested by expert knowledge to be of use. In addition to "metadata" describing flow-based properties such as the duration of the connection and the number of concurrent connections to the same host, a number of content-based features are extracted from both the payload of the packets (e.g., the presence of a shell prompt, extracting FTP commands to obtain a tally of outbound ones, etc.) and headers of the packets (identifying the network protocol, various ports and addresses, protocol-specific flags, and so on). These features obviously make no attempt to model any significant portion of the content of the packets, and so make the prospect of inferring a grammar from them infeasible; at best, some of the manually extracted features act as "telltales" for specific attacks, and thus allow what is effectively signature-based detection.

Manually extracted features act as "telltales" for specific attacks, and thus allow what is effectively signature-based detection"

And indeed, the most effective current approach in intrusion detection remains (anecdotally at least) signature-based solutions [1] such as Snort [3]. The effectiveness of such solutions can be explained precisely within the context of grammatical inference, as a well-written content-based signature is equivalent to a production that is not (or is very rarely) a production of the grammar underlying "good" traffic, and hence form a telltale for the set $A \setminus S$. And indeed, Snort and Bro [2] both contain sophisticated pattern-matching rules that are capable of recognizing a wide range of malicious traffic, in effect acting as small recognizers for subsets of the malicious grammars under consideration. It is worth noting, however, that the rule generation in this case is often done by hand, and even when done in an automated fashion is typically attempting to match a finite subset of malicious traffic for a specific attack, and then tested on a set of larger normal traffic to assess false positives; this is equivalent to the bounded version of the problem posed in [10], which can take place in polynomial time. A key distinction here is that – rather than attempting to model all 'safe' or 'malicious' productions – any method that produces some form of signature is attempting to model a finite number of productions of a single protocol under heavily supervised conditions, and so does not address the question of novel attacks that machine learning-based solutions are often attempting to address [1].

Conclusion

The high performance of machine learning in other domains has stimulated significant interest in applying it to network security, however (as noted in [1]), despite the breakneck pace of major successes with machine learning in many other domains, and the large amount of effort spent to produce machine learning-based intrusion detection systems, in practice most major network defense providers focus continue to use signature-based methods which have been in active use since the late 1990's.

Drawing on the extensive literature on grammatical analysis, we propose that this is a reflection of a fundamental difference between more conventional domains of machine learning and network security. In particular, because network security – particularly network security applications that focus on analysis of packet contents – operates on the domain of formal grammars that are rigorously interpreted (as compared to the domain of natural language translation, where human intuition can often "fill in the gaps" in translation), it is an intrinsically difficult problem that a) is demonstrably intractable in the most general case, and b) cannot be addressed with the relatively crude features that appear to be most common in the literature. While some modest success has been recently realized in applying sequence-to-sequence models (thus at least partially avoiding the question of feature spaces) for grammatical inference in specific instances of specific protocols [32], there remains no method to demonstrate that such methods will generalize even to different instances of the same protocol, let alone novel protocols in the same class.

In fact, results from grammatical inference show that there is quite likely no general method that can be applied to arbitrary data to separate benign and malicious traffic; any practical method should therefore be restricted to a particular domain, analyze that domain carefully, and at least attempt to investigate what properties of the protocol under analysis may allow it to be effectively learned. The empirical effectiveness of Snort and Bro signatures suggest that the domain of malicious traffic is likely more tractable, and may be easier to learn. The appearance of particular byte sequences in malicious but not benign traffic can be viewed (informally) as evidence that the class of malicious languages is of finite elasticity (due to the absence of a limit language) within the class of all protocols that can produce accepting inputs to the system under consideration, thus supporting identifiability. Feature representation is also important. N-gram based features in particular will quite often be insufficiently powerful to model complex grammars or protocols; in some cases, sufficiently large values of *n* may be able to overcome this limitation for specific subclasses of protocols, however this is likely to be highly problem specific, and requires careful evaluation for any given proposed system.

While significant open questions remain – such as methods for performing inference on the restricted classes of grammars that in practical terms make up many existing protocols – the immediate results of applying grammatical inference theory to machine learning for intrusion detection both help explain the lack of widespread adoption of such systems, and suggest appropriate avenues for future work.

REFERENCES

[1] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *IEEE symposium on security and privacy, 2010*.

[2] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23, pp. 2435-2463, 1999.

[3] Roesch, Martin, "Snort: Lightweight Intrusion Detection for Networks," in *LISA* , Seattle, Washington, 1999.

[4] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Transactions on Information and System Security (TISSEC),* vol. 3, no. 3, pp. 186-205, 2000.

[5] A. Kott, "Towards fundamental science of cyber security," in *Network Science and Cybersecurity*, New York, Springer, 2014, pp. 1-13.

[6] M. Sipser, Introduction to the Theory of Computation., Boston, MA: Thomson Course Technology, 2006., 2006.

[7] C. De la Higuera, Grammatical inference: learning automata and grammars, Cambridge University Press, 2010.

[8] M. E. Gold, "Language identification in the limit," *Information and control*, vol. 10, no. 5, pp. 447-474, 1967.

[9] L. G. Valiant, "A theory of the learnable," *Commun. ACM,* vol. 27, no. 11, pp. 1134--1142, 1984.

[10] M. E. Gold, "Complexity of automaton identification from given data," *Information and control*, vol. 37, no. 3, pp. 302-320, 1978.

[11] M. Kearns and L. Valiant, "Cryptographic limitations on learning Boolean formulae and finite automata," *Journal of the ACM*, vol. 41, no. 1, pp. 67-95, 1994.

[12] D. Angluin, "Learning regular sets from queries and counterexamples," *Information and computation*, vol. 75, no. 2, pp. 87-106, 1987.

[13] V. H. Rajesh Parekh, "Learning DFA from simple examples," *Machine Learning*, vol. 44, pp. 9-35, 2001.

[14] D. Angluin, "Finding patterns common to a set of strings," in *Proceedings of the eleventh annual ACM symposium on Theory of computing*, 1979.

[15] D. Angluin, "Inductive inference of formal languages from positive data," *Information and control* , vol. 45, no. 2, pp. 117-135, 1980.

[16] K. N. Wood and R. E. Harang, "Grammatical Inference and Language Frameworks for LANGSEC," in 2015 *IEEE Security and Privacy Workshops*, 2015.

[17] T. Motoki, T. Shinohara and K. Wright, "Correct definition of finite elasticity," Research Institute of Fundamental Information Science, Kyushu University Japan, 1990.

[18] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994--12000, 2009.

[19] P. Laskov, P. Düssel, C. Schäfer and K. Rieck, "Learning intrusion detection: supervised or unsupervised?," in *International Conference on Image Analysis and Processing*, Heidelberg, 2005.

[20] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," 2000.

[21] K. Wang, J. J. Parekh and S. J. Stolfo, "Anagram: A content anomaly detector resistant to mimicry attack," in *Recent Advances in Intrusion Detection*, Heidelberg, 2006.

[22] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto and W. Lee, "McPAD: A multiple classifier system for accurate payload-based anomaly detection," *Computer Networks* , vol. 53, no. 6, pp. 864--881, 2009.

[23] W. Hu, W. Hu and S. Maybank, "Adaboost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* , vol. 38, no. 2, pp. 577--583, 2008.

[24] P. Sangkatsanee, N. Wattanapongsakorn and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Computer Communications* , vol. 38, no. 18, pp. 2227--2235, 2011.

[25] O. Linda, T. Vollmer and M. Manic, "Neural network based intrusion detection system for critical infrastructures," in *International Joint Conference on Neural Networks*, 2009.

[26] N. Abe, B. Zadrozny and J. Langford, "Outlier detection by active learning," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.

[27] H. Xiao, J. Sun, Y. Liu, S.-W. Lin and C. Sun, "Tzuyu: Learning stateful typestates," in *28th International Conference on Automated Software Engineering* , 2013.

[28] M. Sutton, A. Greene and P. Amini, Fuzzing: brute force vulnerability discovery, Pearson Education, 2007.

[29] M. Zalewski, "american fuzzy lop," [Online]. Available: http://lcamtuf.coredump.cx/afl/.

[30] D. Aitel, "The advantages of block-based protocol analysis for security testing," Immunity Inc, 2002.

[31] S. Stolfo, "The Third International Knowledge Discovery and Data Mining," University of California Irvine, 2002. [Online]. Available: http://kdd.ics.uci.edu/databases/kddCup99/kddCup99.html .

[32] C. Sheridan and R. Harang, "Grammatical Inference and Machine Learning Approaches to Post-Hoc LangSec," in *IEEE Security and Privacy Workshop on Language-Theoretic Security*, 2016.

ABOUT THE AUTHOR

Dr. Richard Harang received his PhD in Statistics and Applied Probability from the University of California Santa Barbara in 2010. After a year of postdoctoral research in the Computational Science and Engineering group under Dr. Linda Petzold, he began work at the U.S. Army Research Laboratory in 2011 focusing on applications of statistics and statistical machine learning to problems in network security. His current research interests include machine learning on structured data, analysis and attribution of source code and binary samples, and using generative models of time series data to explore properties of the underlying process.

SYNERGISTIC ARCHITECTURE FOR HUMAN-MACHINE INTRUSION DETECTION

BY: Noam Ben-Asher and Paul Yu, Army Research Laboratory (ARL)

ABSTRACT: *Modern day detection of cyber threats is a highly manual process where teams of human analysts flag suspicious events while using assistive tools such as Bro and Snort. It is the analysts' ability to discern suspicious activity and authority to make decisions on threats that place humans into central roles in the threat detection process. However, over-reliance on human ability can lead to a high volume of undetected threats. As the tempo, diversity and complexity of cyberspace threats continues to increase, this shortcoming can only worsen. Therefore, there is a need for a new detection paradigm that is largely automated but where analysts maintain situational awareness and control of the process. We propose a synergistic detection process that captures the benefits of human cognition and machine computation while mitigating their weaknesses. The analyst provides context and domain knowledge, and the machines provide the ability to handle vast data at speed.*

1 Introduction

As the network takes an increasingly larger role in military operations, there is an increasingly urgent need to plan, execute, and assess operations in cyberspace. A key capability is the accurate, fast, and agile assessment of network actions that are performed by friendly and hostile participants. Of particular interest is the detection of threats, i.e., actions that harm the network.

Despite constant advances in automated threat detection, human analysts and decision makers continue to play critical roles in the struggle to ensure secure networks [1]. A typical threat detection process, as illustrated in Figure 1, starts with observations of network activities that are then filtered by a set of detection tools [2]. The human analyst uses these tools and their output (i.e,, information summaries and alerts) to inform the final decisions of what threats are present in the network and their possible impact on the mission.

Many features of the cyber environment challenge the capabilities and capacity of human cognition, including the ever-increasing volume of network data, the wide variety of data sources, and the frequent and unexpected changes in the network. The analyst assumes bears much responsibility for making quality decisions within the current detection process. As a result, much emphasis is placed on improving analyst training and experience. However, the quality of the decisions is limited by the quality of information presented to the analyst and by the ability to sufficiently process this information. The analyst may even impede detection as core human capabilities like memory capacity and processing speed cannot be easily enhanced to match the ever-growing volume of network traffic. Furthermore, changing the analyst decision making patterns requires a deliberate effort that can happen at a slower pace compared to the appearance of new cyber attack patterns.

We propose a framework for an adaptive detection process where the evidence collector, detection engine, and human analyst work together to share information and make higher accuracy decisions at higher speed. This framework aims to enable efficient investment of the analyst's valuable, though limited, cognitive resources into the detection processes. The goal of the framework is to establish the foundations for a protocol between detection components such that relevant information can be transferred amongst them in order to have an adaptive process that can detect new threats. In contrast to the threat detection workflow that depends heavily on the human, the proposed framework that depicts a process where the components support and complement each other. We introduce within our framework:

- Varying levels of analyst involvement in detection
- Adapting detection according to the threat's life cycle
- Characterizing the types of interactions between detections components

Detection framework will also improve the ability of the human analysts to acquire and maintain situational awareness in cyberspace

While the primary role of the human analyst is to identify the real threats within a large volume of alerts, analysts are also required to gain and maintain cyber situational awareness [3]. More specifically, analysts uncover the meaning of the observed network behavior (e.g., what is that nature and origin of the behavior?) and project how this behavior might evolve and effect the mission (e.g., what the attacker will do next?). This information informs the decision of the analyst how to respond to the situation and what defense will be effective [4, 5]. Though the defense of the network is outside the scope of this article, we believe that the proposed detection framework will also improve the ability of the human analysts to acquire and maintain situational awareness in cyberspace.

2 The Need for Analyst-in-the-Loop Cyber Detection

The nature of cyber attacks has undergone significant changes in recent years, as evidenced by the emergence of more cases of destructive Advanced Persistent Threats (APTs). Leading government agencies and cooperations have experienced increasingly sophisticated attacks that exhibited distinctive characteristics compared to more traditional cyber threats. APTs are typically well sponsored and organized cyber campaigns with very specific and targeted objectives. One of the key objectives of an APT is to achieve a persistent foothold in a system for long period of time by using zero-day exploits, careful propagation, and a small footprint. All of these measures aim to increase the likelihood of remaining undetected (stealth), especially when the defender relies on automated detection tools. Detecting APTs require extensive human involvement and effort when conducting forensic analysis [6].

Throughout the detection workflow, the analyst should be able to utilize various decision support tools. These tools assist the analyst by filtering, mining, summarizing and visualizing data to speeding up the analysis. In some cases, where there is sufficient confidence in the automated diagnosis, tools can even determine the threat and initiate the appropriate response. However, in the current linear detection workflow (Figure 1), the analyst has no influence on the way evidence is collected or processed, and eventually is required to make decisions based on an externally imposed information flow. The lack of control and transparency can hinder the analyst's ability to detect threats quickly and accurately. In mission-critical settings such as threat detection, the analyst needs to trust the supporting tools and also have access to the reasoning behind the recommendations or alerts they generate in order to correctly determine whether or not to accept or reject a recommendation [7].

To detect APTs and ensure high level of mission performance by establishing trust in a decision support tool and compliance to the recommendation it generates, the analyst should have the ability to interact with the underlying detection mechanisms throughout the detection process (as illustrated in Figure 2) and not only at the very end [8]. Furthermore, by the agency of such interactions the analyst can infuse contextual information that can support and improve detection accuracy and speed. The analyst can also continuously tune the detection processes in response to emerging threats and provide instruction on how the detection processes should adopt to changes in the attack surface and attackers' capabilities.

Recent studies on Human-Data Interaction (HDI) propose a human centric approach to understand and develop interactions with dig data, dynamic data flows, algorithms, automated reasoning mechanisms and visualizations [9]. HDI core characteristics of the interactions can be adopted to cyber intrusion detection domain by situating the analyst as an influential component in each and every part of the detection process. Accordingly, in our synergistic analyst-in-the-loop framework we highlight three high-level aspects of the analyst interaction with detection. The first aspect is *legibility*, encapsulating the notion that both data collection mechanisms and analytics algorithms should be transparent and comprehensible to the analyst. The second aspect is *agency*, concerned with the idea that the analyst should have the capacity to control and influence data collection and management processes.

Lastly, *negotiability* addresses the ability of the analyst to influence the data processing and analytics so that data can be processed using different methods, for different purposes and in different context.

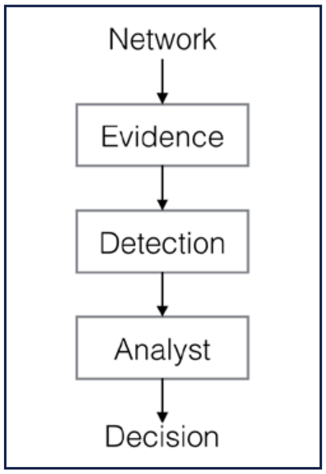


Figure 1: Linear detection process

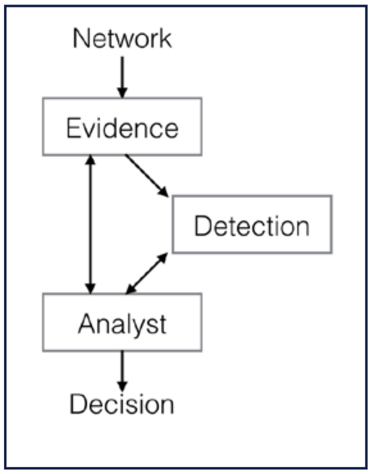


Figure 2: Synergistic detection process

2.1 Varying levels of analyst involvement in detection

The attention of the human analyst is a valuable and scarce resource. As such, human attention and cognitive capabilities should be allocated in the most beneficial way that supports the most critical tasks. Other tasks, that do not benefit significantly from human analytical capabilities or play a less critical role, can be partly or completely automated. Parasuraman and colleagues [10] propose a model for types and levels of human interaction with automation. Among other implementations, this model can be used to assign different levels of automation to the four stages of information processing (information acquisition, information analysis, decision selection, and action implementation). Cyber intrusion detection relies upon information processing, where evidence collection is equivalent to information acquisition, the detection engine operation is equivalent to information analysis and the analyst decision and response stages correspond to decision selection and action implementation stages. Therefore, in each of the detection stages, analyst involvement can range from high to low. high analyst involvement corresponds to a low automation level where the analyst must take all decisions and actions, while low analyst involvement corresponds to high automation where detection processes operate autonomously. With respect to the analyst decision of whether or not there is an intrusion, we consider four levels of collaboration. At the lowest end of the collaboration scale, the analyst operates unaided and detects threats in the raw packet level data. Automation can increase by adding detection mechanisms that provide recommendations (i.e., alerts) to the analyst. The actual level of automation depends on the definition of the analyst's role when responding these alert. The analysts can acknowledge correct detections and detect additional threats that were missed by automated detection. Alternatively, when automation is more extensive and trustworthy, the analyst role can be limited to detecting missed threats and canceling false alerts. At the highest level, all aspect of detection is automated and analyst can direct full attention towards selecting the best response to detected threats.

2.2 Adapting detection to the threat's life cycle

Threats may be characterized by how well they can be detected through automated methods. This tends to be highly correlated with the understanding of the threat. From the analyst's perspective, the threats go through a progression of understanding which is also captured by the life cycle of a threat [11].

Initially the threat is unknown to the analyst; such an unknown threat often targets an unknown vulnerability and is referred to as a *zero-day* exploit. Encountering such a threat requires much forensic examination and study of normal and abnormal network behaviors in order to isolate the threat and gain a preliminary understanding regarding its existence. There can be significant human involvement in this stage, from determining (i.e., labeling) the activity as malicious, to identifying the evidence that indicates the presence of the threat, to identifying its impact. As more examples of the threat are observed, more information is revealed to the human. Eventually, human understanding improves to the point where accurate detection mechanisms can be automated and programmed into the intrusion detection systems. This automation, in parallel to correction (e.g., patching) of the vulnerability, shifts the load away from the analyst. It is critical to provide the analyst support through automation in this part of the threat life cycle as following disclosure the volume of cyber attacks that utilize the specific threat can increase by up to 5 orders of magnitude [12].

At that point, the detection of the now well-understood a threat can be safely relegated primarily to automated mechanisms. The analyst remains responsible for making the final decision based on the automated detection outputs. However, the majority of the processing to ascertain the probability of the threat is done without human involvement.

The dynamics of the threat and detection life cycle highlight the need to allow different levels of automation in the detection processes. This ability is tightly coupled with the analyst's ability to interact with the detection processes, understand how they operate, and influence their operation. Eventually, the needs and role of the human analysts can constantly change and as such, detection processes should be flexible enough to facilitate the operation of the analyst in constantly changing levels of understanding and awareness to cyber threats.

3 Synergistic Analyst-in-the-Loop Cyber Detection

To formulate and demonstrate the synergistic architecture for intrusion detection, we consider the following simplified detection process. Cyber *activity* impacts the environment. The state of the environment is measured by *monitors* and yields *evidence*. Evidence is provided to a detection mechanism (Inference), which decides *hypotheses* by assigning them *weights*.

Based on existing research on intrusion detection and human data interaction, our hypothesis is that having each component interact and share relevant information with the other components allows better decisions to be made at higher speed about threats both new and old. We first introduce the components of the detection process and then discuss the interactions between them.

The state of the environment is measured by monitors and yields evidence

3.1 Components

The three central components of the detection framework are the Evidence Collection Mechanism (denoted by ℂ), the Detection/Inference Engine (D), and the human analyst (A) (see Figure 3). The ultimate decision of threats is made by A with support provided by D and ℂ.

› ℂ, the evidence collector, manages the monitors that report information about the behavior of observable activities for use by the detection engine. The monitors can be deployed at the network level to inspect traffic (e.g., deep packet inspection) or at the host level to monitor processes. The information is then processed into the evidence that is requested by D or A.
› ℂ is aware of the types of evidence that it can collect as well as the cost of doing so. The evidence has many properties such as update frequency, bit rate, variance, reliability, etc. It may set the frequency of the evidence collection to trade accuracy with storage or bandwidth requirements. It may also be aware of the reliability of the evidence, a metric that can vary in real-time (e.g., through interfering processes, network traffic, etc.) These capabilities allow ℂ a variety of behaviors. For example, it can calculate the cost to deploy a proposed set of monitors, or it can message the analyst when the reliability of the collected evidence has been degraded.
› D, the detection engine, processes the available evidence and generates likelihoods for possible threats. The important capability of D is the handling of vast quantities of information. It is able to refine its detection and adapt to changes in the environment through human interaction (e.g., supervised machine learning [13]).
› For a particular threat, D understands to some degree the relationship between the likelihood and evidence, e.g., as necessary or sufficient conditions for a threat. It is able to provide the human with an understanding of how the threat likelihoods are calculated. This implies that its internal logic can be shared with the human in a legible format.
› A, the human analyst, decides which threats are occurring, generally with consideration given to the likelihoods computed by the detection engine.

Rather than focusing on the structure of D and ℂ, this note focuses on how the human analyst can interact with D and ℂ. In the following, we introduce the supporting framework for these interactions.

3.2 Interactive Detection Workflow

Figure 3 illustrates the detection flow where solid arrows indicate on the direction of information from source to destination. The detection flow starts with network and host activities that are observed by a set of monitors. The evidence collector ℂ deploys the monitors and converts the gathered information into evidence. The detection engine D makes inferences on which threats are likely given the provided evidence. The likelihoods are presented to the analyst as a set of weights.

The goal of the analyst A is to observe the set of hypotheses and weights and decide what types of activities occur. In order to improve the performance (accuracy, efficiency, etc.) of the overall detection flow, the analyst can interact with ℂ and D as illustrated by the dashed lines in Figure 3. ℂ and D can also interact directly. Such interactions can be the propagation of analyst's interaction with one component to the other, or a result of the ongoing execution of a process. We formalize the interaction between the components of the detection processes in terms of queries and operations.

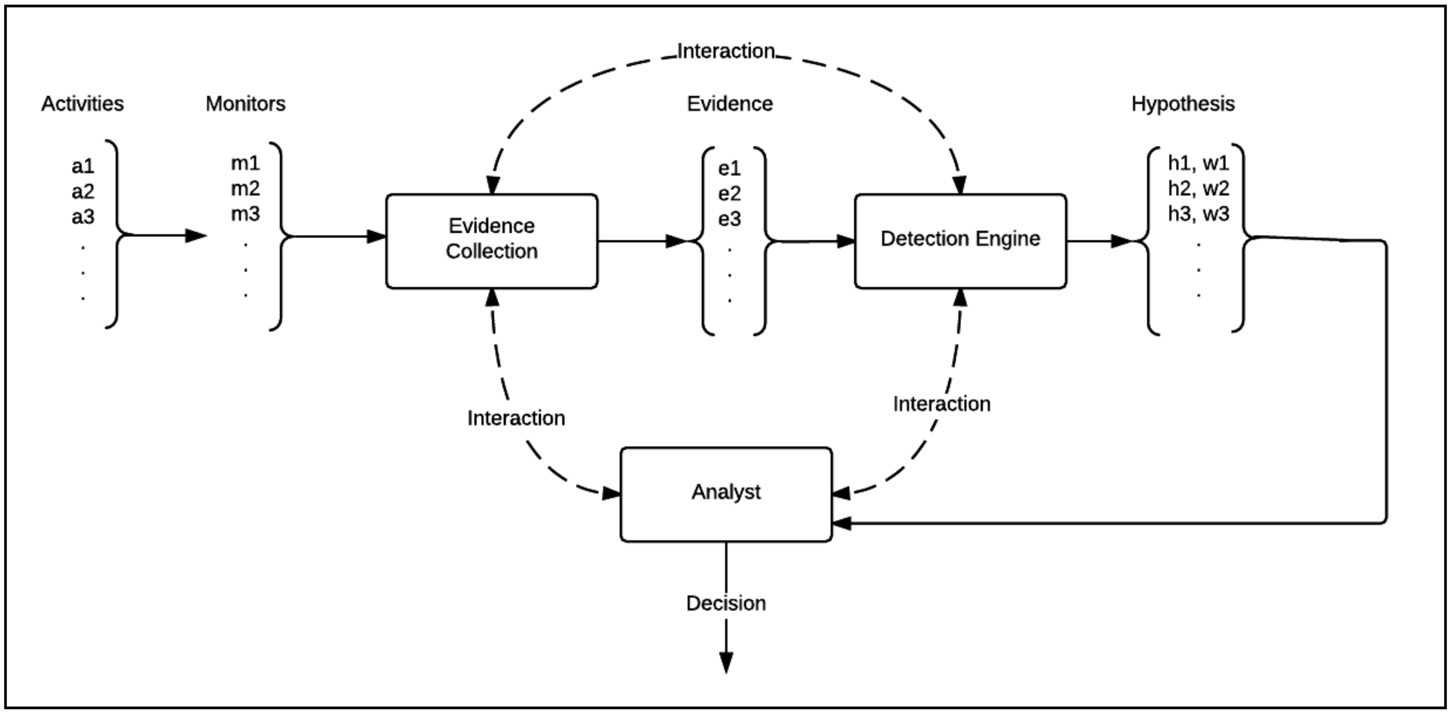


Figure 3: Synergistic detection process flow of information (solid lines) and interactions (dashed lines)

3.3 Interactions

Interactions between detection processes fall into one of two possible categories:

1. Queries - Any type of request for information (i.e., a query) that do not change the operation of a process. A query is always followed by an *answer*.
2. Operation - Any type of request that changes how processes performs. An operation is always followed by *feedback* indicating at least whether the operation was successfully completed or not.

Within a synergistic detection framework queries and operations can take many forms. Instead of enumerating all possible queries and operations, to illustrate the flow of interaction and of some of the most significant queries and operations we use the following a simplified example.

1. ℂ deploys a set of monitors to track network activities.
2. Based on the collected evidence, D provides A alerts that the activities a_1, a_2 are likely, and that activity a_3 is unlikely. Here, we let a_1 represent an unauthorized privilege escalation, a_2 an ongoing SYN flood attack where large amounts of SYN packets is observed by ℂ, and a_3 a malware beaconing to a control server.
3. (a_1) Based on experience, A knows that the D generates accurate alerts for a_1 and as such confirm that a_1 is true. This type of update interaction provides feedback to D that reinforces the use of the mechanism that yielded a_1 with high likelihood.
4. (a_2) A is aware of the current state of the network and the ephemeral tasks the network support. As such A is capable of using this contextual information to dismiss the alert regarding a_2. Again, this interaction provides feedback to D and influences its future operation.
5. (a_3) With respect to a_3, A queries D for details regarding the type of evidence D when making the decision. Given the high risk of overlooking malware activity, A can submit a query to D what additional evidence is required to state whether a_3 occurs or not, with higher confidence. Based on the response, A can instruct ℂ through an operation to collected targeted evidence to resolve the ambiguity around a_3. ℂ processes the operation request and modifies the monitors to accommodate it. When complete, ℂ informs A on the successful execution of the request and D on the modifications in the stream of evidence. Now, with the additional evidence, D can provide A a more confident alert regarding activity a_3.

4 Conclusions

Currently, cyber attackers have an asymmetrical advantage over defenders. This unfavorable and vulnerable position calls for a robust and efficient intrusion detection mechanisms. While current detection workflow involves human defenders, and rely on their analytical capabilities, we argue that in order to improve detection and protect networks against sophisticated attacks there is a need for a non-linear and interactive analyst-in-the-loop approach. This approach posits that cyber defenders should have means to interact with and exert influence on each and every component of the detection processes. Furthermore, we posit that the role of the analyst is to lead and supervise automated detection processes, resolve ambiguity and provide contextual mission relevant information rather than handling large amounts of information and weeding out false alerts. Situating the defender as the controller of the detection process instead of a handler of alerts allows the defender to direct analytical capabilities to the tasks where their contribution has the maximal impact. Efficient allocation of the defender analytical capabilities improves the detection accuracy and speed. This study depicts an analyst in-the-loop detection framework and provides a description of the types of required interactions between the evidence collection, inference engine, and the analyst. The use of queries and operations to improve detection is demonstrated and establishes the foundations for more detailed operational definitions of the interactions.

ABOUT THE AUTHORS

Dr. NoamBen-Asher is a researcher at the Computational and InformationSciences Directorate at US-Army Research Laboratory. Before thisposition, Noam was a postdoctoral fellow at the Dynamic DecisionMaking Laboratory at Carnegie Mellon University. His primaryinterests lie at the intersection of cognitive science, decisionscience and human factors engineering, with a particular interest incyber security. In this field, he combines behavioral studies withcomputational cognitive modeling to study cyber defenders andattackers situation awareness and dynamic decision making processesin cyber warfare.

Paul Yu (Member, IEEE) received a B.S. in Mathematics, a B.S. degree in Computer Engineering, and a Ph.D. degree in Electrical Engineering, all at the University of Maryland, College Park. Since 2006, he has been with the U.S. Army Research Laboratory (ARL) where his work is in the area of signal processing for wireless networking and autonomy. He received the Outstanding Invention of the Year award in 2008 and the Jimmy Lin Award for Innovation and Invention in 2009, both from the University of Maryland, and a Best Paper award at the 2008 Army Science Conference.

REFERENCES

[1] N. Ben-Asher and C. Gonzalez, "Effects of cyber security knowledge on attack detection," *Computers in Human Behavior*, vol. 48, pp. 51–61, 2015.

[2] A. DAmico and K. Whitley, "The real work of computer network defense analysts," in *VizSEC 2007*, pp. 19–37, Springer, 2008.

[3] C. Gonzalez, N. Ben-Asher, A. Oltramari, and C. Lebiere, "*Cognition and technology*," in Cyber Defense and Situational Awareness, pp. 93–117, Springer, 2014.

[4] C. Zhong, J. Yen, P. Liu, and R. F. Erbacher, "Automate cybersecurity data triage by leveraging human analysts' cognitive process," in *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*, pp. 357–363, IEEE, 2016.

[5] O. S. Saydjari, "Cyber defense: art to science," *Communications of the ACM*, vol. 47, no. 3, pp. 52–57, 2004.

[6] N. Virvilis, D. Gritzalis, and T. Apostolopoulos, "Trusted computing vs. advanced persistent threats: Can a defender win this game?," in *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*, pp. 396–403, IEEE, 2013.

[7] K. Ehrlich, S. E. Kirk, J. Patterson, J. C. Rasmussen, S. I. Ross, and D. M. Gruen, "Taking advice from intelligent systems: the double-edged sword of explanations," in *Proceedings of the 16th international conference on Intelligent user interfaces*, pp. 125–134, ACM, 2011.

[8] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 46, no. 1, pp. 50–80, 2004.

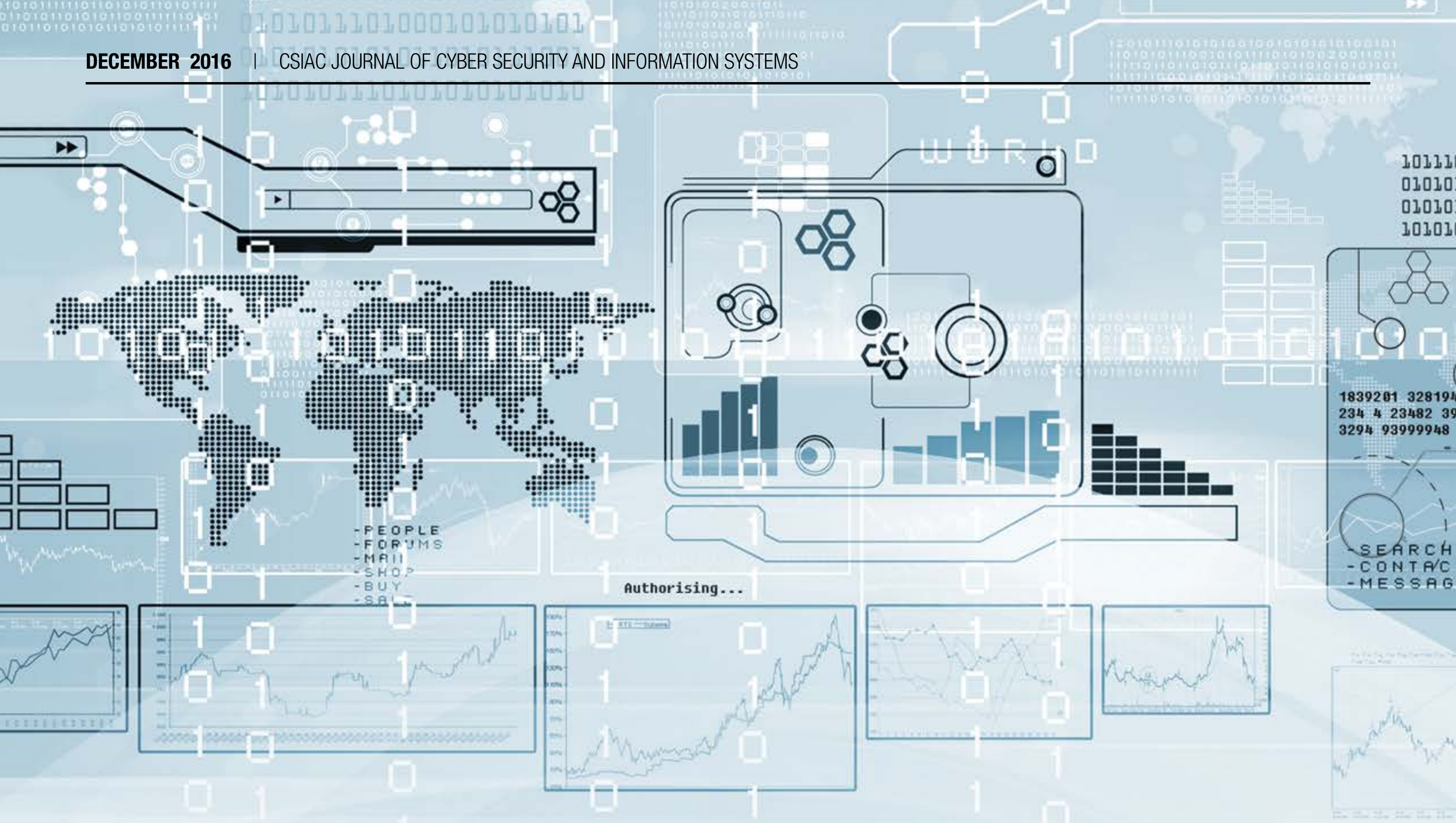
[9] R. Mortier, H. Haddadi, T. Henderson, D. McAuley, and J. Crowcroft, "Human-data interaction: the human face of the data-driven society," *Available at SSRN 2508051*, 2014.

[10] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[11] W. A. Arbaugh, W. L. Fithen, and J. McHugh, "Windows of vulnerability: A case study analysis," *Computer*, vol. 33, no. 12, pp. 52–59, 2000.

[12] L. Bilge and T. Dumitras, "Before we knew it: an empirical study of zero-day attacks in the real world," in *Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 833–844, ACM, 2012.

[13] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, "Ai2 Training a big data machine to defend," in 2016 *IEEE International Conference on Intelligent Data and Security*, pp. 49–54, April 2016.



RISK ANALYSIS WITH EXECUTION-BASED MODEL GENERATION

BY:Jaime C. Acosta, Army Research Laboratory (ARL),
Edgar Padilla, University of Texas at El Paso,
John Homer, Abilene Christian University,
Xinming Ou, University of South Florida

RISK ANALYSIS: *Analyzing risk is critical throughout the software acquisition lifecycle. System risk is assessed by conducting a penetration test, where ethical hackers portray realistic threat on real systems by exploiting vulnerabilities. These tests are very costly, limited in duration, and do not provide stakeholders with "what-if" analyses. To alleviate these issues, system models are used in emulation, simulation, and attack graph generators to enhance test preparation, execution, and supplementary post-test analyses.*

This article describes a method for developing models that can be used to analyze risk in mixed tactical and strategic networks, which are common in the military domain.

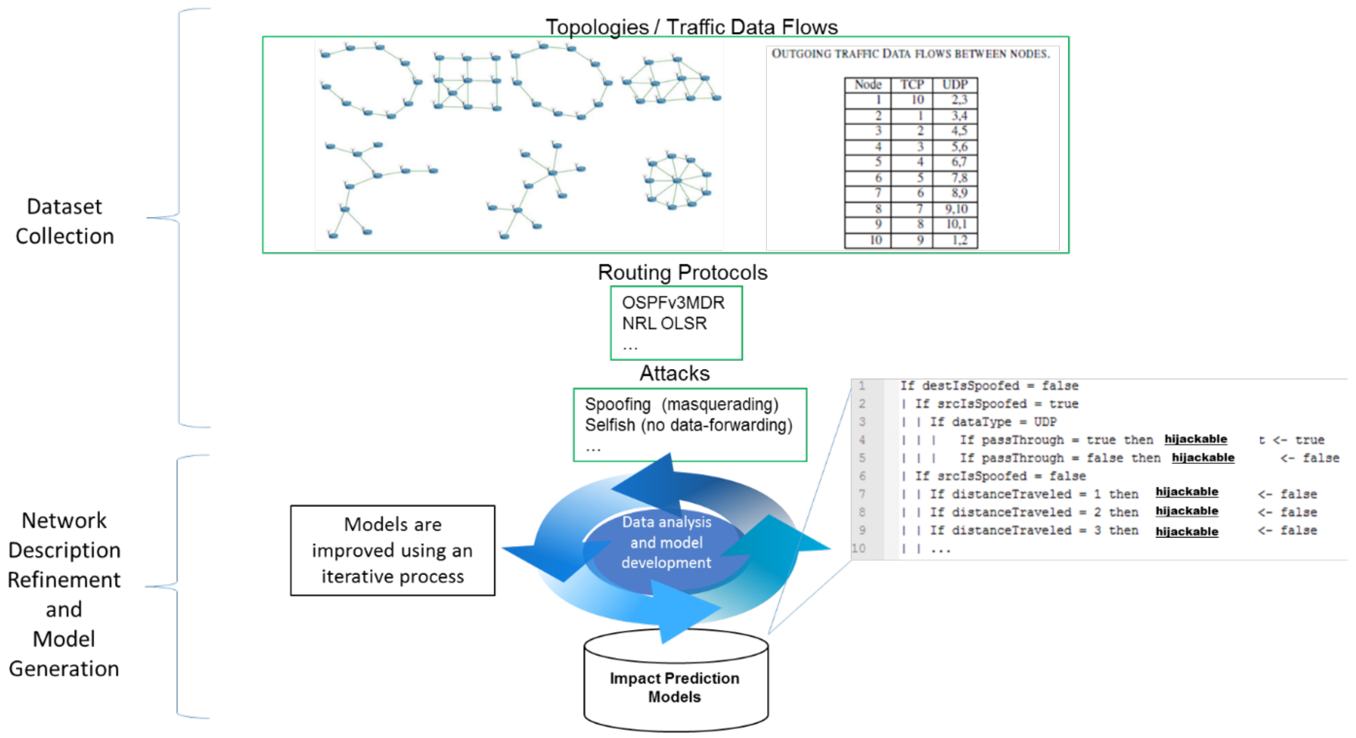
Execution-Based Model Generation

ARL has developed a model creation methodology that uses data collected from several black-box emulation executions. The generated models take the form of decision trees or other complex algorithms and formulas. This approach differs from traditional workflows where models are created and tested before or alongside system development. Some issues with the traditional approach include lack of synchronization between the actual system and the models due to changes in requirements, high cost of manually developed high-accuracy models, and unavailability (either for legal reasons or due to non-existent models). The novel approach starts instead with the end-product (i.e., the executable code). This also makes it possible to extract additional, incidental behaviors such as resilience to adversarial attacks. This methodology has been used to develop models for predicting success or failure of traffic hijacking attacks in MANETs. Unlike previous work in MANET evaluation techniques, the developed models generalize across scenarios and can be used to assess risk in attack graphs.

MANET Security Evaluation

Apart from penetration testing, simulation and emulation are commonly used to evaluate MANETs. This consists of executing multiple scenarios with varying conditions, such as topology and routing protocol. During scenario execution, an attack is executed and performance data (delay, throughput, goodput, etc.) are measured (Pathan, Al-Sakib Khan, 2016). Although this approach is non-exhaustive, well designed scenarios may be credible sources for evaluating security (Andel & Yasinsac, On the credibility of manet simulations, 2006.) Simulation is usually faster, but is more prone to inaccuracies because many times performance gains rely on using abstracted or otherwise incomplete behavior of the network stack and other processes. Emulation is capable of executing real binaries in real runtime environments (operating system, network stack, etc.), but only in realtime. In either case, in addition to being non-exhaustive, results do not generalize to untested scenarios.

Other evaluation techniques include formal methods and machine learning. The former are exhaustive approaches that describe systems mathematically and then, through rigorous analysis, prove or disprove security goals; these methods work for only small and non-mobile networks (Andel & Yasinsac, 2007.) Prior machine learning


Dataset Collection
Topologies / Traffic Data Flows
Routing Protocols
OSPFv3MDR
NRL OLSR
...
Attacks
Spoofing (masquerading)
Selfish (no data-forwarding)
...
Network Description Refinement and Model Generation
Models are improved using an iterative process
Data analysis and model development
Impact Prediction Models

OUTGOING TRAFFIC DATA FLOWS BETWEEN NODES.

Node	TCP	UDP
1	10	2,3
2	1	3,4
3	2	4,5
4	3	5,6
5	4	6,7
6	5	7,8
7	6	8,9
8	7	9,10
9	8	10,1
10	9	1,2

```
If destIsSpoofed = false
| If srcIsSpoofed = true
| | If dataType = UDP
| | |   If passThrough = true then hijackable    t <- true
| | |   If passThrough = false then hijackable     <- false
| If srcIsSpoofed = false
| | If distanceTraveled = 1 then  hijackable     <- false
| | If distanceTraveled = 2 then  hijackable     <- false
| | If distanceTraveled = 3 then  hijackable     <- false
| | ...
```

Figure 1 The experimentation workflow.

```
for attackNode in 1…10
      for topology in "chain" "connected_grid" "cycle" "star" "tree" "two-centroid" "wheel"
        for protocol in "OLSR" "OSPFv3MDR"
          for attack in "forwarding" "spoofing"
            runScenario($attackNode, $topology, $protocol, $attack)
```

Figure 2 Pseudo code for scenario executions

approaches focus only on high-level survivability metrics such as the average number of critical links and surviving paths over multiple executions (Alsheikh, Lin, Niyato, & Tan, 2014; Wang, 2010.) The novel methodology developed by ARL also uses machine learning, but focuses on providing low-level details that are used to create attack graphs, assess risk, and identify mitigations.

MANET Attack Impact Prediction Models

The process for creating models to predict traffic hijacking consists of three steps: collecting a dataset, designing a network description scheme, and building the prediction models (or classifiers). These steps are used in the experimentation workflow depicted in Figure 1.

Collecting a Dataset

The common open research emulator (CORE) (Ahrenholz, Danilov, Henderson, & Kim, 2008) allows analysts to design and run network scenarios using a graphical interface. Topologies are generated by dragging and dropping icons into a workspace. Scenario variables such as node positions, protocols used, and custom processing (for logging data during execution) scripts are stored in a configuration file. A dataset was generated by running combinations of several variables. The pseudo code along with the variables used for this process is shown in Figure 2.

Attack node number. This indicates which of the 10 nodes in the scenario will issue an attack.

Topology. In attempts to achieve a suitable data distribution, rather than using randomly generated topologies, which are sometimes either too sparse or compressed, 10 nodes were used to populate 7 different static topologies. The topologies were generated by first manually positioning nodes to fulfill the desired connectivity. Next, the position values were hardcoded into the execution script (Figure 2). The topologies are labeled chain, connected grid, cycle, star, tree, two-centroid, and wheel. These can be seen at the top of Figure 1.

Eventually, this work will investigate whether survivability predictions extend to mobile scenarios. At first glance, it seems likely; when nodes move, the topologies change. It may be the case that survivability predictions can be made for each topology formed during the movement.

Routing protocol. This is the underlying network layer protocol that will be used to communicate data necessary for route maintenance. The dataset contains OLSR and OSPFv3MDR protocols. Both of these protocols are proactive meaning that they continually publish route information, as opposed to reactive protocols, which publish route information when requested. The OLSR implementation is provided by NRL and uses IPv4 addressing. OSPFv3MDR is part of the Quagga suite of protocols. OSPFv3MDR is a modification of OSPF that is optimized for mobile ad-hoc networks. OSPFv3MDR uses IPv6 addressing.

Attack. The types of attacks are spoofing and forwarding, as described in the previous section. For the automated process, the attack scripts take additional inputs, start time and duration.

The following parameters that were controlled across all emulation executions are described below.

Attack time. This parameter indicates how long a node must wait before executing the attack. This was a set to 60 seconds.

Scenario duration. Each scenario was separated into three phases—before, during, and after an attack was issued.

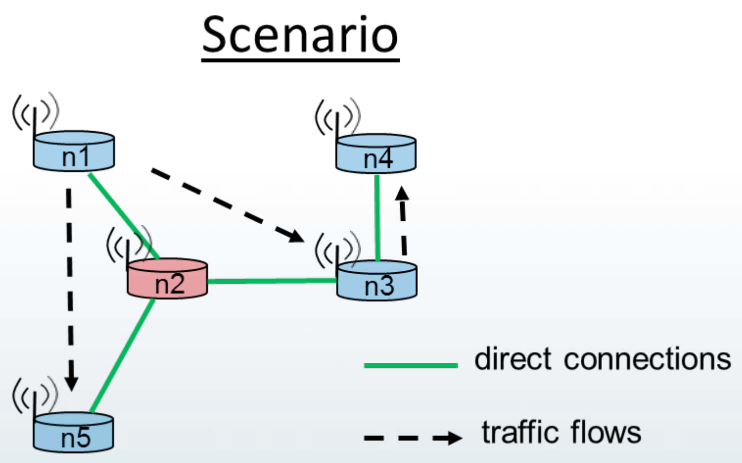
Data flow. During each scenario, nodes communicate using TCP and UDP data packets. Packets are 1280 bytes in size and are sent 50 times per second. The traffic was generated using mgen (Multi-Generator (MGEN), 2016). Each node opens six sockets, three outgoing and three incoming. Table 1 contains the data flows that were used during each instance.

Table 1. Traffic Data flows between nodes.

Node	TCP Outgoing	UDP Outgoing
1	10	2,3
2	1	3,4
3	2	4,5
4	3	5,6
5	4	6,7
6	5	7,8
7	6	8,9
8	7	9,10
9	8	10,1
10	9	1,2

Network Description Scheme

As the number of nodes in a network increases, so does the complexity of analyzing the impact of an attack on the network. Part of the reason for this lies in the fact that describing the live network (including nodes, traffic flows, routes, etc.) quickly leads to state-space explosion.



Description

	flow	attacker/ source distance	attacker/ destination distance	attacker is along this flow	...	Attacker successfully hijacks flow
flow description 1	n3→n4	1	2	no		no
flow description 2	n1→n3 **n1→n5**	1	1	yes		yes

Figure 3 Attacker-centric flow descriptions.

To avoid this, instead of describing the network as a collection of source and destination IP addresses, a set of parameters (based on attacker proximity) are used to describe traffic flows. Figure 3 shows an example where n1 is sending packets to n3 and n5 (denoted as n1/n3 and n1/n5); n3 is sending packets to n4 (n3/n4).

The network is defined as the collection of flow descriptions over an entire emulation instance. Figure 3 demonstrates how both flows n1/n5 and n1/n3 can be described with the same flow description (helping to avoid state space explosion). Flow descriptions are composed of the parameters in Table 2.

Table 2 Flow description parameters

#	Parameter	Description
1	fromHop	Hops from the attacker node to the source.
2	toHop	Hops from the attacker node to the destination.
3	dataType	Data packet type.
4	distanceTraveled	Hops from source to destination.
5	passThrough	Whether this flow hops through the attacker.
6	beforeStats	Packets statistics collected before an attack.
7	attackName	Spoofing or forwarding attack.
8	hijackable	Whether a flow is successfully hijacked during an attack.
9	srcIsSpoofed	Whether the source address is spoofed.
10	destIsSpoofed	Whether the destination address is spoofed.
11	hopsSpoofedToDest	Hops from the spoofed to the destination.
12	spoofedBetweenAttacker	Whether the spoofed is between the attacker and the destination.
13	spoofedBetweenAttackerGW	Whether the spoofed is a gateway (directly connected) node on the path to the destination.
14	destBetween SpoofedAndAttacker	Whether the destination is between the spoofed and the attacker.
15	destBetween SpoofedAndAttackerGW	Whether the destination is a gateway node on the path to the attacker.
16	attackerBetween SpoofedAndDest	Whether the destination is between the spoofed and the attacker.
17	attackerBetween SpoofedAndDestGW	Whether the destination is a gateway node on the path to the attacker.
18	srcBetweenSpoofedAndDest	Whether the destination is between the spoofed and the attacker.
19	srcBetween SpoofedAndDestGW	Whether the destination is a gateway node on the path to the attacker.
20	altPathWithoutAttacker	Whether an alternate path between source and destination exists without the attacker.

Defining parameters was an iterative process. Initially only parameters 1–8 were extracted from the dataset, but after a deeper analysis, it became clear that additional parameters were necessary (9–20) to improve network description accuracy. This deeper analysis consisted of the several steps. First, the data were captured and represented using the initial parameters. A python script generated a hash table (or dictionary) using all parameters, except *hijackable*, as keys in the key/value pair. The boolean parameter (taking on either true or false) *hijackable* was the value in the key/value pair. The python script went through each network description. If a collision was found and *hijackable* differed, these were considered conflicting flows.

For each conflicting flow, the number of times that the *hijackable* parameter resulted as true and false was stored. In the case where there was an equal amount of true and false counts, the emulation instances associated with the flows were run again. In the case where the counts were not equal, the emulation instances associated with the minority were run again. Sometimes the emulation instance encountered an unknown error and all links randomly disconnected.

More often, the reason for the conflicts resulted due to a lack of description of some network characteristic. Analysis of these cases led to the additional parameters. The network description was used to train a classifier to predict the *hijackable* parameter.

Building the Classifier

The dataset was formatted into the WEKA (Hall, et al., 2009) data-mining toolset file format and the REPTree algorithm was used to generate the classifiers. A subset of the parameters was used as training attributes to predict *hijackability* (parameter 8). Initially, parameters 1-7 were used (called the partial set) and then parameters 9-20 were added (called the all set).

As described earlier, the dataset used for evaluation consists of all combinations of the following configurations:

- Routing Protocols: OSPFv3MDR, OLSR
- Topologies: chain, connected_grid, cycle, star, tree, two-centroid, and wheel
- Attacks: forwarding, spoofing

Additionally, there are 10 nodes with 3 outgoing connections (2 UDP and 1 TCP). Each emulation instance contains one attacking node selected using a round-robin approach. In very few cases, a malfunction in CORE caused some nodes to stop capturing data and as a result, the dataset contains a small amount of noise.

Four REPTree classifiers were generated (OLSR forwarding, OLSR spoofing, OSPF forwarding, OSPF spoofing); performance was evaluated using 10 fold cross-validation. Table 3 and Table 4 contain the results for OLSR and OSPF respectively.

Table 3 Classification of hijackability with OLSR

Attack	Parameters Used	True Positive Rate	False Positive Rate	F-Measure
Forwarding	Partial	0.998	0.018	0.998
	All	0.998	0.018	0.998
Spoofing	Partial	0.975	0.161	0.975
	All	0.983	0.103	0.983

Table 4 Classification of hijackability with OSPFv3MDR

Attack	Parameters Used	True Positive Rate	False Positive Rate	F-Measure
Forwarding	Partial	1	0	1
	All	1	0	1
Spoofing	Partial	0.997	0.248	0.991
	All	0.998	0.031	0.998

The results show that the classifiers perform well when the *all* parameters are used. To assess risk associated with traffic hijacking, these models are used in conjunction with attack graph generation software.

Attack Graphs

Attack graphs enable system stakeholders to understand the stepping stones or exploitation procedures that an adversary could potentially execute to impact the confidentiality, integrity, and availability of a network system. These graphs are used to assess risk and to determine components that, when hardened, contribute most to risk reduction. Attack graphs work by reading a system topology and a vulnerability scan of the nodes in a network. Vulnerability database information is used to determine the outcome of exploitation. Until now, attack graphs were incapable of representing traffic hijacking attacks success or failure of these attacks depends on specific implementations of routing protocols and on the operating environment. Through a collaborative effort, ARL has partnered with the University of Texas at El Paso (UTEP), Abilene Christian University (ACU), and University of South Florida (USF) to tackle this issue by leveraging impact prediction models and state of the art attack graph software.

Platform

One of our goals for this work was to augment an existing tool in order to leverage preexisting risk analysis capabilities. MulVAL is used as the platform because it is open source, scales well (O(n2)) with the size of the network, and the underlying reasoning engine is extendable by allowing users to introduce custom interaction rules.

MulVal reads a file with custom interaction rules and a file describing the scenario. The custom interaction rules enable extensible scenario specification. The scenario file models the network using primitives (unit clauses) to introduce facts, e.g., specify available services, known vulnerabilities, user accounts, access on a system, and identifying nodes as web servers, network file servers, etc. The standard and custom interaction rule sets specify how knowledge is derived from known facts.

Figure 4 shows a simple interaction rule that states that if a principal's account on Host is compromised (e.g., through stolen credentials) and Host is accessible (e.g., through the network and a listening login service), an attacker may execute code on Host with the permission levels of the principal's account. MulVAL generates the attack graph showing the primitive and derived facts leading to the attack goal.

```
execCode(Host, Perm) :-
  principalCompromised(Victim),
  hasAccount(Victim, Host, Perm),
  canAccessHost(Host).
```

Figure 4 An execCode interaction rule

There were three major steps to generate attack graphs for network layer vulnerabilities: (1) develop MulVAL custom interaction rules to represent route hijacking, (2) incorporate the elements (e.g., node routes, attacker proximity, routing protocol used, traffic type) that are required to determine the success or failure of hijacking attacks, and (3) improve usability by automating the attack graph generation process.

The scenario in Figure 5 was the basis for the MulVAL rule development step. In this scenario, all nodes are gateways using OLSR for dynamic routing and the attacker is conducting an IP address spoofing attack (masquerading as the ftp server node) in order to hijack traffic from the ftp client.

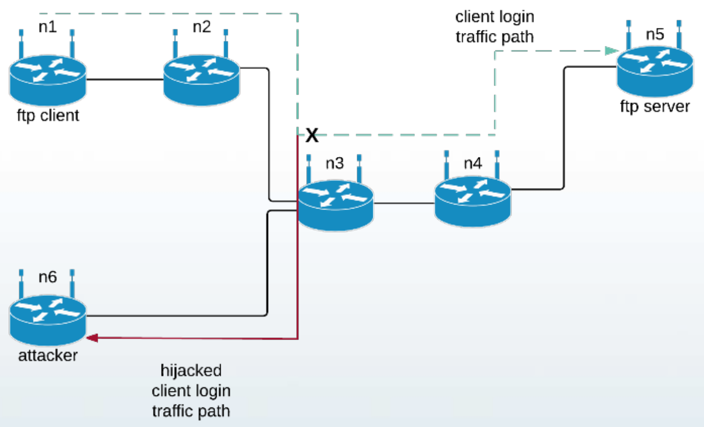


Figure 5 Route Hijack Scenario

Custom Rules for Route Hijacking

One of the existing rules, principalCompromised, was overloaded to account for weak or null authentication in dynamic routing protocols (see Figure 6). This rule can be read as follows: the principal is compromised if it has an account, an active connection to a login service on a node, and OLSR is misconfigured. It is important to note that route hijacking does not require the attacker to be located in the victim's subnet.

The impact prediction models are used to determine if a flow is susceptible to hijacking. This information is then used to generate the MulVAL attack graph input file that uses the custom rules to represent the hijacking behavior. Alternatively, instead of using the impact prediction models, a user may indicate hijacking susceptibility using different sources, e.g., theoretical models such as (Santiraveewan & and Permpoontanalarp, 2004).

```
principalCompromised(Victim) :-
/* The victim has a user account on the remote host */
hasAccount(Victim, RemoteHost, User),
/* nrlolsr is being used */
networkServiceInfo(H, nrlolsr, olsr, _no_port, _user),
/* nrlolsr is misconfigured allowing traffic hijacking */
vulExists(H, nrlolsrVul, nrlolsr, remoteExploit, nrlolsrHijack),
/* The User has an account on a login service on the remote host */
logInService(RemoteHost, Protocol, Port),
/* There is an active connection from the host to the remote machine */
flowExists(H, RemoteHost, Protocol, Port, User).
```

Figure 6 Custom Rule for Null or Weak Authentication in OLSR

Data Pipeline for Generating Attack Graphs

To facilitate the development of these attack graphs, an automated and modular data pipeline was implemented (see Figure 7). This pipeline only requires that a user enter network scenario data at the start of the pipeline, although data can be modified or substituted at any point in between.

All of the components in the pipeline read and write data in XML format. The Attributes Extractor reads information related to a network scenario including attack, attacker position, routing protocol, traffic data, and route tables (which can be manually entered or extracted from emulation, simulation, or field data). For this scenario, data were obtained using CORE to configure the scenario. During scenario execution, and after routes converged, the required data were extracted and copied from each node. These data are used to generate attributes (attribute

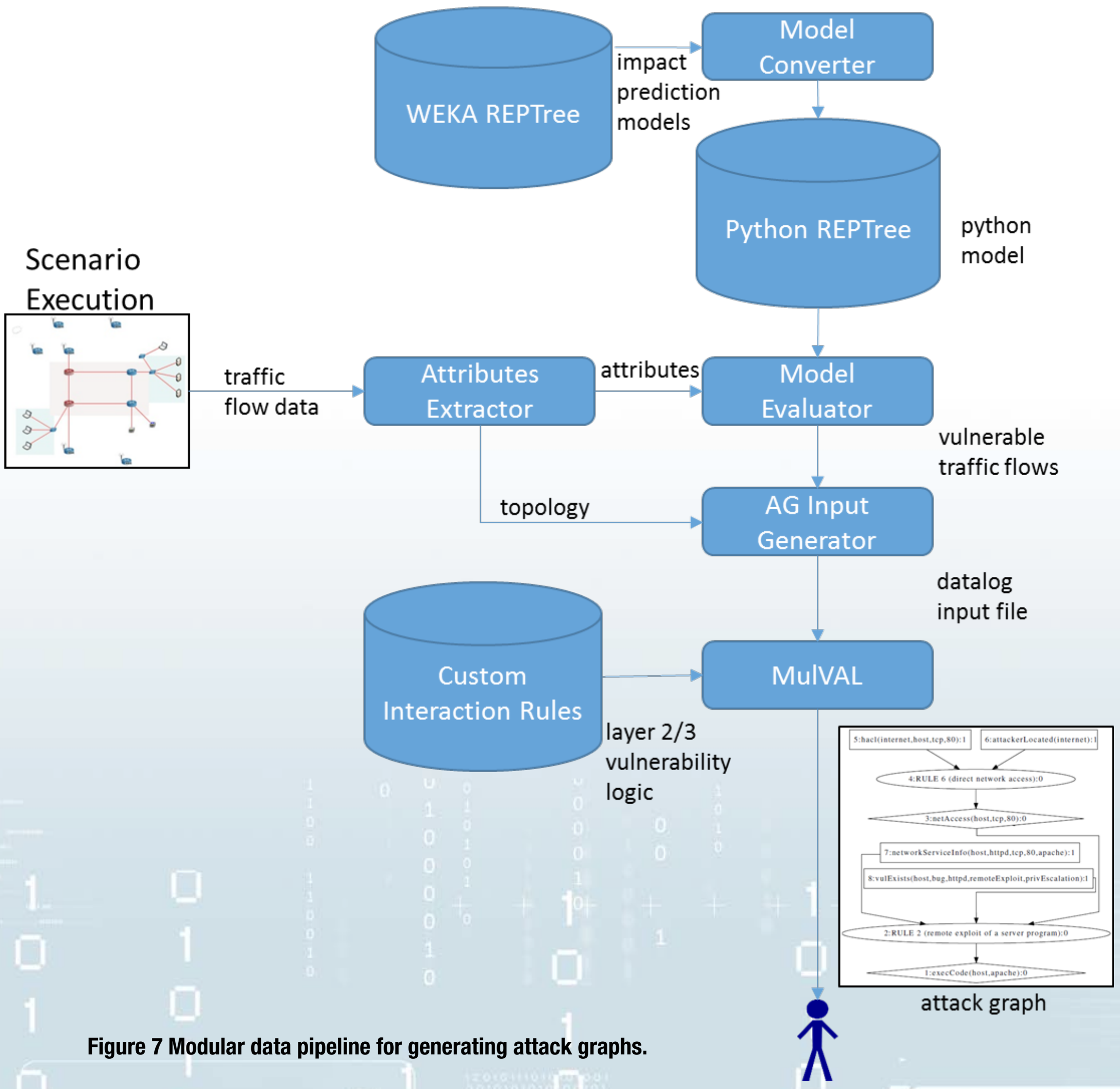

WEKA REPTree
impact prediction models
Model Converter
Python REPTree
python model
Scenario Execution
traffic flow data
Attributes Extractor
attributes
Model Evaluator
vulnerable traffic flows
topology
AG Input Generator
datalog input file
Custom Interaction Rules
MulVAL
layer 2/3 vulnerability logic
5:hacl(internet,host,tcp,80):1
6:attackerLocated(internet):1
4:RULE 6 (direct network access):0
3:netAccess(host,tcp,80):0
7:networkServiceInfo(host,httpd,tcp,80,apache):1
8:vulExists(host,bug,httpd,remoteExploit,privEscalation):1
2:RULE 2 (remote exploit of a server program):0
1:execCode(host,apache):0
attack graph

Figure 7 Modular data pipeline for generating attack graphs.

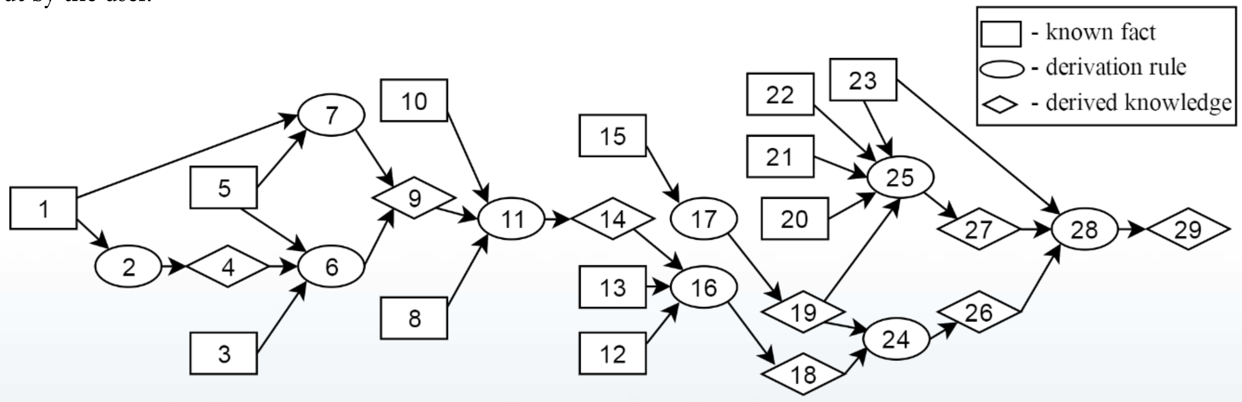
details are described in the Network Description Scheme section of this article). The Model Evaluator uses the attributes to query the WEKA decision tree rules (that are converted into python scripts by the Model Converter) in order to determine which flows are vulnerable (i.e., can be hijacked).

For each vulnerable flow identified, the AG Input Generator will append a vulExists entry (associated with OLSR) to the MulVAL input file. The AG Input Generator also uses the data from the Attributes Extractor to encode network scenario information in the MulVAL file.

Figure 8 shows a sample MulVAL input file generated in this way. The topology information (lines 2–13) is calculated by the Attributes Extractor using the route information from the CORE execution (in our case). The network scenario configuration (lines 17–26) is provided as input by the user. Line 29 is generated using the results of the Model Evaluator.

```
1  /* Network Topology Definitions:*/
2  hacl(ftpClientHost, n2, _proto, _port).
3  hacl(n2, n3, _proto, _port).
4  hacl(n3, n4, _proto, _port).
5  hacl(n4, ftpServerHost, _proto, _port).
6  hacl(n6, n3, _proto, _port).
7
8  gateway(ftpClientHost).
9  gateway(n2).
10 gateway(n3).
11 gateway(n4).
12 gateway(ftpServerHost).
13 gateway(n6).
14
15 /* Flow Definitions: */
16 /* Flow #1 :*/
17 flowExists(ftpClientHost, ftpServerHost,'TCP', 21, flow1Account).
18 hasAccount(flow1Principal, ftpServerHost, flow1Account).
19 networkServiceInfo(ftpClientHost, nrlolsr, olsr, NA_port_layer3, _NA_perm_layer3).
20
21 /* Attack Configuration */
22 attackerLocated(n6).
23 attackGoal(execCode(ftpServerHost, _)).
24
25 /* A cleartext loginService exists on the remote machine */
26 networkServiceInfo(ftpServerHost, ftpd, 'TCP', 21, flow1Account).
27
28 /* Vulnerable Flow */
29 vulExists(ftpClientHost, nrlolsrVul, nrlolsr, remoteExploit, nrlolsrHijack.
```

Figure 8 MulVAL input file for Route Hijack Scenario



1: attackerLocated(n6)
2: direct on-host access
3: gateway(n6)
4: netAccess(n6, tcp, 21)
5: hacl(n6, n3, tcp, 21)
6: multi-hop access by gateway
7: direct network access
8: hacl(n3, n4, tcp, 21)
9: netAccess(n4, tcp, 21)
10: gateway(n3)
11: multi-hop access by gatway
12: gateway(n4)
13: hacl(n4, ftpServerHost, tcp, 21)
14: netAccess(n4, tcp, 21)
15: networkServiceInfo(ftpServerHost, ftpd, tcp, 21, flow1Account)
16: multi-hop access by gateway
17: log in for ftpd
18: netAccess(ftpServerHost, tcp, 21)
19: logInService(ftpServerHost, tcp, 21)
20: flowExists(ftpClientHost, ftpServerHost, tcp, 21, flow1Account)
21: vulExists(ftpClientHost, nrlolsrVul, nrlolsr, remoteExploit, nrlolsrHijack)
22: networkServiceInfo(ftpClientHost, nrlolsr, olsr,_,_)
23: hasAccount(flow1Principal, ftpServerHost, flow1Account)
24: access a host through a log-in service
25: password sniffing through route hijack
26: canAccessHost(ftpServerHost)
27: principalCompromised(flow1Principal)
28: when a principal is compromised any machine he has an account on is compromised
29: execCode(ftpServerHost,flow1Account)

Figure 9 Attack graph for route hijacking scenario

Executing the pipeline with the route hijacking input file produces the attack graph shown in Figure 9. In this figure, rectangles are facts specified in the MulVAL input file; ovals are the rules (either the built-in or custom) that applied to derive new knowledge (diamonds). This graph shows that an attacker can successfully hijack traffic from the ftp client and use the sniffed credentials to execute code on the ftp server.

Future Work

In the short term, ARL is working on generating models for other routing protocols, such as BGP and RIP that exist in the wired domain. Other work is looking at the effects of multiple (possibly collaborating) attackers, and using metrics from the models, such as confidence, to prioritize and prune attack paths. Longer term efforts will focus on using this work to build decision support systems and intelligent agents for improved assessment time, coverage, and accuracy.

REFERENCES

[1] Ahrenholz, J., Danilov, C., Henderson, T. R., & Kim, J. H. (2008). CORE: A real-time network emulator. *IEEE Military Communications Conference* (pp. 1-7). IEEE.

[2] Alsheikh, M. A., Lin, S., Niyato, D., & Tan, H.-P. (2014). Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 1996-2018.

[3] Andel, T. R., & Yasinsac, A. (2006). On the credibility of manet simulations. *Computer*, 48-54.

[4] Andel, T. R., & Yasinsac, A. (2007). Surveying security analysis techniques in MANET routing protocols. I*EEE Communications Surveys & Tutorials*, 70-84.

[5] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*.

[6] Multi-Generator (MGEN). (2016). Retrieved from http://www.nrl.navy.mil/itd/ncs/products/mgen (Accessed July 29, 2016)

[7] Pathan, Al-Sakib Khan. (2016). *Security of self-organizing networks: MANET, WSN, WMN, VANET*. CRC Press.

[8] Santiraveewan, V., & and Permpoontanalarp, Y. (2004). A graph-based methodology for analyzing ip spoofing attack. *Advanced Information Networking and Applications* (pp. 227-230). IEEE.

[9] Wang, T. a. (2010). A neural network model for evaluating mobile ad hoc wireless network survivability. *International Symposium on Neural Networks*.

ABOUT THE AUTHORS

Jaime Acosta earned his Ph.D. degree from the University of Texas at El Paso. He now works at the U.S. Army Research Laboratory. His research interests include execution-based model generation, network security, and cybersecurity assessment methodologies.

Edgar Padilla is a Ph.D. student at the University of Texas at El Paso. He is also a systems programmer for the University's enterprise resource planning systems. His research interests include risk analysis and secure software architecture.

John Homer earned his PhD from Kansas State University. He now teaches at Abilene Christian University and his current research focuses on developing dependable and verifiable approaches to risk identification and assessment.

Xinming Ou is an associate professor at the University of South Florida. He earned his Ph.D. from Princeton University in 2005. His research interest are in cyber defense technologies, cyber physical system security, human factors in security, and mobile system security.

Open Invitations

Journals/Webinars/SME Network

CSIAC offers free webinars on a regular basis with experts in the technical subject areas of Cybersecurity, Software Engineering, Modeling & Simulation, and Knowledge Management/ Information Sharing.

CSIAC also maintains a network of Subject Matter Experts (SMEs) in their four primary domains of technical responsibility. The SMEs participate in a variety of activities they are interested in and provide support to the CSIAC user community.

CSIAC maintains open invitations for guest speakers and authors as contributors to their webinars and journals. CSIAC is also interested in identifying more qualified candidates to join their existing SME network. If you're interested, please visit the CSIAC website (CSIAC.org) or send an email to info@csiac.org for more information.

CSIAC
Cyber Security & Information Systems Information Analysis Center

DoDIACs
Department of Defense Information Analysis Centers

https://www.csiac.org/

SECURITY OF CYBER-PHYSICAL SYSTEMS

BY: Edward J. M. Colbert, Ph.D.
Army Research Laboratory (ARL), Adelphi, MD

ABSTRACT: *Cyber Physical Systems (CPSs) are electronic control systems that control physical machines such as motors and valves in an industrial plant. In a networked environment, the security of the physical machines depends on the security of the electronic control systems, but cybersecurity is not typically the main design concern. The main concern for CPSs is the availability of the physical machines governing operations. As CPS owners continue to install remote network control devices and incorporate an increasing number of insecure Internet-of-Things (IoT) devices in their industrial processes, the underlying security of their operations becomes increasingly vulnerable. This article outlines current cybersecurity issues of CPSs and potential concerns for future CPS designers and operators. Secure future CPSs are necessary for keeping our critical infrastructure safe.*

Introduction

The term Cyber-Physical System (CPS) is a generic term for a variety of other control systems, such as SCADA (Supervisory Control and Data Acquisition) systems, ICSs (Industrial Control Systems), BCSs (Building Control Systems), and the global electrical smart grid. These control systems are comprised of computers, electrical and mechanical devices, and manual processes overseen by humans. CPSs perform automated or partially automated control of physical equipment in manufacturing and chemical plants, electric utilities, distribution and transportation systems and many other industries. CPSs integrate computational resources, communication capabilities, sensing, and actuation in effort to monitor and control physical processes. CPSs are found in critical infrastructure such as transportation networks, Unmanned Aerial Vehicles (UAVs), nuclear power generation, electric power distribution networks, water and gas distribution networks, and advanced communication systems.

A key difference between CPSs and traditional Information Technology (IT) systems is that CPSs interact strongly with the physical environment, and the availability of the physical devices is the most important security aspect. However, CPSs are also cyber systems and are therefore vulnerable to cyber-attacks. This connection with the physical world, however, presents unique challenges and opportunities.

In traditional critical infrastructure systems, great efforts are expended to address concerns about safety and reliability, and to develop the appropriate techniques for fault detection, isolation, and recovery. In CPSs, however, the additional cyber element introduces specific vulnerabilities which are not directly addressed in traditional fault tolerance and reliable computing practices. Addressing the cyber element in the safety and reliability of CPSs is of utmost importance, since the introduction of highly integrated CPSs into critical infrastructures and emerging systems could lead to situations where cyber based attacks against CPSs could adversely affect widespread public safety (e.g. Cardenas, Amin & Sastry 2008).

CPSs monitor and control industrial processes across a myriad of industries and critical infrastructures on a global scale

CPSs monitor and control industrial processes across a myriad of industries and critical infrastructures on a global scale (Weiss 2010) and therefore must be protected. Besides controlling critical infrastructure such as transportation and energy production, CPSs are increasingly used by consumers and therefore influence our everyday personal lives. Current and future CPSs are becoming widespread in our homes, automobiles and on our person, and will eventually be a large part of the "Internet-of-Things" in which an extensive array of physical devices will be heavily interconnected.

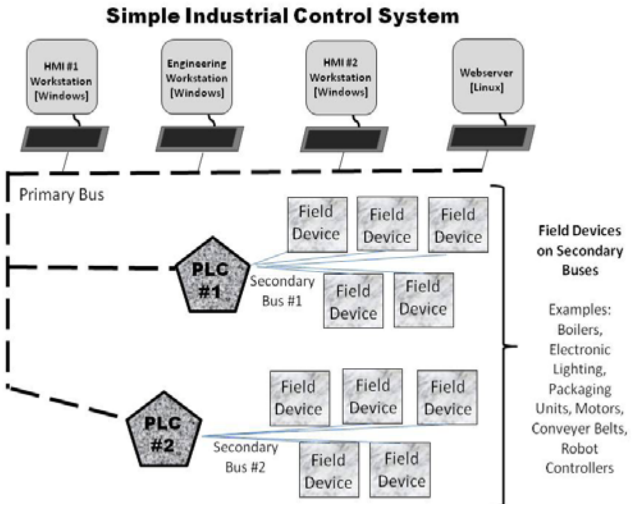

Simple Industrial Control System
HMI #1 Workstation [Windows]
Engineering Workstation [Windows]
HMI #2 Workstation [Windows]
Webserver [Linux]
Primary Bus
PLC #1
Secondary Bus #1
PLC #2
Secondary Bus #2
Field Device
Field Devices on Secondary Buses
Examples: Boilers, Electronic Lighting, Packaging Units, Motors, Conveyer Belts, Robot Controllers

Figure 1: Sample of a Simple CPS

In Figure 3 we show a rough sketch of a simple Industrial Control System (ICS), which is a CPS used in an industrial setting. This control system has two Programmable Logic Controllers (PLCs), each of which are connected (upper panel) to a standard IT device network with a few Workstations. The workstations typically run Microsoft Windows or Linux, as in a standard Enterprise network. In the diagram, this "cyber" network is annotated as "Primary Bus." The traffic on this network is usually IP packet-based.

Downward from the PLCs are Secondary Buses that control field devices, such as boilers, electronic lighting, and packaging units. While these buses or networks may be IP packet-based, they are usually serial links or simple hard-wired cables with specialized voltage or current control needed to run the field devices. In other words, they are not meant to have a standard network communication protocol such as TCP/IP. This is the "physical" component to the CPS.

Also, notice that most of the equipment in the ICS is NOT computer servers, network switches, or routers, such as you might find in an IT network. Even the workstations connected to the Primary Bus (the cyber component) are doing atypical work. They are not meant to be connected to the Internet to browse the web. They are specifically configured to only perform their function in the CPS. There is often little interest in following security measures such as installing anti-virus or keeping the operating system up to date because, ideally, the systems are not supposed to be accessed from the outside, and are not supposed to access the outside. The field devices and PLCs in the Secondary Buses do not run standard operating systems or security protection applications, and most likely will never be modified to do so.

The scope of a CPS may vary enormously. It can range from a single PLC controlling a motor to larger distributed system controlling many devices in a power utility generation plant, for example. CPS configurations also differ greatly. Configurations may range from a single component to a highly distributed configuration with wide area networks spanning a whole continent with many thousands of physical devices.

In spite of such diversity, the basic building blocks of a CPS can be assigned to only a few classes. These include for example Programmable Logic Controllers, Remote Terminal Units, and Communication Gateways. A CPS may be completely automated, but normally is controlled or at least supervised by a human operator. Therefore, human machine interfaces (HMIs) are important components of a CPS.

CPSs are traditionally operational systems, where process control is priority for the human operators (see Hahn 2016). Of the four components of security (Confidentiality, Integrity, Availability, and Safety), Availability and Safety dominate security concerns for CPS. Typically, Confidentiality and Integrity have high priority on IT networks. Whereas IT systems have similar standard computer hardware and network infrastructure, human usage policies, performance requirements, and security defense methods, CPSs are diverse. Their hardware, policies, and process requirements are typically unique to the system, so that a unique security solution for all CPSs is extremely difficult to develop. As IT system technologies begin to converge into CPSs, it becomes more critical to understand and analyze these differences in order to manage expectations of future CPS security. This is especially important if IT security methods are considered for defending CPSs from attack.

Stuxnet was a sophisticated attack with many facets - it is not merely a piece of cyber malware

While strong concerns about security of CPSs, particularly in the context of critical national infrastructure, were expressed even in the early 2000s (US Department of Energy 2002; Luders 2005), it was not until the legendary 2010 Stuxnet episode (e.g., Langner 2011) that security of industrial control systems entered public and government discourse and acquired today's saliency (Executive Order 2013; Stouffer et al 2015).

A recent overview of cyber-security issues in CPSs, specifically Industrial Control Systems, can be found in Colbert & Kott (2016). Some pertinent aspects from that work are repeated here.

Attacks and Threats for Cyber-Physical Systems

Perhaps the most well publicized control system attack is Stuxnet, which was an attack on the uranium enrichment plant in Natanz, Iran. The Stuxnet malware was discovered by security engineer Sergey Ulasen in 2010 (see, e.g., Zetter 2015). Stuxnet was a sophisticated attack with many facets – it is not merely a piece of cyber malware. The Stuxnet took advantage of both the physical nature of the Natanz control system and vulnerable security flaws in the unique cyber components used in the CPS.

As mentioned, CPSs were traditionally designed around availability and safety. Cyber security features were not part of the original design of CPSs (Luiijf 2016) because:

- CPSs were based on specialized hardware, proprietary code and protocol standards. Only specialists knew about how to use them.
- CPSs operated in a closed environment without any connectivity with other domains. Physical security methods were adequate.
- Since CPSs operated in a closed and assumed benign environment, there was no reason for creating secure and robust CPS protocols.

Current threats to CPSs remain numerous and broad. These threats have enabled complex and specific attacks to be executed (see Sullivan 2015 for a current summary). The nature and efficacy of these attacks are largely determined by a complex mix of security deficiencies in CPS systems that aggregate architectures and approaches from several epochs of technological history. For example, SCADA systems of the second generation were distributed, but used non-standard protocols. This enabled centralized supervisory servers and remote PLCs and RTUs. Security was often overlooked in this generation. The third generation of SCADA systems used common network protocols such as TCP/IP. This generation added the concept of Process Control Network (PCN), which allowed SCADA enclaves to connect to the Internet at large. This connection enabled operators to remotely manage the SCADA ecosystem and introduced malware to the enclaves.

Security by design was lacking and designers too often CPS designers on "security by obscurity" – relying on hopes that the attacker would lack knowledge about the inner structure and workings of the system. Elements of common attacks (see Evancich & Li 2016 for details) include malware that used buffer overflow, code injection, and rootkits. CPS attacks were very sophisticated and were commonly believed to require extensive development efforts and resources of a group sponsored by a nation-state. For example, rootkit-based attacks that hide malicious processes from detection by users requires significant development and testing.

Intrusion Detection for Cyber-Physical Systems

Even if the known threats, risk factors and other security metrics are well understood and effectively mitigated, a determined adversary will have non-negligible probability of successful penetration or intrusion of a CPS. Here we use the term "intrusion detection" to refer to a broad range of processes and effects associated with the presence and actions of malicious software and actions against a CPS. Once an intrusion has occurred, the first and necessary step for defeat and remediation is to detect the existence of the intrusion.

In Colbert & Hutchinson (2016), we describe the history of intrusion detection in IT and CPS systems and discuss various methods and Intrusion Detection Systems (IDS). These authors discuss the difficult question of whether insights and approaches intended for information and communications technology (IT) systems can be adapted for CPSs. To answer this question, they explore modern intrusion detection techniques in IT such as host-based techniques and network-based techniques, and the differences and relative advantages of signature-based and non-signature methods.

After approximately 2010, CPS intrusion detection techniques began to focus on knowledge about the processes controlled by the CPSs rather than on direct detection or inference of the malware on the network. The design intent of a CPS is intended to (1) establish appropriate process values to produce desired output and (2) to allow operators to observe aspects of the plant to assure proper operation and safety and quality conditions. The sole purpose and only capability of CPS network traffic control messages is to support the synchronization of the PLC registers and to provide a local, HMI-side copy of these registers, to effect control of the plant processes. IT network traffic has a much wider variety of uses, but is not generally used for process control. While both CPS and IT computers have registers, only a CPS network can change and read register values. Register values directly affect process parameters and hence, the process. Since CPS security is ultimately for safeguarding the process variables and not the network traffic itself, process-oriented designs for monitoring and intrusion detection became of interest.

For example, Hadziosmanovic et al. (2013) attempted to model process variable excursions beyond their appropriate ranges using machine-learning techniques. These authors describe a novel network monitoring approach that utilizes process semantics by (1) extracting the value of process variables from network traffic, (2) characterizing types of variables based on the behavior of time series, and (3) modeling and monitoring the regularity of variable values over time. Approximately 98% of the process control variables used in real-world plans are reliably monitored by their process-oriented method. The remaining 2% of the variables remain challenging to model with this approach. This novel approach demonstrates that process variables can successfully be modeled for ID. However, as they mention, additional work is needed if all of the process variables are to be monitored reliably. Semantic modeling of plant control variables in the control system process became a favorable and presumably effective intrusion detection method for CPS.

Semantic Security Monitoring (SSM) by Hadziomanovic et al. (2013) used analysis of control-bus traffic messages to construct a 3rd copy of the plant-PLC registers for a new purpose: to detect events that suggest that plant operations may be out of specification, out of compliance, or out of a desired safety range. An important caveat of using network data to construct a security model is that the network control messages were never intended for security monitoring. The rates and precision of the information in the control messages are designed to be sufficient to accomplish control to maintain quality output, but they may not be appropriate or sufficient for security and safety monitoring operations.

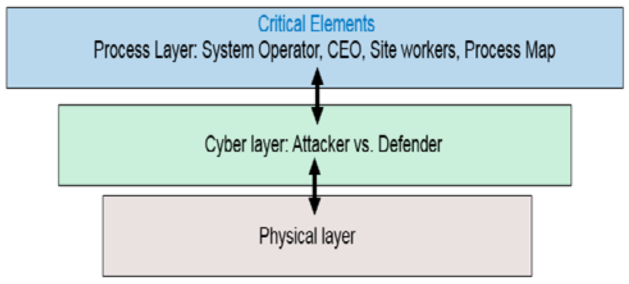

Critical Elements
Process Layer: System Operator, CEO, Site workers, Process Map
Cyber layer: Attacker vs. Defender
Physical layer

Figure 2: Three layers of a Cyber Physical System

A second method of semantic modeling, developed at the US Army Research Laboratory, was proposed by Colbert et. al (2016). This method requires plant personnel input to define critical process variable limits instead of inferred input to the security model from network control traffic.

One can view the CPS as a three-layer system to better understand our process-oriented intrusion detection method. As mentioned, CPSs inherently have physical and cyber layers, in which physical machinery and attackers and defenders operate, respectively (see Figure 2). In our model, intrusion detection occurs on a third layer (the "process" layer), in which the system operator and system owner operate. A process diagram, plant policies and procedures, and continuous system monitoring by the system operator determine the critical elements and requirements needed to keep the system operational.

Our CPS intrusion detection research at the Army Research Laboratory (ARL) is based on the assumption that all of the process variables do not need to be monitored for alerting. Rather, there are critical process variables (or, more generically, critical elements of the process) that need to be monitored for alerting. Abnormal values of the remaining variables are not significant enough to harm the underlying plant process. We argue that identifying the critical values and determining the allowed ranges of those critical values is extremely difficult if only network traffic data is used. We use a collaborative approach to constructing the security model which requires plant operator or plant SME input and potentially out-of-band (OOB) sensor data in addition to data from network packets.

Our model recognizes that, just as in IT intrusion detection, reference information from plant sensors, configurations, semantics, and policies (acceptable security/safety value ranges) must be captured, maintained, shared, and made available to the security/safety monitoring analysts in timely, orderly, and priority-relevant means to enhance decision-making. However, it also recognizes that CPS process sampling methods and process control methods (e.g. MODBUS) were never intended to feed security/safety analyses. Thus, as stated earlier, many process parameters seen in network traffic may not be relevant, or may not be sampled at sufficient rate or fidelity. Moreover, there may be other process variables that are indeed critical, but they are not represented in network traffic, i.e. they are out of band. In this case, independent sensing of these parameters would be needed to create sufficient uplift in timeliness, accuracy, and relevance to the security/safety monitoring mission. In the ARL model, the SME defines the critical security model variables based on his knowledge and analysis of the plant processes, and the IDS security engineer implements the appropriate security model. We refer to this model as "collaborative" since the security engineer utilizes human input from the plant operator/SME input for constructing the IDS security model.

Our ARL intrusion detection development platform (e.g. see Long 2004) defines 'alerting' as automatic information generation to be sent to a human analyst for further consideration. The analyst then examines the alerts and other relevant information and determines when to send an 'alarm' to the system owner. An alarm is a notice of a possible compromise or other insecure situation, as determined by the human analyst, whereas an alert is automatically generated information from a sensor or algorithm.

Our collaborative intrusion detection model was implemented in the ARL intrusion detection development platform in a live testbed at ARL. General findings from our testbed experiments are described in Sullivan & Colbert (2016) and Sullivan, Colbert & Kott (2016). In Figure 3, we show the implemented IDS architecture in our testbed. A network tap (e.g. SPAN port on a switch) provides network capture data to one or more sensor nodes. Some of the data are pre-processed on the sensor nodes into 'detects' (detect/alert information) and index data. The Ingest node then forwards that data to a master node, which stores raw data and provides indexed information for analyst web tools. More complicated analytics are executed by the Analysis Node, which again places results back on the Master Node for the web interface to display. The Web Interface contains an HTTP web server with web analytics and web links for execution of additional analysis tools. The Human Analyst then examines alerting information that resides in the system using various analytical tools.

In our testbed implementation, IDS alerting by the Sensor Node is generated from anomalies on the process layer by monitoring critical process values. As mentioned, critical process variables are those that have been collaboratively defined to signify whether the control system is successfully operational or not. Sensor nodes are modified specifically to monitor the value of all critical process variables. For example, nominal values, and upper and lower limits for critical values, and criticality of the alert are programmed into

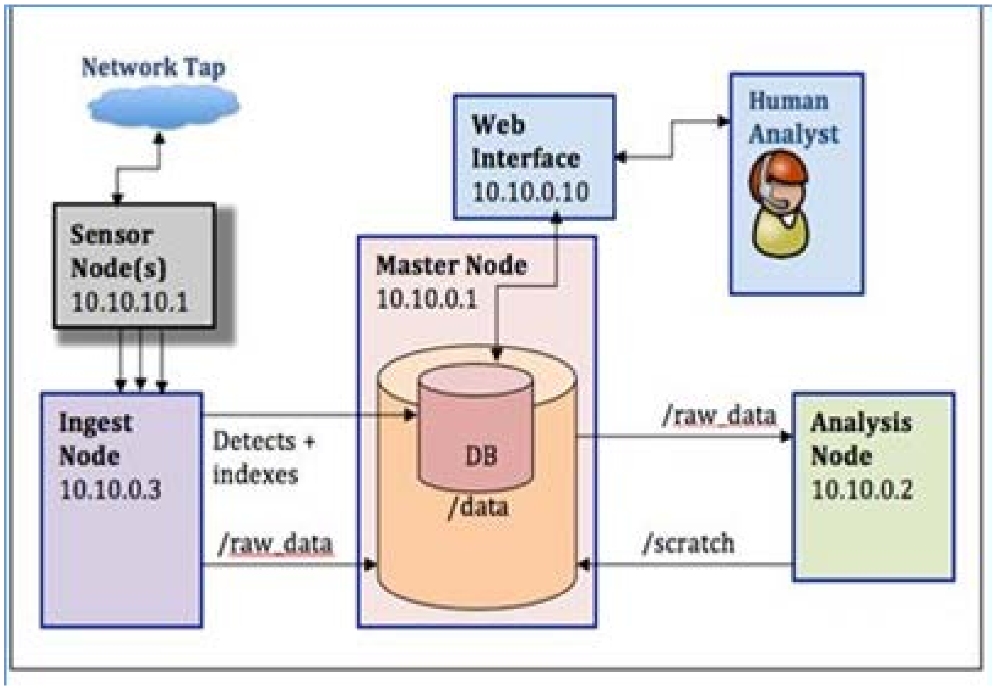


Figure 3: Generic Intrusion Detection Architecture

the sensor node. This process-oriented intrusion detection method is meant to be used in parallel with anomaly-based and signature-based intrusion detection methods that are available for CPSs (see Colbert & Hutchinson 2016).

Security Models for Cyber-Physical Systems

There are five well known classes of methods for cyber risk assessment and management for CPSs (Henry et al. 2016). An Expert Elicited Model method involves computational models to assess risk based on expert elicited identification and characterization of cyber system attributes such as network data flows and the estimation of the susceptibility of those resources and data flows to different types of compromise. This approach possesses significant appeal for many applications, including cases involving complicated networks for which little design information is readily available and cases in which a relatively quick analysis is needed. One major drawback of this approach is lack of completeness. Second, the Attack Graph method advocates construction of attack trees or graphs, either by hand or through automated interrogation of a system of interest. This approach has many advantages. Principal among these is a very light data requirement. Models in this class do not suffer precision or fidelity shortcomings because they are constructed directly from system data without abstraction or aggregation. Another advantage of this approach is flexibility. Third, game theoretic models explicitly account for the interaction of attackers and defenders in a game theoretic framework. Models in this class are much more varied, and the approach is much less mature than the expert elicited and graph-based approaches described previously. Games can inform how the playing field can be better tilted in favor of the defender by adopting architectural changes and new access control policies. The fourth method is Petri Net models, which are favored by the authors of this chapter. This chapter's Petri net approach is derived from the Attack Graph school of thought. A Petri net is a directed bipartite graph, in which a cyber attack is modeled as the successive exploitation of vulnerabilities on hosts to escalate and then exploit privileges on the network. The final method described involves stochastic games overlaid on Petri nets, creating a much more powerful, and more challenging, approach. In this model, transitions based on attacks corresponding to network defense measures replace exploit-specific transitions.

Security models can be useful for estimating risk and other security metrics. Metrics are defined as measurable properties of a system that quantify the degree to which objectives of the system are achieved. Metrics can provide cyber defenders of a CPS with critical insights regarding the system. Metrics are generally acquired by analyzing relevant attributes of that system.

In terms of cyber security metrics, CPSs tend to have unique features: in many cases, these systems are older technologies that were designed for functionality rather than security. They are also extremely diverse systems that have different requirements and objectives. Therefore, metrics for CPSs must be tailored to a diverse group of systems with many features and perform many different functions.

As described in the previous section, in our ARL CPS research, we visualize three layers for the control system: physical, cyber, and process. See Figure 2, for example, which we used to describe our intrusion detection methodology. We use the same 3-layer model to construct a security model for CPSs. Our current approach is to use game-theoretical methods similar to Zhu & Basar (2015), who have developed elegant game-theoretic methods for the physical and cyber layers (a 2-layer model).

As the number of devices in IoT and control system increases, software and hardware vulnerabilities will also increase"

Game theory has been successfully used for security models for cyber systems. A simplistic cyber encounter between an attacker and a defender (security engineer) can be described by a zero-sum game between two players who both have complete information about the cyber system and their opponent. The rational moves of the two players are well-defined by saddle-points (Nash equilibria points) once the costs and awards are defined over all game strategies. There are some important differences between CPS attack scenarios and a simplistic security game.

As mentioned, CPSs are not merely cyber networks. They are connected to physical systems and are affected by the physical systems. Attacks focused on the physical system can penetrate into the cyber network. In addition, the operator of the control process is an important player to consider. He or she dutifully monitors critical elements of the process and makes optimal choices to maintain system operability given policy constraints dictated by the system owner. There are clearly more players and more systems to consider than the attacker and defender in the simplistic game. Our three-game model in which defender and attacker play in the cyber regime, physical control devices and perturbations (intentional or accidental) play in the physical regime, and operator and system owner play in an abstracted process regime. All three regimes and all players can affect each other in this complex game.

Next, one nominally assumes that if all of the information in the game is readily available to the players, the players will choose the optimal path so that they suffer the least cost. If cost is a monetary measure, this may not be true, especially for state-sponsored attackers. Their defensive opponent however may indeed act to minimize monetary costs. Even if cost measures were completely known of all players, players are inefficient and often not rational. They can be coerced by psychological affects or swayed by political demands of their peers and supervisors. For extended attacks, multiple humans may play the part of a single actor. Human behavior can be modelled in some circumstances so that these uncertainties can be taken into account.

Lastly, assuming costs and behavior can be modelled well, the attacker will often not have complete knowledge of the three regimes when they begin their attack. They may have done some reconnaissance work, but will be missing important pieces of information. This lack of information will affect their instantaneous strategy, and their path taken through attack space may be highly dependent on the amount of information available. Incomplete-information game models using Bayesian methods can be used to accommodate this effect.

After the analytical model for the 3-layer games is developed, we can use linear methods to solve models for internal variables of the game-theoretical model. As expected, the plant operator or plant SME will need to collaborate with the security engineer to configure the game parameters, but internal security model parameter can be used to construct security metrics which can be compared between systems. This can be compared with security metrics based on hardware vulnerability assessments when the systems have diverse hardware.

Cyber Security of the Internet of Things

IT and control systems manufacturers are seizing the opportunity of selling new novel hardware devices to consumers, as excitement continues to increase about the coming "Internet of Things" (IoT). As the number of devices continues to increase, more automation will be required for both the consumer (e.g. home and car) and the industrial owner. As the number of devices in IoT and control system increases, software and hardware vulnerabilities will also increase. It is not clear how all of these devices will be adequately protected. Eventually the technology will need to be present in tactical environments in order to accommodate advanced cyber strategies of future adversaries.

Currently, data from IoT hardware sensors and devices are typically handled by proxy network servers (such as a cellphones) since current end devices and wearables have little or no built-in security. The security of the proxy device will be critical if sensor information needs to be safeguarded. The number of sensors per proxy will eventually become large enough so that it will be inconvenient for a single user to manually manage all of the apps for their IoT sensors. This implies new application technologies will be needed that controls many "things" and solves the data management (and vendor collaboration) problem.

An exponentially larger volume of software will be needed to support the future IoT. The average number of software bugs per line of code has not changed, which means there will consequentially be an exponentially larger volume of bugs for adversaries to exploit.

Until there are better standards for privacy protection of personal information and better security guidelines on communication methods and data/cloud storage, security of wearable and other mobility devices will remain poor. More work needs to be spent on designing IoT devices before too many devices are built with default (little or no) security. The ability to create secure IoT devices and services depends upon the definition of security standards and agreements between vendors. ISPs and telecommunication companies will control access to sensor data "in the cloud" and they cannot provide 100% protection against unauthorized access. IoT user data will be at risk.

Diversity of the hardware and software in the future IoT provides strong market competition, but this diversity is also a security issue in that there is no single security architect overseeing the entire "system" of the IoT. The "mission" of the entire IoT "system" was not pre-defined; it is dynamically defined by the demand of the consumer and the response of vendors. Little or no governance exists and current standards are weak. Cooperation and collaboration between vendors is essential for a secure future IoT, and there is no guarantee of success.

The growth of the IoT and the increase in the number of vulnerable commercial sensors has created a situation similar to the current situation of CPSs – a large number of unique hardware devices are interconnected with little or no regard to security, and with little or no communication and security standards. It is not clear that these issues will be resolved before it is necessary to use some of the current and near-future IoT technologies on the battlefield.

Some IoT technologies will necessarily migrate from the consumer arena to the tactical arena, where soldiers will entertain the interconnectivity of a large number of sensors and devices. One technique that can be used to approach the enormous security tasks of the IoT and the "Internet-of-Battle-Things" is to accept the inherent risks of IoT technologies and focus on the most critical areas to protect one's asses. As for our intrusion detection and security modeling methods, one can define the critical elements in one's personal zone of influence and monitor or model only those particular elements. Trying to monitor and measure all possible elements of the IoT system will be increasingly difficult and eventually impossible. In effect, each person or soldier will be analogous to a CPS operator and the devices of interest will have physical, cyber, and process components, as illustrated in Figure 2. Security research of the commercial IoT and the Internet of Battle Things is a current and future area of focus at ARL.

REFERENCES

[1] Cardenas, A. A., Amin, S., & Sastry, S. (2008, June). "Secure Control: Towards Survivable Cyber-Physical Systems," in Proceedings of the 28th International Conference on Distributed Computing Systems Workshops-Volume 00, IEEE Computer Society, pp. 495-500

[2] Colbert, E. & Hutchinson, S. (2016) "Intrusion Detection in Industrial Control Systems," in Cyber-security of SCADA and Other Industrial Control Systems (eds. E. Colbert & A. Kott) (Springer: New York), p. XXX

[3] Colbert, E., Sullivan, D., Hutchinson, S., Renard, K., and Smith, S. (2016) "A Process-Oriented Intrusion Detection Method for Industrial Control Systems," in Proceedings of the 11th International Conference on Cyber Warfare and Security (ICCWS2016), p. 497

[4] Colbert, E., & Kott, A. (2016) Cyber Security of SCADA and Other Industrial Control Systems (Springer: New York)

[5] Evancich, N., & Li, J. (2016) "Attacks on Industrial Control Systems in Industrial Control Systems," iIn Cyber-security of SCADA and Other Industrial Control Systems (eds. E. Colbert & A. Kott) (Springer: New York), p. XXX

[6] Hadziosmanovic, D., Sommer, R., Zambon, E., and Hartel, P. (2014) "Through the eye of the PLC," in Proceedings of the 30th Annual Computer Society Applications Conference (ACSAC 2014), pp. 126-135

[7] Hahn, A. (2016) "Operational Technology and Information Technology in Industrial Control Systems," in Cyber-security of SCADA and Other Industrial Control Systems (eds. E. Colbert & A. Kott) (Springer: New York), p. XXX

[8] Henry, M. H., Zaret, D. R., Carr, J. R., Gordon, J. D., and Layer, R. M. (2016) "yber Risk in Industrial Control Systems," in Cyber-security of SCADA and Other Industrial Control Systems (eds. E. Colbert & A. Kott) (Springer: New York), p. XXX

[9] Langner, R. (2011) "Stuxnet: Dissecting a cyberwarfare weapon," Security & Privacy, IEEE 9.3, pp. 49-51

[10] Long, K. (2004) "Catching the Cyber Spy: ARL's Interrogator," in Proc. of the 24th Army Science Conference, Orlando, FL, DTIC report ADA432198

[11] Luders, S. (2005) "Control Systems under Attack?" in 10th ICALEPCS International Conference on Accelerator and Large Epert. Physics Control Systems, Geneva

[12] Luiijf, E. (2016) "Threats in Industrial Control Systems," in Cyber-security of SCADA and Other Industrial Control Systems (eds. E. Colbert & A. Kott) (Springer: New York), p. XXX

[13] Stouffer, K., Lightman, S., Pillitteri, V., Abrams, M., and Hahn, A. (2015) "Guide to Industrial Control Systems (ICS) Security," NIST Special Publication 800-82 Rev. 2

[14] Sullivan, D. (2015) "Survey of Malware Threats and Recommendations to Improve Cybersecurity for Industrial Control Systems," ARL Technical Report ARL-CR-0759 , February

[15] Sullivan, D., Colbert, E., & Kott, A. (2016) MILCOM, "Network Analysis of Reconnaissance and Intrusion of an Industrial Control System," MILCOM 2016, in press

[16] Sullivan, D., & Colbert, E. (2016) "Network Analysis of Reconnaissance and Intrusion of an Industrial Control System," ARL Technical Report, in press

[17] US Department of Energy (2002) "21 Steps to Improve Cyber Security of SCADA Networks," Washington DC: US Department of Energy

[18] US Executive Order No. 13636 (2013) "Improving Infrastructure Cybersecurity"

[19] Weiss, J. (2010) Protecting Industrial Control Systems from Electronic Threats (Momentum Press: New York)

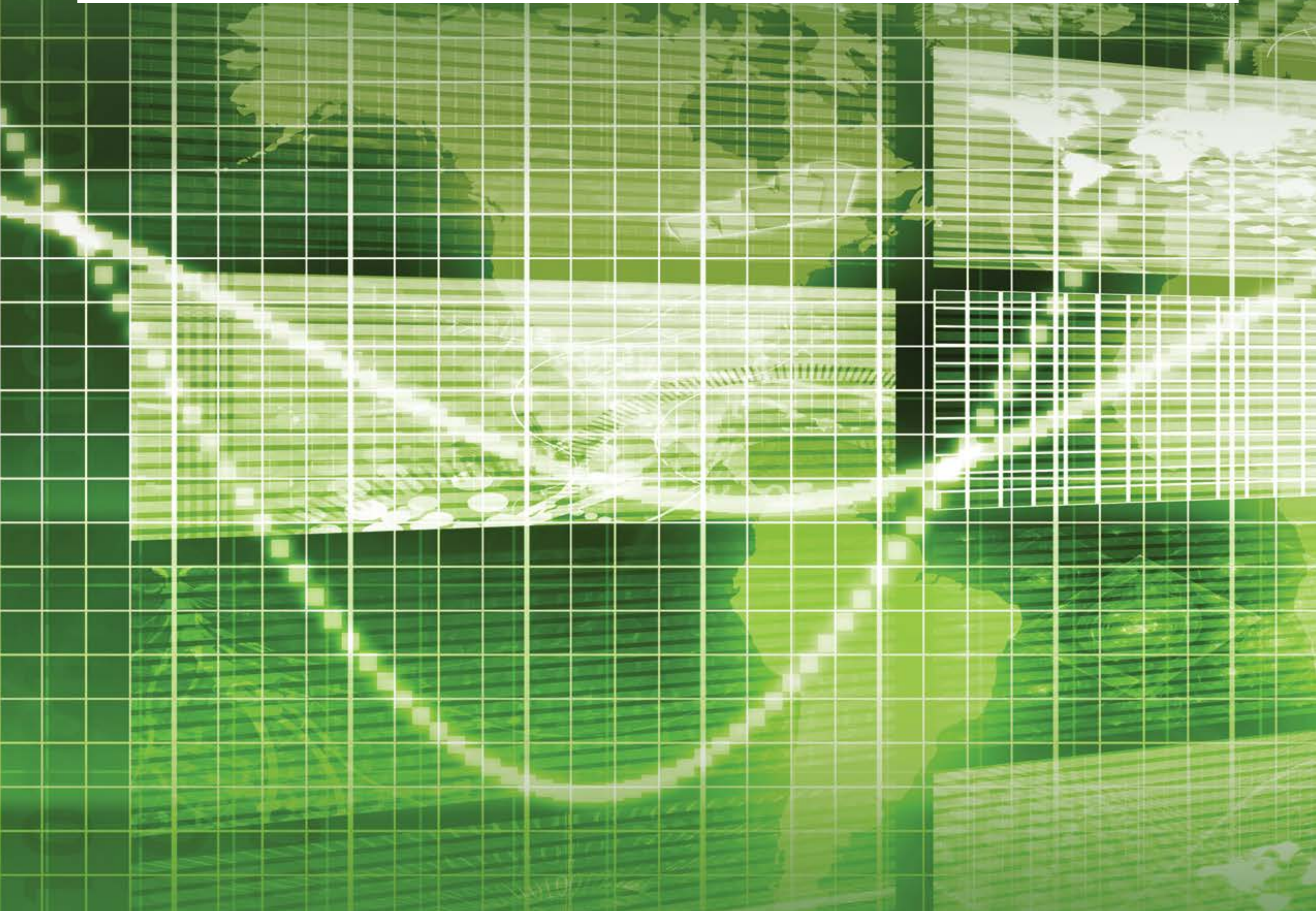
[20] Zetter, K. (2015). Countdown to zero day: Stuxnet and the launch of the world's first digital weapon, (Crown: New York)

[21] Zhu, Q., & Basar, T. (2015) "Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: games-in-games principle for optimal cross-layer resilient control systems," IEEE Control Systems, 35(1), pp. 46-65.

ABOUT THE AUTHOR

Dr. Edward Colbert leads security research on methods for defending Army control systems and Internet of Things (IoT) systems in the Network Science Division at the US Army Research Laboratory (ARL). Before working at ARL, Dr. Colbert performed telecommunications research for the Department of Defense, Verizon, and the Johns Hopkins University Applied Physics Laboratory. Dr. Colbert received the Bachelor of Science degree in Engineering Physics from the University of Illinois (1987), the Master of Science in Physics from the University of Illinois (1988), the Master of Science in astronomy from the University of Maryland (1993), and the Ph.D.

In Astronomy from the University of Maryland (1997). Dr. Colbert holds a research professorship at the Catholic University of America in Washington, DC, and is currently advising several Ph.D. Students at various local institutions. He is also manages the Cyber-security research alliance at ARL, which is a joint academic research program between Army, industry, and academic partners. Dr. Colbert has over 50 publications in refereed journals, and is editor of a recent book by Springer entitled *Cyber Security of SCADA and Other Industrial Control Systems*.



INFORMATION SECURITY CONTINUOUS MONITORING (ISCM)

BY: Mr. Akhilomen Oniha, Mr. Greg Weaver, Mr. Curtis Arnold, and Mr. Thomas Schreck

OVERVIEW: *The ability for commanders to know and understand an organizational attack surface, its vulnerabilities, and associated risks is a fundamental aspect of command decision-making. In the cyberspace domain, ongoing monitoring sufficient to ensure and assure effectiveness of security controls related to systems, networks, and cyberspace, by assessing security control implementation and organizational security status in accordance with organizational risk tolerance and within a reporting structure designed to make real time, data-driven risk management decisions are paramount.*

The National Institute of Standards and Technology (NIST) Special Publication (SP) 800 137, Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations, defines Information Security Continuous Monitoring (ISCM) as “maintaining ongoing awareness of information security, vulnerabilities, and threats to support organizational risk management decisions.”

The Risk Management Framework (RMF) is the unified information security framework for the entire federal government. According to Office of Management and Budget (OMB), by institutionalizing the RMF, “agencies can improve the effectiveness of the safeguards and countermeasures protecting federal information and information systems in order to keep pace with the dynamic threat landscape.”[1] The RMF, developed by the NIST, describes a disciplined and structured process that integrates information security and risk management activities into the system development life cycle. ISCM is a critical part of the RMF process. As such, a foundational component of the ISCM strategy is the need to not only focus on monitoring, but also to support risk management decisions across the multiple mission areas of operations affected by the cyberspace domain.

To assist with the operationalization of ISCM across the entire federal government, the OMB released Memorandum M-14-03, Enhancing the Security of Federal Information and Information Systems. The Memorandum provides guidance to implement ISCM across the Federal Government and help manage information security risk on a continuous bases. In response to M-14-03, the U.S. Army Research Laboratory (ARL) team initiated a program to develop risk scoring at the scale and complexity needed for the DoD. This project, named Information Security Continuous Monitoring (ISCM), is intended to provide a capability that not only allows for the identification of a system risk, but also to allow for that risk to be changed dynamically based on the threat or mission need. This project required a novel approach to risk scoring, as well as a platform that could ingest and visualize the various data types needed, all while fostering collaboration with our federal, academic, and industry partners.

This article discusses the history of ISCM at ARL; the approach and current status of ARL’s ISCM capability; the data, entity creation, and risk scoring processes and models; and the next step and way ahead for ARL’s ISCM capability.

History of ARL ISCM and Initial Approach

In 2011, at the request of the DoD, the ARL team began investigating how to enhance the situational awareness provided by the cyber security tools used in the defense of transactions on DoD information networks. This was the DoD's first major thrust into continuous monitoring based on the success of the State Department's efforts [2]. The ARL team approached ISCM with the primary goal of developing a capability that could continuously correlate and aggregate disparately formatted events generated by intrusion detection, vulnerability assessment, and host-based security tools. At the outset, the minimum bar for success required that ISCM satisfy the following:

- Enhanced cyber situational awareness – The ability to ingest, aggregate, correlate and enrich cyber data from a variety of sources and provide an interface or dashboard view that enables commanders and mission owners to make higher confidence decisions.
- Continuous monitoring – The ability to transform the historically static security control assessment and authorization process into an integral part of a dynamic enterprise-wide risk management process. Providing the Army with an ongoing, near real-time, cyber defense awareness and asset assessment capability.
- Technical transfer – The ability for ISCM to be packaged and transitioned to other organizations with a similar cyber security mission and data sets. In particular, it is important that ISCM be transferable with minimal software refactoring and systems reengineering.

Developing a capability that could continuously correlate and aggregate disparately formatted events

Building off the success from the State Department's continuous monitoring program, in 2011 the State Department's source code was transitioned to the Defense Information Systems Agency (DISA) and National Security Agency (NSA), and was further developed and transitioned to ARL in 2012. This initial ISCM prototype was named JIGSAW and was built atop Splunk [3]. JIGSAW was a collaborative effort between ARL and the DoD High Performance Computing Modernization Program and consisted of a Red Hat Enterprise Linux server, running Splunk on 12 CPU cores, 48GB of RAM and 3TB disk storage. The JIGSAW pilot ran for the majority of 2012 and consisted of a variety of experiments ingesting and exploring approximately 50 gigabytes per day of vulnerability assessment data, host based security logs, intrusion detection events, and network flow data from the DoD Defense Research and Engineering Network.

JIGSAW provided good insights into each individual data set, but its correlation and aggregation capabilities weren't robust enough for our long-term vision. In JIGSAW, there was no entity construct. Every stored row was an event. We could perform aggregations such as: For all data sources indexed, show me all the results per hostname. However, if we then attempted to associate a risk value with the hostname, it was not possible. The aggregation per hostname only existed in the context of the original query results. We potentially could have exported the results to a relational database, established the hostname risk association in a separate table, and then exported the hostname/risk object back to the JIGSAW as a new event. However, we were not willing to contend with the complexity of a transaction that required layering database upon database in order to sure up the deficiencies in each. Furthermore, the cost associated with scaling to the 1TB+ daily volume of data we were expecting to ingest and index made the JIGSAW solution unsuitable for our use case.

Splunk was removed from the ISCM solution and replaced with a relational database backend, PostgreSQL [4]. A Python [5] frontend with a variety of Python libraries were adopted for visualizations and custom Python scripts were developed to perform the data parsing, data correlation, and aggregation. This new configuration addressed entity construct and cost concerns associated with Splunk; however, the scalability issues persisted. In relational databases, the notion of clustering and horizontal scaling is in support of high availability and not scalability or sharding of large data sets. Server parallelism and increase data ingest rates, storage and processing of terabytes worth of data proved to be a difficult task. Furthermore, achieving historical and trending analysis for several months' worth of cyber data was next to impossible, again due to the scalability limitations.

There were many lessons learned from the ISCM prototypes and it helped refine our minimum bar for success to include the following requirement:

- Scalable architecture – ISCM needed to be a scalable architecture that could quickly be augmented with minimal impact to uptime and support the storage and processing of large data sets at the Petabyte scale.

ARL needed to adopt an architecture that could easily scale horizontally and support several months (100TB+) of historical and trending data. Additionally, we needed to consistently ingest and process terabytes of semi-structured data in parallel. With JIGSAW, flow data could only be stored for a couple weeks before we had to delete older data to preserve disk space. This fact led us to commence an investigation of distributed computing and NoSQL architectures, specifically, the Apache Hadoop [6] ecosystem. Several entities in the DoD had already begun to engineer big data frameworks using Hadoop in order to address their mission needs. The ARL team made technology transfer requests in order to build upon existing source code and lessons learned.

Two distributed computation frameworks were evaluated by ARL. The first came from the U.S. Army Intelligence & Security Command and is called Red Disk [7]. The second came from DISA and is called the Big Data Platform (BDP) [8][9]. Many of the components of Red Disk and the BDP are similar. At their core, they are both Hadoop clusters providing a distributed computing framework, with software components capable of ingesting, storing, processing, and visualizing large volumes of data from an assortment of information sources. Both environments are comprised of open source and unclassified components, and also leverage technology transfer from other DoD entities. During our evaluations, we compared the streaming ingest capabilities of each framework for ingesting cyber events via topology constructs (graphs of computations that contain data processing logic) in Apache Storm [10]. Red Disk experienced performance issues when attempts were made to ingest ARL's sensor data into Apache Accumulo [11]. Its custom data processing framework and data-modeling construct averaged less than 1MB/s ingest rate. The BDP performed substantially better with ingest rates that average 50MB/s, with peak rates near 100MB/s.

In the latter part of 2014, the ARL team adopted the BDP to build our ISCM solution as well as future cyber analytic capabilities. Based upon our evaluations, we determined that doing so would substantially reduce the amount of time the ARL team had to spend on architecting a custom Hadoop solution for ingest, storage and processing of our cyber data sources. Additionally, adopting the BDP helped to satisfy requirement for technical transfer and enables a federated approach towards the creation of cyber analytic capabilities among other entities using the BDP. With the BDP acting as the core framework for data ingest, storage and processing, cyber security researchers, scientists, and engineers can focus less on systems engineering and systems integration tasks and more on data modeling and application of statistical, algorithmic and analytic methods to the data in order to glean deeper insight. In the next section, we discuss the current status of ISCM and its supporting hardware.

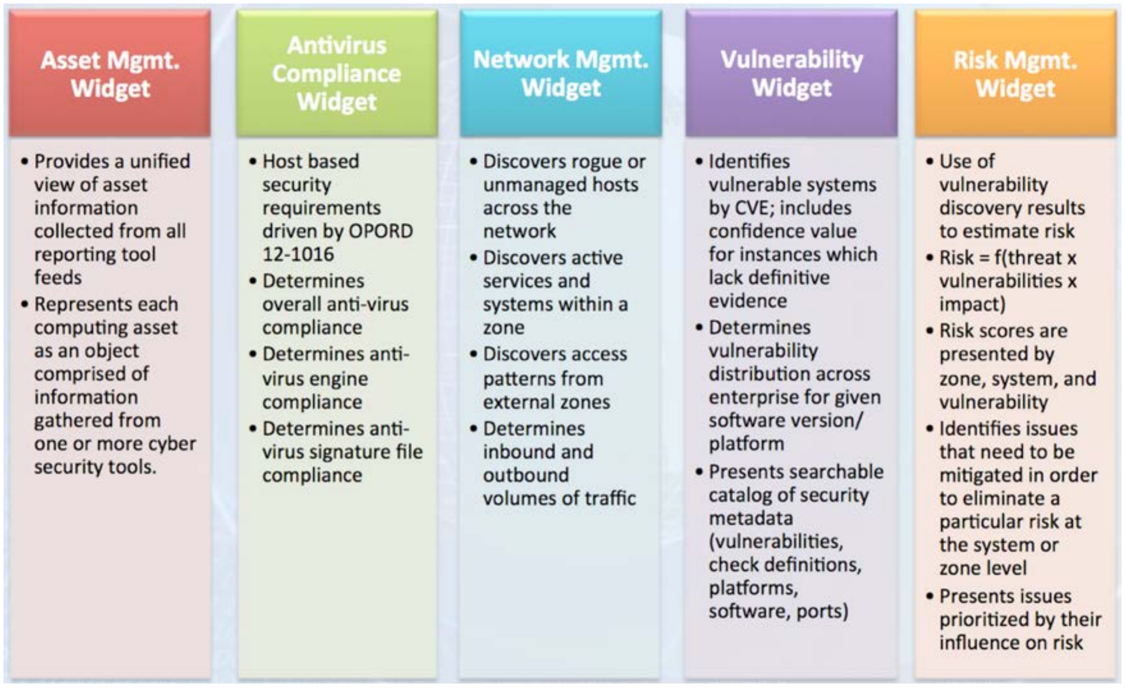
Current Status of ISCM and SUPPORTING Hardware

The current ARL ISCM solution which is built atop of the BDP is based upon five individual widget/analytic capabilities, working in conjunction with one another to provide a dynamic cyber hunting capability and enhanced decision support via a risk categorization and prioritization capability. As illustrated in Fig. 1 below, those individual capabilities include asset management, antivirus compliance, network management, vulnerability management and risk management. The first four capabilities serve as the building blocks to generate the risk picture in the fifth capability.

The ISCM capabilities are based upon attributes and events primarily correlated from four cyber data sources:

- Vulnerability scanning reports collected via the DISA Assured Compliance Assessment Solution (ACAS) [12]
- Host Based Security System (HBSS) reports from tools such as Anti-Virus/Anti-Spyware, collected via Intel/McAfee Enterprise Security products [13]
- Network flow information from ARL's Interrogator Intrusion Detection System [14]
- National Vulnerability Database [15]

Producing the ISCM capability required building multiple instances of the BDP to support the various stages of the software development lifecycle. We acquired hardware to support four medium-size (~30



Asset Mgmt. Widget
- Provides a unified view of asset information collected from all reporting tool feeds
- Represents each computing asset as an object comprised of information gathered from one or more cyber security tools.

Antivirus Compliance Widget
- Host based security requirements driven by OPORD 12-1016
- Determines overall anti-virus compliance
- Determines anti-virus engine compliance
- Determines anti-virus signature file compliance

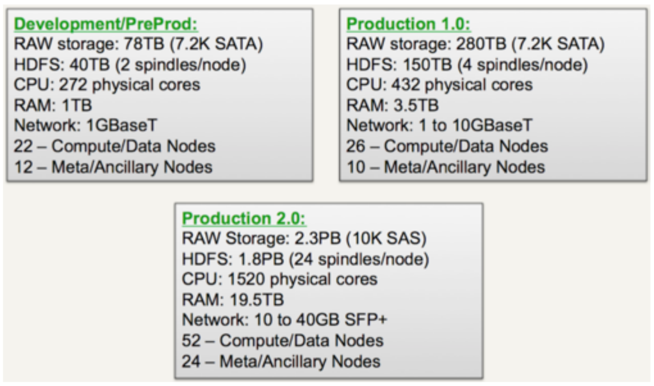
Network Mgmt. Widget
- Discovers rogue or unmanaged hosts across the network
- Discovers active services and systems within a zone
- Discovers access patterns from external zones
- Determines inbound and outbound volumes of traffic

Vulnerability Widget
- Identifies vulnerable systems by CVE; includes confidence value for instances which lack definitive evidence
- Determines vulnerability distribution across enterprise for given software version/ platform
- Presents searchable catalog of security metadata (vulnerabilities, check definitions, platforms, software, ports)

Risk Mgmt. Widget
- Use of vulnerability discovery results to estimate risk
- Risk = f(threat x vulnerabilities x impact)
- Risk scores are presented by zone, system, and vulnerability
- Identifies issues that need to be mitigated in order to eliminate a particular risk at the system or zone level
- Presents issues prioritized by their influence on risk

Fig. 1 ISCM capabilities summary

node) clusters. The clusters served the following roles: A testing and evaluation environment, an external collaboration environment for cyber researchers from academia and other government entities, an environment for capabilities ready for pre-production and a production environment for cyber security analysts at ARL and other stakeholders. The hardware specifications for the various cluster environments are outlined in Fig. 2 below:



Development/PreProd:
RAW storage: 78TB (7.2K SATA)
HDFS: 40TB (2 spindles/node)
CPU: 272 physical cores
RAM: 1TB
Network: 1GBaseT
22 – Compute/Data Nodes
12 – Meta/Ancillary Nodes

Production 1.0:
RAW storage: 280TB (7.2K SATA)
HDFS: 150TB (4 spindles/node)
CPU: 432 physical cores
RAM: 3.5TB
Network: 1 to 10GBaseT
26 – Compute/Data Nodes
10 – Meta/Ancillary Nodes

Production 2.0:
RAW Storage: 2.3PB (10K SAS)
HDFS: 1.8PB (24 spindles/node)
CPU: 1520 physical cores
RAM: 19.5TB
Network: 10 to 40GB SFP+
52 – Compute/Data Nodes
24 – Meta/Ancillary Nodes

Fig. 2 ARL ISCM cluster hardware specifications

The dramatic increase in capability between production 1.0 and production 2.0 was primarily driven by a desire for enhanced historical and trending analysis by increasing the window of time elapsed before deleting data. Furthermore, we had experienced I/O bottlenecks with the 1.0 version of production and determined that more spindles per node would alleviate that problem. Finally, we wanted to leverage in-memory computation engines like Apache Spark [16], so we nearly tripled the available random access memory per node. With the hardware in place, we were ready to tackle the ISCM data modeling and knowledge engineering tasks.

Data modeling

Figure 3 below illustrates the core of the ISCM data model, the entity construct. At the core of the model is the host element, which is defined as a computer, printer or other managed network device (e.g. switch, router or firewall).

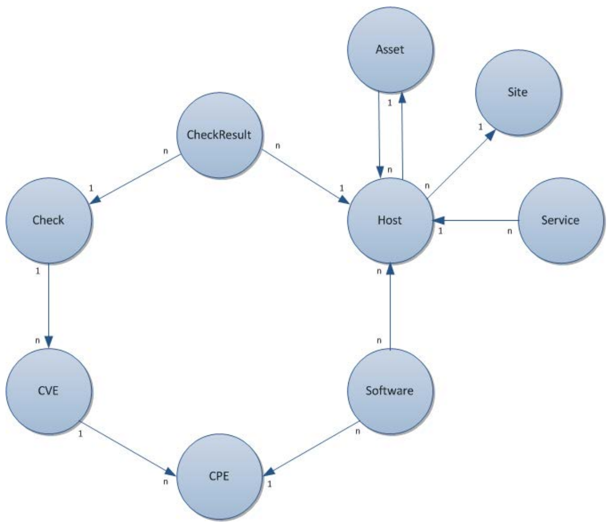


Fig. 3 ISCM entity model

ISCM leverages Storm's stream processing capabilities to create entities and stores them in an entity-modeling construct in Accumulo. Each line of input or raw data is processed for any relevant entity data and the attributes and relationships that go into that entity. Entities, attributes, and relationships are created without querying the entity model first. This may seem counter-intuitive because duplicate entities may already exist in Accumulo; however, duplicates are expected and if an entity is repeated, only the entity with the most current timestamp is returned during the table scan. Furthermore, only three of the same entity identifiers are preserved after the database has gone through compaction (written from memory to disk). This puts the responsibility of handling excessive duplication on Accumulo, allowing Storm to ingest and parse the data as quickly as possible. Finally, storing the state of the entity object (including some duplicates) over time provides the basis for historical and trending analysis in ISCM. Currently ISCM can keep state for three months' worth of entity objects.

Some data sources, such as ACAS vulnerability assessment reports or HBSS system properties, allow for the creation of two entities and the bidirectional relationship between them. This is a simple case since the entities are available when the relationship is created and the entity is known to explicitly exist.

Other data sources, such as HBSS asset configuration compliance module (ACCM), only creates one entity and the relationships with the host presumed to be reported by another feed type; system properties in this example. Although we do not have a host entity because we derive the host ID from the given data, the software entity is created with all the normal attributes while a relationship is added. Furthermore, we can create the opposite direction of the relationship by creating an entity representing that host that will have the relationship added to it. This works even in the case of the ACCM data being processed before the system properties data because the entity model allows for hanging relationships (i.e. when the host is not actually stored yet). If one were to run a query and discover this, one would know the ID of the host, and some of the software relationships it has, even if all of the host information is not yet available.

In March of 2015, when the asset management widget (see Fig. 1) was completed, we noticed that querying the entity model from the web frontend, certain types of queries took several minutes to return results. We realized that this was due to the fact that Accumulo was not designed to support query-focused datasets. Operations such as order by, group by, and count could not be accomplished without pre-computing the queries via a Hadoop MapReduce [17] job. This was unacceptable because our vision was to allow the adopters of ISCM the ability to ask any questions of the data that were of interest to them. Question such as:

› How many assets are in a given enclave?
› Which assets are in a specific VLAN?
› Which assets have outdated anti-virus signatures?
› Which assets have vulnerabilities greater than 200?
› Which assets have outdated scan results?
› Which assets have communicated with a foreign country recently?

As a result of this limitation in Accumulo, we further refined our initial minimum bar for success to include the following requirement:

- Low latency queries – As a query focused capability, ISCM needs to provide rapid responses to simple and compound queries from both end users and statistic/analytic processes.

The ARL team petitioned the DISA BDP change advisory board to incorporate the Elasticsearch [18] into the BDP baseline. Elasticsearch enhanced the user experience and allowed for users to issue dynamic queries, and in many cases, receive sub-second responses. This allowed for a much more dynamic experience.

At this point, the BDP architecture satisfied all of our requirements and the remainder of 2015 was spent developing and integrating the three remaining widgets: antivirus compliance, network management and vulnerability management. In November 2015, we began our investigation of how to illustrate the risk picture.

The ARL team began by leveraging the generic algorithm for cyber risk, where risk is a function of threats, vulnerabilities and impact, $R = f(T x V x I)$. We supplemented the equation with additional characteristics that were critical to the defensive operations mission, including:

Confidence Level: The belief that an asset is exposed to a particular vulnerability by taking into account all relevant observations (i.e. output from all tools: HBSS, ACAS, etc.) This value is a derived percentage, currently implemented using term frequency–inverse document frequency algorithm [19] and represents the certainty that the host in question actually has the factors deemed to be vulnerable (i.e. software version, patch version, operating system, etc.)

Threat Multiplier: A factor associated with the exposure of a vulnerability to an external network for remote exploitation. Vulnerabilities exposed to a wide area network and remotely exploitable, produce the highest threat multiplier.

Risk Management Approach

Similar to first four ISCM widgets, the risk management widget (RMW) is designed to provide situational awareness from the command level down to the asset level. The ARL team's initial approach towards risk management provides a mechanism for stakeholders to prioritize the systems with the highest risk of compromise, based upon the NVD common vulnerabilities and exposures (CVE) and other factors. Fig. 4 above illustrates how ISCM visually represents the count of hosts (line graph) and the count of risks (bar graph) associated with a particular enclave. Each risk factor is a vulnerability identified through the installed software and CVE as depicted in Fig. 5.

The two primary functions of the RMW are risk identification and risk scoring. The functionality that determines the cause of the risk, identifies issues that are mitigated in order to eliminate a particular risk at the asset or site level. Issues are prioritized by their influence on risk. The relative risk scoring functionality uses vulnerability discovery results to estimate risk where risk score is based on the exposure of vulnerable services to external networks. Risk scores are presented by site, asset, or vulnerability.

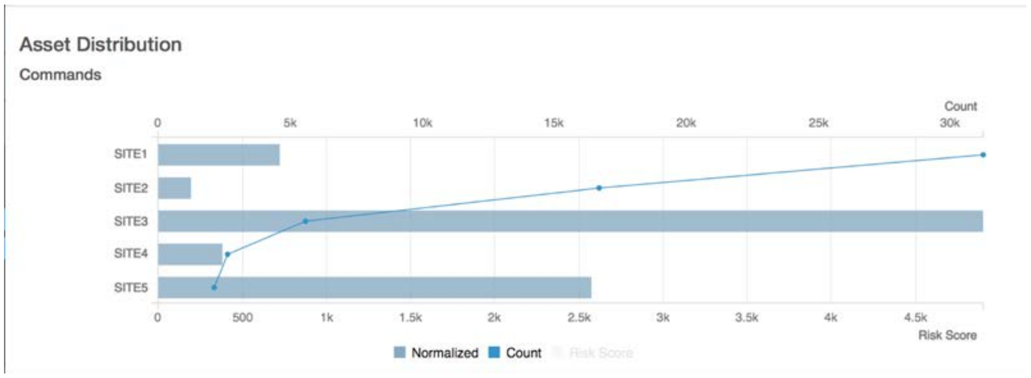
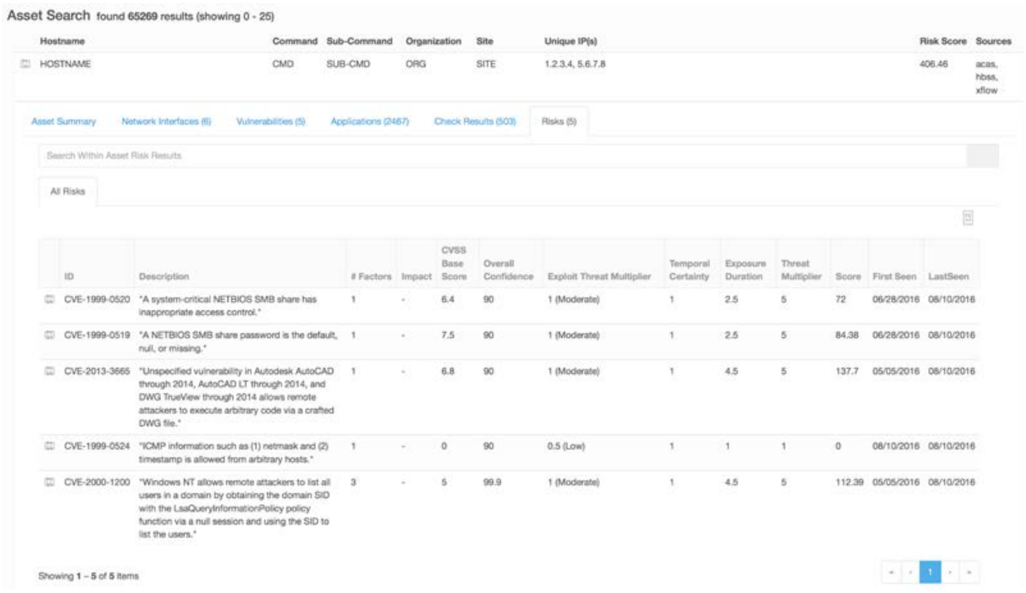


Fig. 4 ISCM user interface - host count and risk score visualization



Asset Search found 65269 results (showing 0 - 25)

Hostname	Command	Sub-Command	Organization	Site	Unique IP(s)	Risk Score	Sources
HOSTNAME	CMD	SUB-CMD	ORG	SITE	1.2.3.4, 5.6.7.8	406.46	acas, hbss, xflow

Asset Summary | Network Interfaces (6) | Vulnerabilities (5) | Applications (2467) | Check Results (503) | Risks (5)

Search Within Asset Risk Results

All Risks

ID	Description	# Factors	Impact	CVSS Base Score	Overall Confidence	Exploit Threat Multiplier	Temporal Certainty	Exposure Duration	Threat Multiplier	Score	First Seen	LastSeen
CVE-1999-0520	"A system-critical NETBIOS SMB share has inappropriate access control."	1	-	6.4	90	1 (Moderate)	1	2.5	5	72	06/28/2016	08/10/2016
CVE-1999-0519	"A NETBIOS SMB share password is the default, null, or missing."	1	-	7.5	90	1 (Moderate)	1	2.5	5	84.38	06/28/2016	08/10/2016
CVE-2013-3665	"Unspecified vulnerability in Autodesk AutoCAD through 2014, AutoCAD LT through 2014, and DWG TrueView through 2014 allows remote attackers to execute arbitrary code via a crafted DWG file."	1	-	6.8	90	1 (Moderate)	1	4.5	5	137.7	05/05/2016	08/10/2016
CVE-1999-0524	"ICMP information such as (1) netmask and (2) timestamp is allowed from arbitrary hosts."	1	-	0	90	0.5 (Low)	1	1	1	0	08/10/2016	08/10/2016
CVE-2000-1200	"Windows NT allows remote attackers to list all users in a domain by obtaining the domain SID with the LsaQueryInformationPolicy policy function via a null session and using the SID to list the users."	3	-	5	99.9	1 (Moderate)	1	4.5	5	112.39	05/05/2016	08/10/2016

Showing 1 – 5 of 5 items

Fig. 5 ISCM user interface - example host risk posture and score

Temporal Certainty Multiplier: A factor associated with the age and freshness of the vulnerability scan reports. As scan information ages, the potential risk from vulnerabilities that cannot be confirmed as mitigated increases. The temporal certainty multiplier represents the increase as a factor, which is multiplied against the vulnerability instance risk score. The time period is determined by comparing the greatest last seen of all risk factors to current time.

Exposure Duration Multiplier: A factor indicating how long an unresolved risk was first detected. The time period is determined by comparing the earliest first seen of all risk factors to current time.

Exploit Threat Multiplier: A dynamic factor that allows a security analyst to amplify/decrease the weighting of a CVE risk score throughout the system. The value is set on the user interface through a RESTful service, which updates the entity model. The factor values can be very low, low, moderate, high and very high, depending upon the level of exploitation and other threat intelligence. If no multiplier is stored in the entity model, then the default value is moderate as depicted in Fig. 5.

ISCM's current approach to risk management satisfies its initial goal of aiding stakeholders in the comparison and prioritization of higher-risk versus lower-risk assets. However, when looking at assets independently, the risk score does not provide the context necessary to assess the actual risk of an asset being compromised. The ARL team is actively implementing a probabilistic risk scoring widget, primarily based upon research performed at Massachusetts Institute of Technology – Lincoln Laboratory [20] and Johns Hopkins University Applied Physics Laboratory [21].

Future Direction

ISCM is not only a technical problem, it also requires policy actions in order to achieve and sustain its goals. The ISCM program at ARL will continue to be incrementally improved with the appropriate rigor and assessment frequencies to support the mission/business requirements, risk tolerance, and security categorization. By leveraging an integrated operational and technical ISCM portal, the Cyber Security Service Provider (CSSP) operations process and knowledge management capabilities ensure sustained and continuous assessments can be synchronized across the Army. To support ongoing risk determinations and future risk acceptance decisions by senior leaders, policies supporting the following six steps are necessary for achieving and sustaining an effective ISCM:

- Define an ISCM policy, strategy, and supporting doctrine based on risk tolerance that promotes clear visibility into assets, awareness of vulnerabilities, up-to-date threat information, and mission/business impacts.
- Ensure its ISCM program determines metrics, status monitoring frequencies, control assessment frequencies, and an ISCM technical architecture.
- Automate collection, analysis, and reporting of data where possible. Collect the security-related information required for metrics, assessments, and reporting.
- Analyze the data collected and report findings, determining the appropriate response. It may be necessary to collect additional information to clarify or supplement existing monitoring data.
- Respond to findings with technical, management, and operational mitigating activities or acceptance, transference/sharing, or avoidance/rejection.
- Review and update the monitoring program, adjusting the ISCM strategy and maturing measurement capabilities to increase visibility into organizational assets and awareness of vulnerabilities, further enable data–driven control of the security of an organization's information infrastructure, and increase organizational resilience.

In 2017, ARL will release an ISCM Widget to support continuing re-authorization capabilities. This capability facilitates the NIST SP 800-137 requirement "that security controls and organizational risks are assessed and analyzed at a frequency sufficient to support risk-based security decisions…"

In 2017, ARL will propose a widget(s) that could support Mission Assurance Continuous Monitoring (MACM), an integrated observation of mission-aligned ISCM with operational and technical information network operations capabilities to create and preserve information assurance on the DoD information networks and increase organizational resilience.

In 2018, ARL will propose a widget(s) that could support Cyber Defense Continuous Monitoring (CDCM), an integrated global observation of mission-aligned partners through passive and active cyberspace operations intended to preserve the ability to utilize friendly cyberspace capabilities and protect data, networks, net-centric capabilities, and other war fighting and support enabling systems.

ARL will continue to develop ISCM and ensure that its requirements are well informed and reflect the best practices, lessons learned, and efficiencies developed across the Army.

REFERENCES

[1] Burwell, S. M. (2013, November 18). “Enhancing the Security of Federal Information and Information Systems” [Memorandum]. Washington, DC: Office of Management and Budget. Retrieved from https://www.whitehouse.gov/sites/default/files/omb/memoranda/2014/m-14-03.pdf

[2] “Implementing Continuous Risk Monitoring at the Department of State” (2010, May). Retrieved from http://www.state.gov/documents/organization/156865.pdf

[3] Splunk. http://www.splunk.com/en_us/products/splunk-enterprise/features.html

[4] PostgreSQL. https://www.postgresql.org/about/

[5] Python. https://www.python.org/about/

[6] Apache Hadoop. http://hadoop.apache.org/

[7] Richardson, R. D. (n.d.). “INSCOM - Big Data”. Retrieved from https://info.publicintelligence.net/INSCOM-BigData.pdf

[8] Bart, D. V. (2016, April 22). “Big Data Platform (BDP) and Cyber Situational Awareness Analytic Capabilities (CSAAC)”. Retrieved from http://www.disa.mil/~/media/Files/DISA/News/Conference/2016/AFCEA-Symposium/4-Bart_Big-Data_Platform_Cyber.pdf

[9] “DISA’s Big Data Platform and Analytics Capabilities” (2016, May 16). Retrieved from http://www.disa.mil/NewsandEvents/News/2016/Big-Data-Platform

[10] Apache Storm. http://storm.apache.org/

[11] Apache Accumulo. https://accumulo.apache.org/

[12] ASSURED COMPLIANCE ASSESSMENT SOLUTION (ACAS). Retrieved July 20, 2016 from http://www.disa.mil/cybersecurity/network-defense/acas

[13] ANTI-VIRUS/ANTI-SPYWARE SOLUTIONS. Retrieved July 20, 2016 from http://www.disa.mil/Cybersecurity/Network-Defense/Antivirus

[14] Long, K. S. (2004, December). “CATCHING THE CYBER SPY: ARL’S INTERROGATOR”. Retrieved from http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA432198

[15] National Vulnerability Database. https://nvd.nist.gov/

[16] Apache Spark. http://spark.apache.org/

[17] Hadoop MapReduce. https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html#Overview

[18] Elasticsearch. https://www.elastic.co/products/elasticsearch

[19] Term Frequency-Inverse Document Frequency (n.d.) Retrieved from http://www.tfidf.com/

[20] Lippmann, R.P, Riordan J.F, Yu T.H, and Watson K.K. (2012, May 22). “Continuous Security Metrics for Prevalent Network Threats: Introduction and First Four Metrics,” [Whitepaper]. MIT-Lincoln Labs. Retrieved from https://www.ll.mit.edu/mission/cybersec/publications/publication-files/full_papers/2012_05_22_Lippmann_TechReport_FP.pdf

[21] Watkins, L.A., Hurley, J.S. “Cyber Maturity as Measured by Scientific Risk-Based Metrics” Journal of Information Warfare (2015) 14.3: 60-69. Retrieved from https://www.researchgate.net/publication/280953172_Cyber_Maturity_as_Measured_by_Scientific_Risk-Based_Metrics

ABOUT THE AUTHORS

Mr. Akhilomen Oniha has over a decade of experience in the areas of information technology, Linux systems engineering, distributed computing and security engineering. Mr. Oniha is the Lead for Technical Architecture at the U.S. Army Research Laboratory (ARL) Sustaining Base Network Assurance Branch (SBNAB). Mr. Oniha holds a BS in Computer Science and a 2nd BS in Information Technology from University of Maryland University College. He also holds an MS in Computer Science with a focus on Information Assurance from Johns Hopkins University. His research interests include data science techniques, malware reverse engineering and vulnerability analysis.

Mr. Curtis Arnold is the Chief of the Sustaining Base Network Assurance Branch at the U.S. Army Research Laboratory. The Sustaining Base Network Assurance Branch is responsible for performing a wide-range of Information Assurance activities from Research & Development to providing 24/7 Computer Network Defense services. Computer Network Defense Services include oversight of more than 100 external customers and monitoring of over 300 intrusion detection sensors around the world.

Mr. Arnold has supported ARL for over 10 years in a variety of leadership, policy, and technical roles. Before joining ARL, Mr. Arnold was a Non-Commissioned Officer in the U.S. Army Judge Advocate General’s Corps. Mr. Arnold holds a BS in Information Security and an M.S. in Information Technology from Johns Hopkins University. Mr. Arnold is currently pursuing his Doctorate in Information Assurance from Capitol College.

Mr. Thomas Schreck has 13 years of varied development experience from distributed computing, cryptography, user interface design, and legacy systems. Mr. Schreck is the lead KeyW Corporation analytics team developer, working on the Information Security Continuous Monitoring (ISCM) project for U.S. Army Research Laboratory (ARL). Mr. Schreck holds a BS in Computer Science and completed the course requirements for a BS in Applied Mathematics at New Jersey Institute of Technology. He also holds an MS in Computer Science with a focus in Information Assurance from Johns Hopkins University.

Cyber Security and Information Systems
Information Analysis Center
100 Seymour Road
Suite C-102
Utica, NY 13502

Return Service Requested

PRSRT STD
U.S. Postage
P A I D
Permit #566
UTICA, NY

THE CENTER OF EXCELLENCE IN CYBER SECURITY AND INFORMATION SYSTEMS

Leveraging the best practices and expertise from government, industry, and academia in order to solve your scientific and technical problems

HTTPS://WWW.CSIAC.ORG/JOURNAL-ISSUE/