IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON SECURITY AND TRUSTED COMPUTING FOR INDUSTRIAL INTERNET OF THINGS

Received June 30, 2018, accepted August 12, 2018, date of publication August 28, 2018, date of current version September 21, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2867556

A Master Attack Methodology for an AI-Based Automated Attack Planner for Smart Cities

GREGORY FALCO[1], ARUN VISWANATHAN[2], CARLOS CALDERA[1], AND HOWARD SHROBE[1]
[1]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109 USA

Corresponding author: Gregory Falco (gfalco@mit.edu)

This work was supported in part by the Jet Propulsion Laboratory (JPL), California Institute of Technology, through a contract with the National Aeronautics and Space Administration (NASA) and in part by the Internet Policy Research Initiative (IPRI) at the Massachusetts Institute of Technology (MIT).

ABSTRACT America's critical infrastructure is becoming ''smarter'' and increasingly dependent on highly specialized computers called industrial control systems (ICS). Networked ICS components now called the industrial Internet of Things (IIoT) are at the heart of the ''smart city'', controlling critical infrastructure, such as CCTV security networks, electric grids, water networks, and transportation systems. Without the continuous, reliable functioning of these assets, economic and social disruption will ensue. Unfortunately, IIoT are hackable and difficult to secure from cyberattacks. This leaves our future smart cities in a state of perpetual uncertainty and the risk that the stability of our lives will be upended. The Local government has largely been absent from conversations about cybersecurity of critical infrastructure, despite its importance. One reason for this is public administrators do not have a good way of knowing which assets and which components of those assets are at the greatest risk. This is further complicated by the highly technical nature of the tools and techniques required to assess these risks. Using artificial intelligence planning techniques, an automated tool can be developed to evaluate the cyber risks to critical infrastructure. It can be used to automatically identify the adversarial strategies (attack trees) that can compromise these systems. This tool can enable both security novices and specialists to identify attack pathways. We propose and provide an example of an automated attack generation method that can produce detailed, scalable, and consistent attack trees–the first step in securing critical infrastructure from cyberattack.

INDEX TERMS AI planning, attack trees, cyber audit tools, cyber risk, cybersecurity, IIoT, IoT, smart cities.

I. INTRODUCTION

Critical infrastructure such as CCTV security networks, the electric grid, water networks and transportation systems operate using industrial control systems (ICS). Increasingly, as cities move to become ''smart cities'', ICS are networked together for ease of use and expense reduction. ICS devices and their associated sensors are interconnected via a network that now comprises the Industrial Internet of Things (IIoT). The IIoT is a component of what Cisco originally coined as the Internet of Everything (IoE) which describes IIoT devices used for the purposes of smart cities [10].

While IIoT provides convenience, it comes at an associated cost. ICS that make up the IIoT is constantly subject to cyber-attack. Kaspersky Labs, a leading cybersecurity research and antivirus company, found that in 2016 one in every five ICS are attacked each month [15]. Further, not all of these attacks used the internet to penetrate the IIoT. Others used vectors including removable media [15].

Public administrators need to understand cyber threats. This is abundantly apparent from the recent attacks against state and city infrastructure such as the Colorado Department of Transportation (CDOT) ransomware attack in February 2018 that disabled CDOT processes for days [7] and the SamSam ransomware attack that brought city services in Atlanta to a halt in March 2018 [2]. However, given the number of critical infrastructure components in any municipality, and the vast variety of configurations involved, it would be too time consuming to enumerate every attack pathway adversaries might take. To date, local government administrators have not been active participants in conversations on cybersecurity and have prioritized this matter [17]. Perhaps one reason for this is that public administrators have an inadequate understanding of their digital asset's risk profile [35]. The traditional approach to enumerating attack vectors and understanding technical digital risk involves creating attack trees (also called attack graphs). Developing an attack tree

for each critical infrastructure would be tedious and require highly technical knowledge as well as associated knowledge about mechanisms that might be used to attack each system. A public administrator or his/her team is not likely to have the necessary expertise to do this. This leaves cities fully exposed.

Today, artificial intelligence (AI) is being used in many industrial sectors and government organizations to enhance efficiency and scale operations. The challenge of quickly and easily enumerating critical infrastructure attack vectors can be addressed using AI. In this paper, we describe an AI planning system design that can enumerate a set of multi-step attack plans capable of penetrating and compromising systems across IP-networked devices. Importantly, our proposed method is ''industry sector agnostic'' meaning that it is designed to accommodate a wide range of organizations and computing systems. While automated attack planners have been developed previously, they have not used standardized cybersecurity frameworks, leading to semantic deficiencies when describing particular attack plans. Further, existing attack planners, because of the speed at which things are changing, do not have rule sets built to accommodate modern IoT/IIoT systems. The contribution of this research will be to develop a master attack planner's ontology. We call it ''master'' attack ontology because our goal is to design an attack ontology that accommodates any IP networked system in any industry sector. The example attack graph we develop provides automatic identification of adversarial strategies that can be used to compromise a CCTV network whose typology is similar to other IP-based networks. While an example of the automated methodology is provided, and compared with a manually generated tree, we do not have sufficient system environment data available to test our new AI planning system across more than one critical infrastructure system. That research will follow in future studies. Therefore, this study is limited to developing, but not testing across multiple environments, the efficacy of our automated alternative to existing attack planning systems.

II. BACKGROUND

Attack trees are used to enumerate the threat pathways that attackers could use to penetrate a system. The first publication on attack trees was by Bruce Schneier in 1999 in Dr. Dobb's Journal of Software Tools [30]. His article described an approach based on a well-documented and frequently used reliability analysis technique created in 1962 at Bell Telephone Laboratories called Fault Tree Analysis [9]. The intent of fault tree analysis was to evaluate system failure risks that could cause an inadvertent launch of an intercontinental ballistic missile.

As a general example illustrated in Fig. 1, the root of the fault tree structure is the failure, and the leaves are possible causes of the failure. Each leaf has an associated probability that the cause will occur. Causes may be dependently linked and categorized as ''and'' logic gates. ''And'' leaves must both occur for the failure to take place. This is distinct from ''or'' leaves where two leaves may be present but only one of the two causes is needed to generate a failure. The tree proceeds downward from the root with some causes having subsequent levels of sub-causes which may include both the ''and'' and ''or'' logic gates [5].

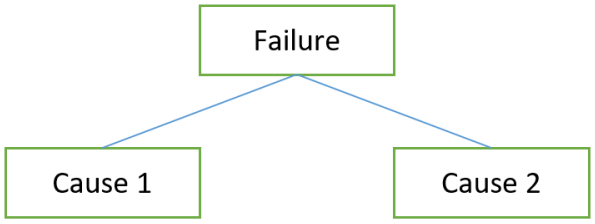


FIGURE 1. Sample fault tree hierarchy.

The fault tree is completed once there are no longer any traceable causes or sub-causes for a given failure that have not already been included.

A. ATTACK TREE FEATURES

Attack trees are functionally similar to fault trees, however, they usually serve a different objective, rely on different risk quantification methods and call for different outcome interpretation.

1) OBJECTIVE

Where fault trees start with a specific failure of a system as the root of the tree, the root of an attack tree is the goal of the attacker. The goal might be stealing money from a safe, or stealing passwords from a secure online database [30]. The leaves of the attack tree, unlike a fault tree, are the discrete actions that must be taken to achieve the objective. An example of an attack tree appears in Fig. 2.

2) QUANTIFICATION

In addition to the root being different in attack trees, the quantification method varies as well. While some attack trees use probabilities to quantify risk, it is more common for attack trees to use qualitative (i.e. ordinal) measures to score each leaf [5]. Such qualitative measures make more sense because of the obstacles to assigning a probability to the likelihood of an attack vector being pursued. Ordinal measures might involve a rating of ''difficulty to penetrate'' and use a ranking of the leaf from 1 to 5. Or, they might involve ranking the level of knowledge needed to penetrate on a scale of 1–10. Another way of quantifying the leaves in an attack tree is to use economic indicators. An example might be that it costs $10 to pay for a dictionary password cracker versus $100 to buy a listserv address so that a successful phishing attack can be waged. Still another quantification metric might be timing, where password cracking could be rated at 10 hours and a phishing attack could be rated at 24 hours.

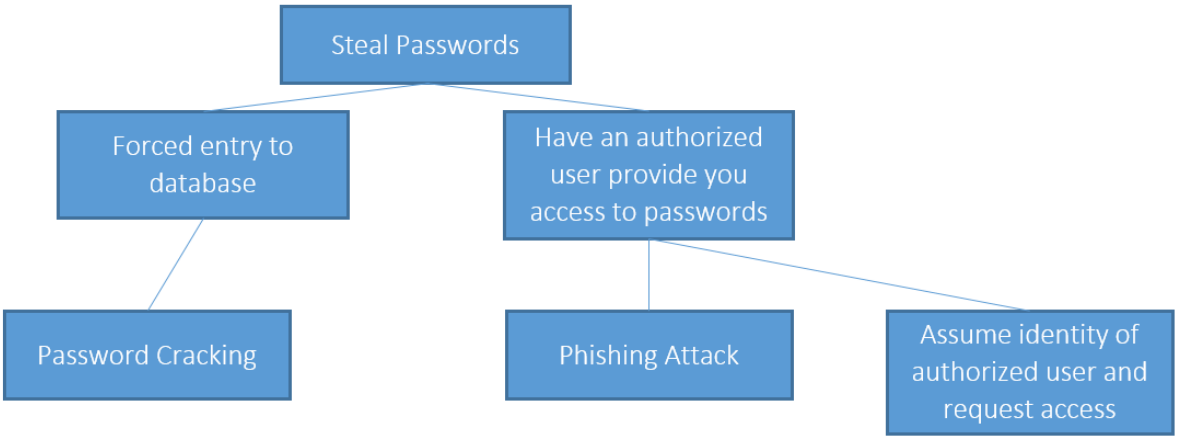


FIGURE 2. Example attack tree.

3) OUTCOME INTERPRETATION

Attack trees are most appropriately used to determine the easiest or optimal line of attack for a hacker. Some researchers will assess this by determining which pathways on the tree have the fewest leaves on them (suggesting the least complicated attack route) while others might use the combined quantification metrics as a guide to determine the most desirable attack vector (from an attacker's standpoint).

B. BENEFITS OF ATTACK TREES

The benefits of attack trees are manifold. Fundamentally, attack trees enable the user to identify the potential areas of intrusion based on a goal established by a putative attacker [5].

Attack trees provide a causal framework for thinking about possible disruptive events. They also help to structure the complex problem of defending against cyberattacks [25]. The sequential attack plans represented by each leaf, force system ''defenders'' to think through all possible avenues of attack. Further, attack trees make it easier for defenders to enumerate possible defense mechanisms associated with each leaf of the tree [27].

The flexibility of attack trees is valuable from a usability perspective. A security researcher who cares more about managerial cybersecurity rather than technical cybersecurity can use the attack tree in a way best suited to his or her purpose. This is because attack trees allow researchers to conduct analyses at multiple levels of abstraction. Researchers can acknowledge an attack vector in their attack tree without deep knowledge of the sub-leaves of the pathway, and instead focus on the topics of greatest investigative interest [5]. For example, if a researcher wants to focus on possible social engineering attacks on a target, and wants to develop an attack tree, the researcher can take note of the top level leaves describing technical exploitation of the target but spend most of their time focusing on the nodes most vulnerable to social engineering forms of attack.

Attack trees can be used as predictive tools as well as reactive security tools. When designing a cyber system, an attack tree can be used to evaluate the various security requirements needed to protect that system. Alternatively, an attack tree can be used to audit or evaluate the security of a legacy or existing system to determine how vulnerable it might be to attack. This could clarify the best investments for securing the system.

Finally, attack trees, if structured properly, can be scalable. Because the end goal of the hacker is not always specific to a particular model of a system, common attacker goal trees are reusable, and can help to anticipate more complex system attacks [32].

C. CHALLENGES OF ATTACK TREES

While fault tree analysis is considered the ''gold standard'' for aeronautic reliability testing, attack trees have not yet achieved that standard in the cybersecurity arena [9]. The primary reason fault tree analysis assigns probabilities to each leaf is to quantify risk. This provides a historical baseline for analysis [9]. In 1931, Shewhart categorized causes of failure as ''assignable'' or ''chance'' [31]. Assignable causes related to failures can be detected and controlled as opposed to chance causes which are uncontrollable and random. Mechanical systems have assignable causes of failure whereas cyber systems have both assignable and chance causes. Failure rates for mechanical systems can easily be determined in a lab by running the mechanical system constantly and subjecting it to all possible conditions [11]. Even if a cyber system is constantly run and exposed to all known conditions, it is not possible to calculate precise failure probabilities because it cannot be known how or at what frequency an intelligent attacker might attempt to exploit a vulnerability in a cyber system. The scope of intelligent attacker threats makes establishing accurate probabilities of attack difficult.

While the failure potential of a physical system is finite, this is not the case for cyber systems. For example, there are only so many ways a person can break into a safe. A cyber system, due to its complexity and interconnectivity, can be

exploited in innumerable ways. The exploitability of a system largely depends on the other systems to which the device of interest is connected.

Perhaps the most significant challenge with attack trees is that they need to be prepared by an expert who has both full knowledge of the system and a comprehensive understanding of how best to attack the system. It is not always possible to secure the services of such an expert. Developing comprehensive attack trees is time consuming. Further, manually creating attack trees always starts from ground zero, and thus is inconsistent across security experts. Semantic idiosyncrasies in the security researcher community introduce additional challenges when attempting to compare risks across different systems [26].

D. CURRENT STATE

To address some of these challenges, attack trees can be made more accessible and readily available by using artificial intelligence planning logic. Shortly after attack trees were first documented, Shrobe and Howard [32] developed an early automated attack tree generator using classical planning at the Massachusetts Institute of Technology's Computer Science and Artificial Intelligence Laboratory (CSAIL). Classical planning is a branch of artificial intelligence. Classical planning requires an initial state, a goal state and a series of operators. The goal is to sequence these operators to achieve the goal state, starting from the initial state. Because of the deterministic functions of computing systems and associated attacks, classical planning is an efficient means of developing attack trees in a scalable way.

There are two fundamental components of an attack planner:

1) an abstracted rule set describing various methods and techniques for attacking a system, which ideally should be broadly applicable across all systems;
2) a detailed system description of the environment for which the attack tree is needed.

The system environment describes the network topology, the system components and their subsystems, data access rights and locations, and associated dependency relationships.

For an organization seeking to understand the cyber risks they face, the automated generator removes the requirement of having a person develop the tree who is knowledgeable about all possible ways of attacking their system. The only input required is a system description of the environment. These tend to be generally available. A system that automatically generates attack trees then enables organizations to spend more time establishing the correct metrics for evaluating risk, rather than focusing on the enumeration of attack vectors.

The attack tree generator used as the basis for our work was developed by Shrobe and Howard [32]. This planner was built to enumerate attacks against the CSAIL computing network as it was configured when the planner was designed in 2002. The planner incorporated a system model of the CSAIL computing environment as well as an attack method ruleset designed to defeat the security triad (i.e. confidentiality, integrity and availability) of the system environment. When the attack planner is given the goal of compromising a node of the system environment, the planner uses directed backward search to reason through each operator (consisting of the attack ruleset) to enumerate all possible pathways to achieve the attack goal [32]. This is how the attack tree is ultimately generated.

The number of system variables in the system model description results in a very large search space. This could prove problematic if the speed of the planner was important for our work. Because we foresee this planner being used in offline activities like a cybersecurity audit, the considerable search space and resulting time required to generate all pathways is not something we are concerned with for our current work. Should we ultimately want to use the planner for real-time system analysis, we would need to optimize the planner accordingly to address the search space issue.

While existing automated attack generators are more consistent than manually created attack trees, issues remain. Automated generators today do not incorporate standardized language from the cybersecurity community into the trees. This misses an opportunity to incorporate valuable cross-system data integration into tree construction. Such data could include MITRE's Common Vulnerabilities and Exposures(CVEs)[1] or First.org's Common Vulnerability Scoring System (CVSS)[2] Scores. Also, the attack rules in existing planners do not cover all modern systems – especially with the recent surge of IoT and IIoT systems. In order to develop an attack tree generator that is suitable for multiple systems, taking account of diverse attack goals, it is important to standardize the categorization of methodologies used by attacking systems to create a master attack rule set applicable across many system types and industry sectors.

III. SCALABLE, CROSS-SECTOR ATTACK TREE DESIGN

Since Shrobe, others have advanced the thought and application areas behind using classical planning for cybersecurity [4], [12], but none have focused on refining the cyber rule set so that it can be used across disparate industry sectors. We propose a standardized approach to developing attack trees guided by a common sequence of methods that attackers use to penetrate a wide range of systems. There are a considerable number of tools and frameworks available to hackers and security researchers alike. Unfortunately, none of these tools or frameworks address the full lifecycle of an attack. To accomplish this, we developed an integrated methodology that combines a number of established frameworks. This method will generate the information needed to develop attack trees that should be

[1] CVEs are documented vulnerabilities for computing systems which are submitted by security researchers to MITRE who maintains a running catalog of vulnerabilities in their database.

[2] CVSS Scores are developed by First.org and aim to evaluate the extent of threat that a given vulnerability poses to an organization.

scalable across industries and computing systems. To do this, we formulated a master attack methodology using various established frameworks for vulnerability, threat and exploit analysis that represent the anatomy of an attack's "when", "where", "what" and "how". The phasing sequence of the attack, or what we call the "when", leverages Lockheed Martin's cyber kill chain [13]. The surface area of where the attack could occur, or what we call the "where", references the Open Web Application Security Project's (OWASP) attack surface areas [24]. The actions required to successfully accomplish the given phase of attack, or the "what", is represented by both MITRE's Common Attack Pattern Enumeration and Classifications (CAPEC) [20] and MITRE's Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) framework [19]. Finally, the tools used to execute the actions, or the "how", are represented by both Kali Linux tools [14] and known exploit tactics by MITRE's ATT&CK Matrix [19]. Each framework occupies a level in the traditional attack tree format as seen in Fig. 3.

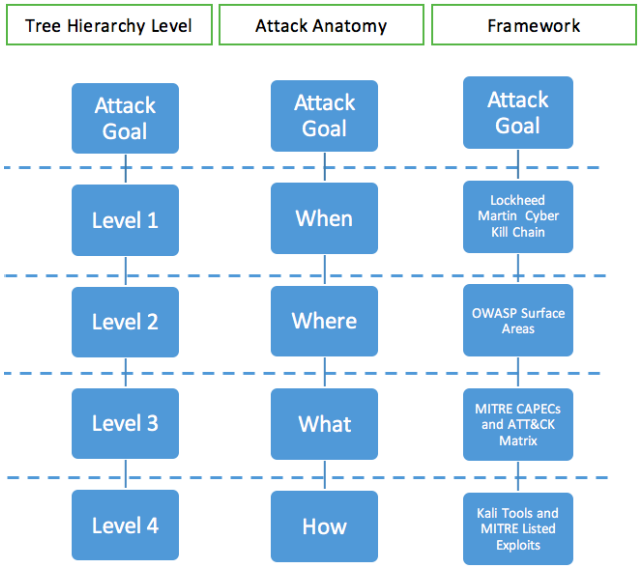


FIGURE 3. Attack tree framework mapping.

A. SEQUENCE OF PHASES FOR WAGING ATTACKS

Lockheed Martin originally developed the Cyber Kill Chain which lists the phases of a possible attack. The Cyber Kill Chain was initially published in 2011 and was developed to help security researchers map how attackers executed advanced persistent threats (APTs), including sophisticated cyberattacks conducted by nation states. The Cyber Kill Chain was inspired by the U.S. military's kill chain for traditional warfare which involved the steps required to "target and engage an adversary to create desired effects" [13]. After being created by Lockheed Martin, MITRE rebranded these steps as the Cyber Attack Lifecycle [21]. The Cyber Attack Lifecycle can be found in Fig. 4 and its associated description can be found in Table 1. The Cyber Attack Lifecycle and Cyber Kill Chain are interchangeable for the purpose of our study.

TABLE 1. Industry perspective on cyber resiliency – lifecycle for executives [21].

#	Phase	Description
1	Reconnaissance	Adversary develops a target
2	Weaponize	Attack is put in a form to be executed on the victim's computer/network
3	Deliver	Means by which the vulnerability is delivered to the target
4	Exploit	Initial attack on target is executed
5	Control	Mechanisms are employed to manage the initial victims
6	Execute	Leveraging numerous techniques, the adversary executes the plan
7	Maintain	Long-term access is achieved

While the phases indicate the order in which an attack is waged, these phases are not always performed in sequence. Much depends on the attacker's goal. For example, it is possible to skip the Weaponize, Deliver and Exploit phases of an attack, if during Recon, credentials are discovered which offers Control. Further, throughout an attack, it is likely that an attacker iterates previous phases of the lifecycle to continue gathering information about their target and refining their attack.

While developing an attack tree, each phase belongs at Level 1 of the tree hierarchy underneath the goal as seen in Fig. 3. Depending on the goal, some phases will be needed while others will not. All phases should contain AND gates indicating that each phase listed must happen and involve of its own branch of operators. The kill chain phases should consistently fall directly underneath the goal and be the top-level nodes for all attack trees. For example, if the goal of the tree was to delete data (which would fall at the top of the tree), immediately underneath should be the various phases of the kill chain.

Lockheed Martin's cyber kill chain has been represented as part of an attack tree in previous literature [29]. To date, attack trees that reference the kill chain move through phases of an attack for a given goal within a single branch as can be seen in Fig. 5. However, this inaccurately reflects the depth of each attack phase that an attacker might move through to reach their goal. Instead of having a single branch where each level in the hierarchy represents a new phase of attack and a subsequent phase is a leaf of its precedent, each phase of attack could be a separate branch connected by AND gates in the second layer of the attack hierarchy. This new representation would better illustrate the depth of complexity behind each phase of an attack by showing that there are many subroutines required to complete each phase. Also, this would lead to a more consistent approach to specifying an attack tree, making it easier to automate the generation of attack

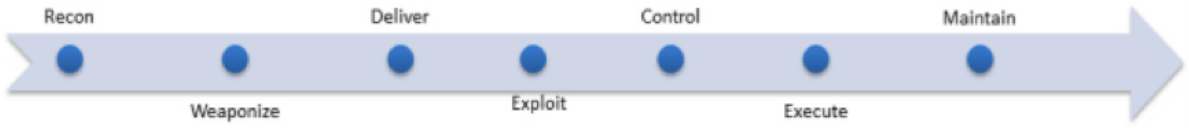


FIGURE 4. MITRE's cyber attack lifecycle.

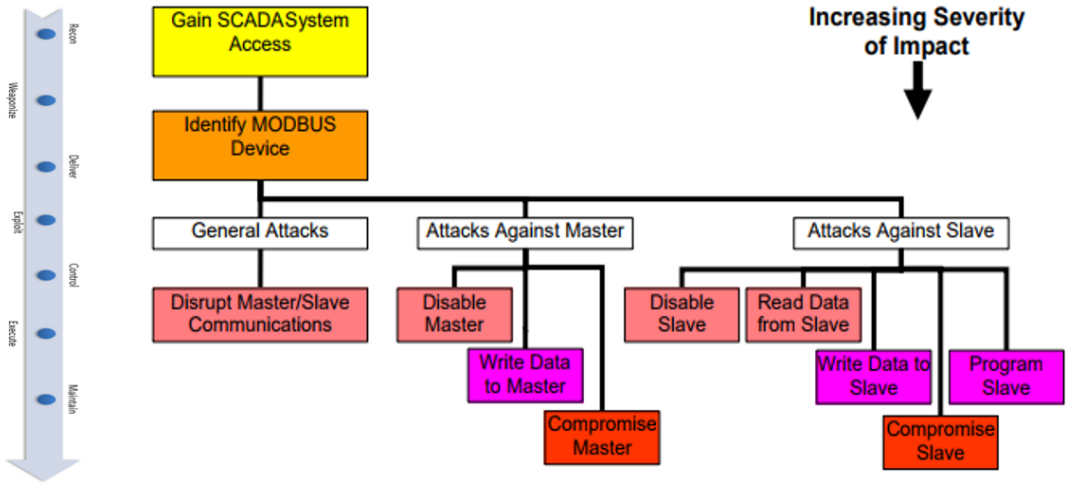


FIGURE 5. Example attack tree with kill chain for each branch [5].

trees across systems. The details behind each phase of an attack cannot be fully described in a consistent and scalable manner when enumerated as nested leaves within a single branch.

B. SURFACE AREA FOR WAGING AN ATTACK

Each phase of an attack must occur on a given surface area of a system environment. While an attack goal might need to move through all phases of the kill chain to be successful, it would be unlikely for any attack goal to involve a tree that covers all surface areas of a given system. A limited surface area is more likely to be required to achieve a given kill chain phase relevant to a specific attack goal. For example, an attack goal of ''exfiltrate server data from a system'' probably does not require recon on every surface area of a given system. It only requires recon on relevant system components.

OWASP has developed a list of seventeen surface areas for IoT systems [24]. Over the course of writing this paper, OWASP added to its list of surface areas. The list will continue to evolve over time as more threats are discovered and documented by security researchers involved in the Open Web Application Security Project. In the interest of attempting to future-proof the proposed surface areas from further edits, we distilled them into four categories: software/hardware, architecture, network and organizational. Software/hardware relates to the physical or digital features and functions of a system, architecture refers to design decisions and system configuration, network includes anything involving communications, and organizational refers to how the system is managed and any security policies in effect. These surface areas and their component parts are listed in Table 2 below.

The column labeled ''Vulnerability Examples'' describes the types of vulnerabilities likely to be associated with a surface area category. Further, the vulnerability examples listed for a surface area describe the types of vulnerabilities likely to be taken advantage of along a given surface area. (The vulnerabilities listed here are not used as part of our attack tree; rather, they are for descriptive purposes only.)

Each attack surface category will make up a different nested branch on the attack tree. For the kill chain phases of Recon, Weaponize, Deliver and Exploit, any surface area may be relevant. However, as the tree is populated towards the latter half of the kill chain under the phases control, execute and maintain, the surface areas that an attacker is seeking to act on will be a subset of those from the previous phases. For example, if Recon is only conducted on Network and Software/Hardware surface areas, Maintain would not include the surface areas of Architecture or Organization.

C. ACTIONS REQUIRED FOR WAGING AN ATTACK

Level 3 of the attack tree must represent ''what'' actions need to be performed during each phase of the attack on the given surface area. For the pre-attack phases consisting

TABLE 2. IoT surface areas and associated category [24].

Category	Attack Surface	Vulnerability Examples
Organizational	**Ecosystem**	Interoperability standards, Data governance, System wide failure, Individual stakeholder risks, Implicit trust between components, Enrollment security, Decommissioning system, Lost access procedures
Software/Hardware	**Device Memory**	Sensitive data, Cleartext usernames, Cleartext passwords, Third-party credentials, Encryption keys
Architecture	**Device Physical Interfaces**	Firmware extraction, User CLI, Admin CLI, Privilege escalation, Reset to insecure state, Removal of storage media, Tamper resistance, Debug port, Device ID/Serial number exposure
Architecture	**Device Web Interface**	Standard set of web application vulnerabilities, Credential management vulnerabilities
Software/Hardware	**Device Firmware**	Sensitive data exposure, Firmware version display and/or last update date, Vulnerable services (web, ssh, tftp, etc.), Security related function API exposure, Firmware downgrade possibility
Network	**Device Network Services**	Information disclosure, User CLI, Administrative CLI, Injection, Denial of Service, Unencrypted Services, Poorly implemented encryption, Test/Development Services, Buffer Overflow, UPnP, Vulnerable UDP Services, DoS, Device Firmware OTA update block, Firmware loaded over insecure channel (no TLS), Replay attack, Lack of payload verification, Lack of message integrity check, Credential management vulnerabilities, Insecure password recovery mechanism
Architecture	**Administrative Interface**	Standard set of web application vulnerabilities, Credential management vulnerabilities, Security/encryption options, Logging options, Two-factor authentication, Check for insecure direct object references, Inability to wipe device
Organizational	**Local Data Storage**	Unencrypted data, Data encrypted with discovered keys, Lack of data integrity checks, Use of static same enc/dec key
Architecture	**Cloud Web Interface**	Standard set of web application vulnerabilities, Credential management vulnerabilities, Transport encryption, Two-factor authentication
Organizational	**Third-party Backend APIs**	Unencrypted PII sent, Encrypted PII sent, Device information leaked, Location leaked
Architecture	**Update Mechanism**	Update sent without encryption, Updates not signed, Update location writable, Update verification, Update authentication, Malicious update, Missing update mechanism, No manual update mechanism
Architecture	**Mobile Application**	Implicitly trusted by device or cloud, Username enumeration, Account lockout, Known default credentials, Weak passwords, Insecure data storage, Transport encryption, Insecure password recovery mechanism, Two-factor authentication
Organizational	**Vendor Backend APIs**	Inherent trust of cloud or mobile application, Weak authentication, Weak access controls, Injection attacks, Hidden services
Network	**Ecosystem Communication**	Health checks, Heartbeats, Ecosystem commands, Deprovisioning, Pushing updates
Network	**Network Traffic**	LAN, LAN to Internet, Short range, Non-standard,Wireless (WiFi, Z-wave, XBee, Zigbee, Bluetooth, LoRA), Protocol fuzzing
Architecture	**Authentication/Authorization**	Authentication/Authorization related values (session key, token, cookie, etc.) disclosure, Reusing of session key, token, etc. Device to device authentication, Device to mobile Application authentication, Device to cloud system authentication, Mobile application to cloud system authentication, Web application to cloud system authentication, Lack of dynamic authentication
Organizational	**Privacy**	User data disclosure, User/device location disclosure, Differential privacy
Software/Hardware	**Hardware (Sensors)**	Sensing Environment Manipulation, Tampering (Physically), Damage (Physical)

of Recon, Weaponize, Deliver and Exploit, we primarily use MITRE's Common Attack Pattern Enumeration and Classification (CAPECs) to populate the "what" level of the tree hierarchy. Specifically, CAPECs are used for the Recon and Exploit phases of an attack. The mapping of the CAPEC mechanism of attack and the phases of attack can be seen in Table 3.

The Weaponize and Deliver phases of attack do not match any given CAPEC. Instead, we use the Lockheed Martin kill chain recommendations for this hierarchy level to note "what" is being Weaponized and "what" is being Delivered. The Lockheed Martin kill chain was used instead of MITRE's recently released Pre-ATT&CK Matrix because there are no details concerning "what" is being Weaponized or

TABLE 3. Mapping for ATT&CK matrix to cyber kill chain.

Attack Phases	Recon	Weaponize	Deliver	Exploit	Control	Execute	Maintain
CAPEC	• Collect and Analyze Information			• Inject Unexpected Items • Engage in Deceptive Interactions • Manipulate Timing and State • Abuse Existing Functionality • Employ Probabilistic Techniques • Subvert Access Control • Manipulate Data Structures • Manipulate System Resources			
Lockheed Martin		• Client applications	• Email • Websites • Removable media				
ATT&CK Matrix					• Command and control • Credential access • Privilege escalation • Discovery • Lateral movement	• Execution • Collection	• Defense evasion • Escalation • Persistence

delivered in the first release of MITRE's Pre-ATT&CK Matrix.

For control, execute and maintain, MITRE has developed the ATT&CK matrix that describes the branch level of detail below these phases. We have mapped the ATT&CK matrix categories to specific phases of the kill chain as can be seen in Table 3. We use these categories to populate level 3. Note that Table 3 is only meant to reflect the top-level domain of the mapping, and does not represent the depth contained within each domain which is described later.

D. TOOLS NEEDED FOR WAGING AN ATTACK

The final layer of the attack tree, hierarchy level 4, represents "how" an attack is likely to be carried out. Here, the weakness or vulnerability of the system, represented as a Common Weakness Enumeration (CWE)[3] or Common Vulnerabilities and Exposures (CVE) can be listed along with the tools and malware associated with each attack component. CWEs or CVEs may not always be available, but if included in the system model, they should be noted as leaves (since theoretically, each tool or malware takes advantage of a given CWE or CVE). To determine the tools and malware, for the pre-attack phases of recon and exploit, we assigned each CAPEC a tool from Kali Linux, a comprehensive penetration testing toolkit which contains most of the popular technical tools used by hackers to compromise systems. We applied a semi-automated procedure to match Kali tools to a given CAPEC using a document-distance algorithm with manual corrections. This mapping is used to populate Level 4 of the tree hierarchy describing how recon and exploit is ultimately carried out. The other pre-attack phases of deliver and exploit do not have Level 4 components due to the nature of the phase. For the attack phases control, execute and maintain, the Level 4 nodes of the tree consist of various malware that are described within the respective ATT&CK matrix categories.

[3]CWEs are categories of different types of vulnerabilities or CVEs that are created and maintained by MITRE.

E. DESIGN SUMMARY

To develop an attack tree generator that can scale and be equally effective across multiple critical infrastructure sectors, a comprehensive attacker methodology framework must populate the master attack rule set. Leveraging frameworks from respected cybersecurity authorities including MITRE, Lockheed Martin and Offensive Security (the Kali Linux creators), we have compiled a master ontology database that accounts for virtually all known attack vectors across all system types. An abstracted and complete version of this framework can be found in Fig. 6 with an example goal (exfiltrate data), which describes the various permutations

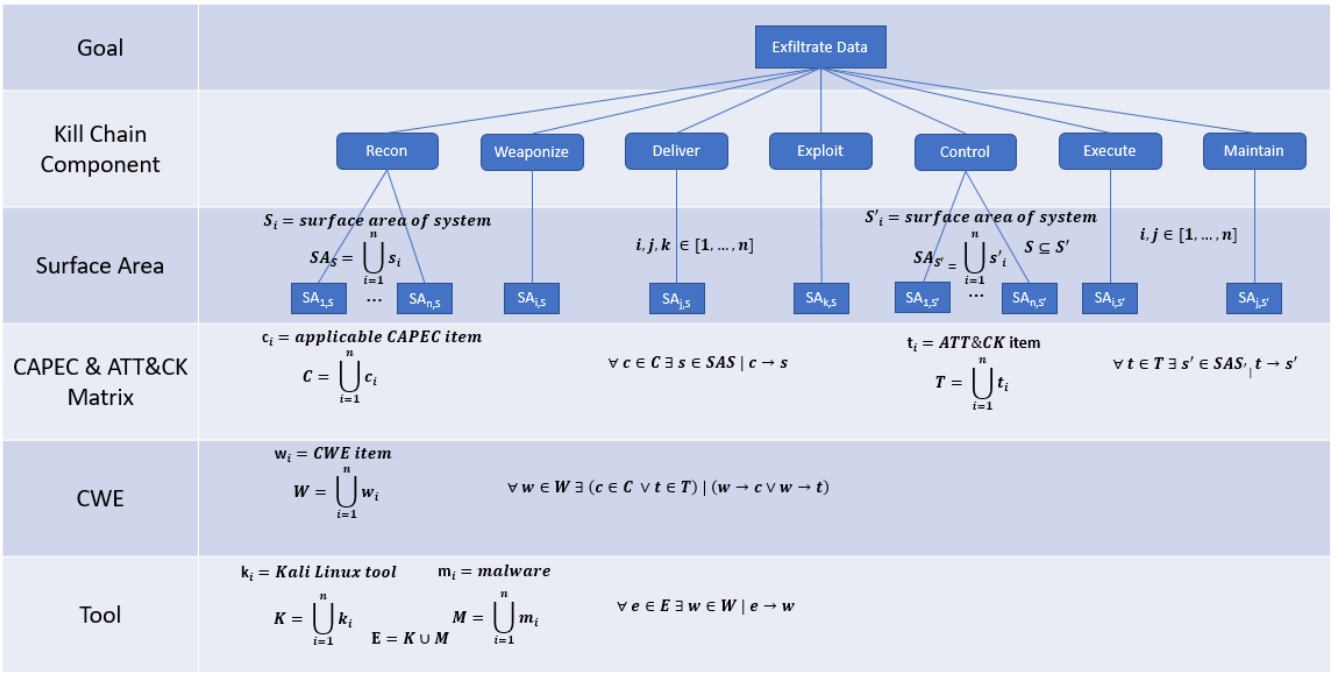


FIGURE 6. Master attack methodology.

possible for each constituent framework that makes up the master attack methodology. The attack goal of exfiltrating data as seen in Fig. 6 is only one example of goals for which the methodology is relevant.

IV. EXAMPLE USE CASE

To demonstrate the core concepts in this methodology that we hope can be used for automatically generating an attack tree for numerous critical infrastructure sectors, we have created a test case for a generic CCTV urban surveillance system. After developing an automated tree using the master attack method, we create an attack tree for the same system by hand. The manually created attack tree is then compared to the automated attack tree using our proposed standardized methodology to demonstrate some perceived benefits of the proposed method.

A. AUTOMATED ATTACK TREE WITH MASTER ATTACK METHODOLOGY

1) SYSTEM MODEL

First, we developed a system model of the CCTV network based on our interpretation of essential elements of the network typology found in Fig. 7. This CCTV network typology was selected because it has representative components included in most IP networks (i.e. storage servers, analytics engines, peripheral devices/sensors, command modules, visualization components, etc.).

For purposes of abstraction and simplification, our system model focuses on several key areas of the CCTV system circled in red. We selected these specific areas because we believe them to be essential to the operation of the CCTV system. These included the IP surveillance camera, the storage device, the video surveillance manager (VSM), the console server (CM), the display server (DP), the video processing server (MD), the LAN and its associated router and switch. We developed the system description using assumptions about the interactions and functions of these system components. These assumptions about the system description are documented in Table 4.

B. ATTACK RULE SET

For purposes of this example, we selected an attacker goal of exfiltrating data from the IP camera. Based on the system environment and the master attack framework, attack rules were selected to develop the branches of the tree. The rules that were applied for the first branch of the tree (recon) for this attack goal are reflected in Fig. 8 and represented in a nested tree structure along with their respective AND/OR gates. The ellipses represent further branches and leaves of the tree which are omitted for purposes of brevity. It is important to note that based on the system model and selected goal, not all level 2 surface areas are applicable at every attack phase and not all CAPECs and ATT&CK level 3 ''what'' categories are applicable to every surface area. Further, if there is no CWE or CVE available for level 4, the generator will skip this and jump directly to the Tools/Malware.

C. MANUAL ATTACK TREE

For comparative purposes, we asked several security researchers to hand-draw attack trees to demonstrate the differences for the same CCTV system.

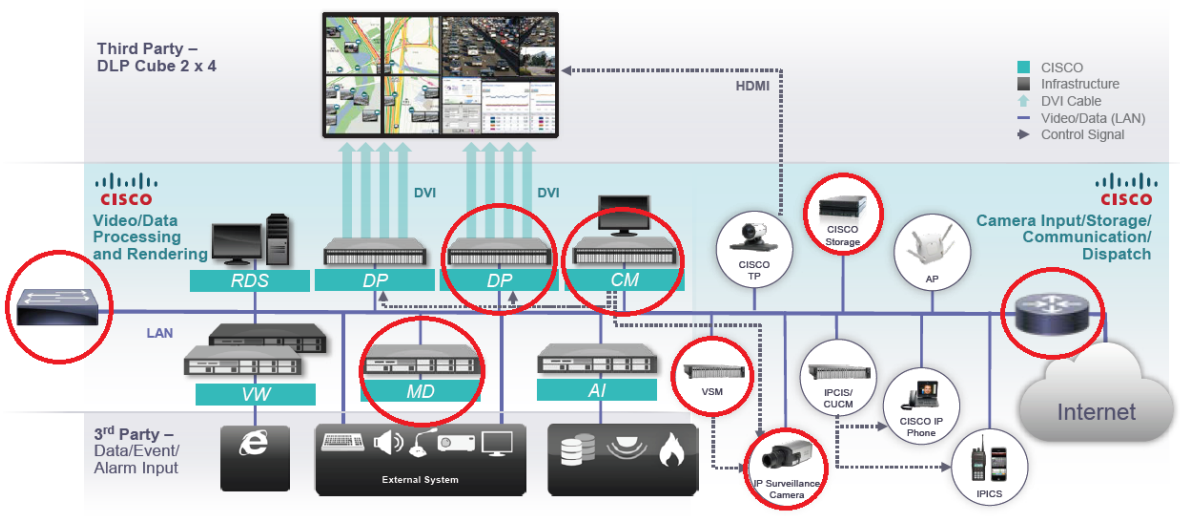


FIGURE 7. Example CCTV network topology [8].

TABLE 4. Description of pertinent network topology features [8].

System Component	Description
IP Surveillance Camera	• Multiple edge devices are networked together • Centrally managed • Edge devices are running an embedded Linux OS • Have only admin privilege level
Storage Server	• Edge devices have limited data storage capacities • After 8 hours of video recording data is sent from IP camera to storage server
Video Surveillance Manager (VSM)	• All edge devices are centrally managed by the video surveillance manager • VSM has admin privileges across all IP cameras and storage device
Console Server (CM)	• Windows OS • Supervisory control across all servers and the IP camera network
Display Server (DP)	• Renders data from storage server • Transmits video data to third party display
Video Processing Server (MD)	• Processes data for interoperable analysis across different peripherals such as occupancy sensors • Enables processing for eiganfaces so that facial recognition can be performed via computer vision as well as other needed computations that can interact with alarms
LAN, Router & Switch	• Bus and hardware that enables communication across devices

Fig. 9 shows a representative manually created tree. To successfully create this tree, the security researchers not only needed system knowledge of the CCTV network, but also an understanding of possible attack patterns relevant to the CCTV system.

D. COMPARATIVE DISCUSSION

Perhaps the most important difference between the automated tree and the manual tree was the time required to create each one. The manual tree required time to research and map the system environment and possible attack vectors – this took an average of an hour for each security researcher to complete the manual tree, while the automated tree was completed in minutes by loading the model-based system description into the planner. The manual tree does not incorporate standardized language from various established cybersecurity frameworks. This is problematic because if two different security researchers tried to enumerate all attack pathways,

Goal: Exfiltrate data from IP camera
If the goal is to Exfiltrate Data
then you have to do
AND
Recon
If the goal is to do Recon
then you have to do Recon on
AND
Network
AND
Device Network Services
If the goal is to do Recon on Device Network Services
then you have to do
OR
Fingerprinting
AND
If the goal is to fingerprint network services
then exploit “Information Exposure” weakness (CWE-200)
If the goal is to exploit the "Information Exposure” weakness
then use nmap
Protocol Analysis..
Footprinting…
Ecosystem Communication…
Network Traffic…
Software/Hardware…
Architecture…
Organization…
Weaponize….
Deliver…
Exploit…
Control…
Execute…
Maintain….

FIGURE 8. Automatically generated attack tree for data exfiltration.

Goal: Exfiltrate data from IP camera
If the goal is to Exfiltrate Data
then you have to
AND
Find the Data
If the goal is to find the data
then you have to
OR
Find the Local Camera
If the goal is to find the data on the local camera
then you need to
AND
Access the camera
OR
Steal Password
Exploit Vulnerability
Find the Hosting Server
If the goal is to find the data on the hosting server
then you need to
AND
Access the hosting server
OR
Steal Password
Exploit Vulnerability
Steal the Data

FIGURE 9. Hand-drawn attack tree for data exfiltration.

there might be discrepancies between the trees. Of course, time is required to build a model-based system description of a computing environment, but our assumption is that system models may be readily available for use considering they are important for a variety of testing purposes. Standardization also makes it easier to train non-experts or semi-experts in generating attack trees without requiring detailed cybersecurity domain knowledge.

The automated master attack method of tree construction contains references to specific CAPECs which often connect back to specific CWEs and CVEs, while the manual tree lacks this information. This provides actionable insight to operators so they know which security patches are most important. The automated tree can scale for massive systems based on the details provided in the system description, while by hand it would not be possible to account for all vectors in an extremely complex environment without consuming considerable resources. An automated approach will also greatly reduce incomplete and incorrect attack trees caused by oversight and errors in enumerating the attack surface in complex systems.

Perhaps the most important difference between the two is that the automated tree has considerably more depth and information than the manual tree because it moves through each phase of the Cyber Kill Chain. This generates step-by-step insight into how the attack would probably be carried out compared to a less detailed hand-drawn tree.

The comprehensive nature of the automated tree provides more insight to a defender on the risks of the system.

Traditionally, the domain knowledge or the thinking required to create unique attack vectors has resided with individual experts and has never been shared. Our automated approach enables us to capture domain knowledge from multiple experts for future reuse.

Automated attack tree generation using the proposed method shifts the burden from creating individual attack trees to system specification. Experts spend more time understanding and specifying system components, interfaces and interconnections. This is advantageous once system assumptions are explicitly coded; it should result in more consistent and robust attack trees for a variety of industries and types of computing systems. With the manual approach, all system information and assumptions reside with the expert, making it difficult to validate the experts' understanding of the system.

V. RESEARCH IMPLICATIONS

Cyberattacks against public infrastructure systems can be extremely costly as seen in the aftermath of the Atlanta ransomware attack that is now expected to cost upwards of $12 Million [16]. This proposed master attack methodology deployed for an AI planning system can be useful for public administrators to understand cyber risks for their smart cities. By using an attack planner auditing tool to evaluate smart city digital asset risk, defensive measures can be taken to mitigate the potential cyberattacks and their associated financial damages.

Many publicly available security tools that are intended for good can also be used by malicious actors. The master attack planner proposed would be no different. Potential attackers ranging from novice script kiddies to nation states seeking to wage advanced persistent threats against a smart city can leverage this tool to plan out their attack. The tool can theoretically help them to determine their most effective and efficient attack options to penetrate and disrupt city operations. We can take solace knowing that the master attack methodology alone will not sufficiently help a malicious actor plot a cyberattack against a smart city. To effectively plan an attack, a hacker would need access to a model of the target smart city system. The master attack methodology must be applied to a system model to be an effective tool.

Gaining access to a smart city system model is far from an impossibility. An insider threat is a significant concern and likely mechanism for a hacker to procure a smart city system model. Smart city operators and IT personnel may have considerable access to digital asset information including system architectures, network topologies and even current system vulnerabilities and patching schedules. The scope of insiders expands beyond system operators and IT personnel for smart cities because of the wide range of users of smart city technologies. Because many smart city IoT systems are open to the public for access, the surface area of insider threat attack becomes significantly larger [23]. One of the most well-known insider attacks against smart city systems was a cyberattack against the Maroochy Shire sewage treatment plant in Queensland, Australia which resulted in tons of sewage being spilled into the municipality [33]. Attackers do not need to be insiders to procure system model information. Social engineering can be effectively used against citizen insiders with access to smart city infrastructure to surreptitiously collect necessary information to wage an attack [1], [18], [34].

The main benefits of providing public administrators a cyber auditing tool that can be used across IIoT sectors is that it can considerably increase visibility to smart city cyber risks and do so quickly and accurately. These cyber risks can then be addressed. We would argue that these benefits outweigh the risk of malicious actors using the master attack methodology planner to plot an attack against a smart city. A potential attacker can and will wage an attack regardless of the availability of our planner. Withholding our planner from the public would be more of a disservice to public administrators than attackers.

VI. FUTURE WORK

To date, this research has involved designing a method for a scalable master attack method that can be used with automated attack generators. The proposed method has only been tested on a single critical infrastructure system environment – the example CCTV network shown above. Therefore, we cannot yet conclude that it can effectively scale across industries. The next phase of our research will aim to demonstrate that the master attack method we have proposed can be applied across multiple critical infrastructure sectors involving different system models and computing systems. A persistent challenge, though, is getting access to data for system environments due to their often proprietary or sensitive nature. We are in active discussions with NASA's Jet Propulsion Laboratory to collaborate on testing this method across some of their mission system environments. The goal will be to determine if the proposed design method can be broadly applied and scaled across systems and sectors. Other future work will involve refining and updating the categories in the frameworks maintained by multiple organizations.

Beyond testing the proposed scalable master attack method, we plan to incorporate logic into the attack planner that can help account for ''chance causes'' as described by Shewhart and Andrew [31]. This can possibly be accomplished by altering the planning algorithm. Instead of using directed backward search, we will test a Monte Carlo planning algorithm as outlined by Nakhost *et al.* [22] which may help to identify unexpected attack patterns.

VII. CONCLUSION

As critical infrastructure continues to be subject to cyberattacks, defenders must remain vigilant and evaluate all possible attack vectors involving multiple systems. While attack trees can be used to provide guidance about possible attack vectors that defenders need to protect, there are operational challenges associated with developing such

attack trees. Automating the attack tree generation process using AI planning can ease this operational burden.

The design of automated attack trees to date has not generated a comprehensive attack rule set that can be used across disparate critical infrastructure sectors. By combining attack frameworks from a number of respected authorities, we have developed a master attack method that can be used with classical planners to generate automated attack trees. We hope that this new approach, with further testing and refinement, will be useful across multiple critical infrastructure sectors, thereby easing the operational burden of cyber risk enumeration.

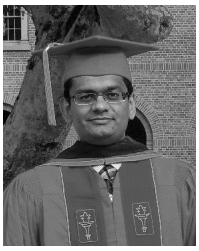
ACKNOWLEDGEMENT

The authors would like to thank members of the Cyber Defense Engineering and Research Group at the Jet Propulsion Laboratory for discussions and feedback on the ideas expressed in the paper.

REFERENCES

[1] P.-E. Arduin. *Insider Threats*. Hoboken, NJ, USA: Wiley, 2018.
[2] Atlanta, GA, USA. (2018). *Ransomware Cyberattack Information*. Accessed: Jun. 2018. [Online]. Available: https://www.atlantaga.gov/government/ransomware-cyberattack-information
[3] S. Bistarelli, M. Dall'Aglio, and P. Peretti, "Strategic games on defense trees," in *Proc. Int. Workshop Formal Aspects Secur. Trust*. Berlin, Germany: Springer, 2006.
[4] M. Boddy, J. Gohde, T. Haigh, and S. Harp, "Course of action generation for cyber security using classical planning," in *Proc. ICAPS*, 2005, pp. 12–21.
[5] E. J. Byres, M. Franz, and D. Miller, "The use of attack trees in assessing vulnerabilities in SCADA systems," in *Proc. Int. Infrastruct. Survivability Workshop*, 2004, pp. 3–10.
[6] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: Risk assessment, detection, and response," in *Proc. 6th ACM Symp. Inf., Comput. Commun. Secur.*, 2011, pp. 355–366.
[7] T. Chuang. (2018). SamSam ransomware virus keeps CDOT employees offline for fourth day. DenverPost. Accessed: Jun. 2018. [Online]. Available: https://www.denverpost.com/2018/02/26/samsam-ransomware-virus-cdot/
[8] *Cisco Smart+Connected City Operations Center: Unified Management for City Infrastructure*, Cisco, San Jose, CA, USA, 2014. Accessed: Nov. 3, 2017.
[9] T. W. DeLong, "A fault tree manual," Intern Training Center, Army Materiel Command, Texarkana, TX, USA, Tech. Rep. 1, 1970.
[10] D. Evans, "The Internet of everything: How more relevant and valuable connections will change the world," Cisco, San Jose, CA, USA, Tech. Rep. 1, 2012, pp. 1–9.
[11] M. Gjendemsjø, "Creating a weapon of mass disruption: Attacking programmable logic controllers," M.S. thesis, Norwegian Univ. Sci. Technol., Trondheim, Norway, 2013. Accessed: Dec. 13, 2015.
[12] J. Hoffmann, "Simulated penetration testing: From 'Dijkstra' to 'turing test++,'" in *Proc. ICAPS*, 2015, pp. 364–372.
[13] E. Hutchins, M. Cloppert, and R. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues Inf. Warfare Secur. Res.*, vol. 1, no. 1, p. 80, 2011.
[14] Kali. (2017). *Kali Linux Tools Listing—Penetration Testing Tools*. Accessed: Sep. 15, 2017. [Online]. Available: https://tools.kali.org/tools-listing
[15] Kaspersky Lab ICS CERT. (2017). *Threat Landscape for Industrial Automation Systems in the Second Half of 2016*. Accessed: Mar. 31, 2017. [Online]. Available: https://ics-cert.kaspersky.com/reports/2017/03/28/threat-landscape-for-industrial-automation-systemsin-the-second-half-of-2016/
[16] L. Kearney. (2018). *Atlanta Officials Reveal Worsening Effects of Cyber Attack*. Accessed: Jun. 27, 2018. [Online]. Available: https://www.reuters.com/article/us-usa-cyber-atlanta-budget/atlanta-officials-reveal-worsening-effectsof-cyber-attack-idUSKCN1J231M
[17] T. McGalliard. (2018). Opinion | How local governments can prevent cyberattacks. NYTimes. Accessed: Jun. 27, 2018. [Online]. Available: https://www.nytimes.com/2018/03/30/opinion/local-government-cyberattack.html
[18] K. D. Mitnick and W. L. Simon, *The Art of Deception: Controlling the Human Element of Security*. Hoboken, NJ, USA: Wiley, 2003.
[19] MITRE. (2017). *ATT&CK Matrix for Enterprise*. Accessed: Sep. 15, 2017. [Online]. Available: https://attack.mitre.org/wiki/ATT&CK_Matrix
[20] MITRE. (2017). *CAPEC—Common Attack Pattern Enumeration and Classification (CAPEC)*. Accessed: Sep. 14, 2017. [Online]. Available: https://capec.mitre.org/
[21] MITRE. (2017). *Industry Perspective on Cyber Resiliency—Lifecycle for Executives*. Accessed: Nov. 3, 2017. [Online]. Available: http://www2.mitre.org/public/industry-perspective/lifecycle.html
[22] H. Nakhost and M. Müller, "Monte-Carlo exploration for deterministic planning," in *Proc. IJCAI*, 2009, pp. 1766–1771.
[23] J. R. C. Nurse, A. Erola, I. Agrafiotis, M. Goldsmith, and S. Creese, "Smart insiders: Exploring the threat from insiders using the Internet-of-Things," in *Proc. Int. Workshop Secure Internet Things (SIoT)*, Sep. 2015, pp. 5–14.
[24] OWASP. (2017). *IoT Attack Surface Areas*. Accessed: Sep. 7, 2017. [Online]. Available: https://www.owasp.org/index.php/IoT_Attack_Surface_Areas
[25] P. A. S. Ralston, J. H. Graham, and J. L. Hieb, "Cyber security risk assessment for SCADA and DCS networks," *ISA Trans.*, vol. 46, no. 4, pp. 583–594, 2007.
[26] R. Ramirez and N. Choucri, "Improving interdisciplinary communication with standardized cyber security terminology: A literature review," *IEEE Access*, vol. 4, pp. 2216–2243, 2016.
[27] R. Arpan, D. S. Kim, and K. S. Trivedi, "Cyber security analysis using attack countermeasure trees," in *Proc. 6th Annu. Workshop Cyber Secur. Inf. Intell. Res.*, Oak Ridge, TN, USA, Apr. 2010, Art. no. 28.
[28] H. M. Salim, "Cyber safety: A systems thinking and systems theory approach to managing cyber security risks," Ph.D. dissertation, Compos. Inf. Syst. Lab., Massachusetts Inst. Technol., Cambridge, MA, USA, 2014.
[29] C. Sarraute. (2013). "Automated attack planning." [Online]. Available: https://arxiv.org/abs/1307.7808
[30] B. Schneier, "Attack trees," *Dr. Dobb's J.*, vol. 24, no. 12, pp. 21–29, 1999.
[31] W. A. Shewhart, *Economic Control of Quality of Manufactured Product*. Milwaukee, WI, USA: ASQ Quality Press, 1931.
[32] H. E. Shrobe, "Computational vulnerability analysis for information survivability," *AI Mag.*, vol. 23, no. 4, pp. 81–94, 2002.
[33] J. Slay and M. Miller, "Lessons learned from the Maroochy water breach," in *Proc. Int. Conf. Crit. Infrastruct. Protection*. Boston, MA, USA: Springer, 2007, pp. 73–82.
[34] R. Willison and M. Warkentin, "Beyond deterrence: An expanded view of employee computer abuse," *MIS Quart.*, vol. 37, no. 1, pp. 1–20, 2013.
[35] M. Wright. (2018). A ransomware attack brought Atlanta to its knees—And no one seems to care. TheHill. Accessed: Jun. 27, 2018. [Online]. Available: http://thehill.com/opinion/cybersecurity/381594-a-ransomware-attack-broughtatlanta-to-its-knees-and-no-one-seems-to

GREGORY FALCO received the B.S. degree from Cornell University, the M.S. degree from Columbia University, and the Ph.D. degree in cybersecurity from the Massachusetts Institute of Technology (MIT). At the Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT, he collaborates with the Jet Propulsion Laboratory, NASA, to design and build artificial Intelligence tools that can be used to secure critical infrastructure and space mission systems. He is currently a Security Researcher and a Post-Doctoral Scholar with CSAIL, MIT. He is also a Post-Doctoral Scholar with Stanford University, where he researches cyber risk and cyber insurance. He is the Co-Founder and the CEO of NeuroMesh, a security company that uses the blockchain to secure IIoT embedded systems.



ARUN VISWANATHAN received the M.S. and Ph.D. degrees in computer science from the University of Southern California, where he explored model-driven approaches for situational awareness, and involved in addressing cyber security challenges in the smart grid. He is currently a Cyber Security Researcher with the Cyber Defense Engineering and Research Group, Jet Propulsion Laboratory, California Institute of Technology. He is currently involved in the intersection of cyber security and artificial intelligence, with a goal to build intelligent systems that can automatically detect, diagnose, and recover from cyber-attacks. He is currently investigating the application of advanced machine learning methods for detecting cyber intrusions, along with leading the research and development of tools aimed at improving cyber situational awareness in mission critical systems.

CARLOS CALDERA received the B.Sc. degree in computer science and electrical engineering from MIT and the M.E. degree in computer science and electrical engineering from MIT, with a focus on computer systems. He is currently a Software Engineer with Oracle and a Research Affiliate with MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL).

HOWARD SHROBE has served as the Assistant Director of DARPA, and the Chief Scientist and the Program Manager of the Information Technology Office. He is currently a Principal Research Scientist with the Computer Science and Artificial Intelligence Laboratory, MIT. He is also the Director of CyberSecurity@CSAIL.