

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/317830136>

Markov Chain modeling of cyber threats

Article in *The Journal of Defense Modeling and Simulation Applications Methodology Technology* · July 2017

DOI: 10.1177/1548512916683451

CITATIONS

16

READS

575

3 authors:



Ross Gore

Old Dominion University

84 PUBLICATIONS 1,185 CITATIONS

SEE PROFILE



José Julian Padilla

University of Guadalajara

108 PUBLICATIONS 1,282 CITATIONS

SEE PROFILE




Saikou Y. Diallo

Old Dominion University

162 PUBLICATIONS 2,252 CITATIONS

SEE PROFILE

Markov Chain Modeling of Cyber Threats

Journal Title
XX(X):1-10
©The Author(s) 2016
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/


Ross Gore¹, Jose Padilla¹ and Saikou Diallo¹

Abstract

Cyber security is a complex, multifaceted, poorly understood problem domain. As the use of digital technology grows, the threat environment continues to evolve dynamically. Traditional approaches for cyber security focus on understanding and addressing vulnerabilities. While this mindset is necessary it is not sufficient. A better understanding of the nature of existing and future cyber threats is needed to make informed defensive decisions that optimize the use of limited resources. Here, we address this deficiency by applying Markov Chain methods to descriptions of observed cyber threats. The goal of this effort is to identify previously unknown themes of common vulnerabilities. We present the results of our study and discuss its implications. Then we conclude and provide direction for future work.

Keywords

modeling, cyber security, markov chain

Introduction

Today's evolving cyber threat environment contains a myriad of advanced attack scenarios. Adversary behavior is no longer primarily focused on widespread, disruptive activity. Instead it is characterized by targeted, multi-stage attacks that aim to achieve specific tactical objectives and establish a persistent foothold into vulnerable enterprises.

From a defensive perspective these cyber threats reflect the *kill chain* shown in Figure 1 [1]. The adversary's attack unfolds in a series of steps, ending with the attacker having an established foothold in the vulnerable enterprise's network. Adversaries are typically assumed to be nation states and those engaged in conducting cyber crime, financial threats, industrial espionage and terrorism [2, 3].

The ability of these cyber threats to cause ongoing damage requires a more proactive approach to cyber security. Responding to incidents after an exploit is costly in terms of the effective impact and the level of effort necessary to root out the adversary's established foothold. To be proactive, cyber defenders must move towards stopping the adversary's advance before the exploit stage of the kill chain. This defensive strategy requires a move from after-the-fact incident investigation and response to one driven by sharing information about cyber threats [4, 5].

The Structured Threat Information eXpression language (STIX) facilitates this effort [6, 7, 8]. STIX is a

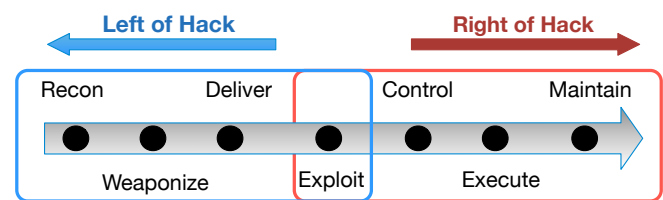


Figure 1. Cyber kill chain from left to right. Graphic reproduced from [1]

collaborative community-driven standardized language to represent structured cyber threat information. It provides a common mechanism for analyzing, specifying indicator patterns, sharing information and managing responses related to cyber threats. Numerous organizations have employed STIX to characterize observed threats and share this information with other trusted partners. This information sharing paradigm has improved the understanding of the overall landscape of cyber threats [1, 9, 10].

¹Virginia, Modeling, Analysis and Simulation Center at Old Dominion University Suffolk, VA, USA
Received: 12-18-2015; Revised: 5-23-2016; Accepted: 11-14-2016

Corresponding author:

Ross Gore, Virginia, Modeling, Analysis and Simulation Center Old Dominion University, 1030 University Blvd, Suffolk, VA, 23435, USA
Email: rgore@odu.edu

Unfortunately, even this use of information sharing is still reactive. The collection and sharing of data via STIX requires that at least one organization fall victim to a cyber threat before information can be collected. Ideally, a cyber intelligence gathering approach would exist that would enable all organizations to move *left of the hack* without any becoming exploited. Such an approach would provide understanding of the overall landscape of cyber threats, even those that had not yet been attempted by adversaries, to enable organizations to be proactive about cyber security [11?].

In this paper, we work towards this goal by taking a Markov Chain approach to modeling cyber threats. A Markov Chain is a random process that undergoes transitions from one state to another on a state space. The state transitions in a Markov Chain are *memorylessness* - the probability of the next state depends only on the current state and not on the sequence of events that preceded it. In our work we create a Markov Chain of STIX descriptions of two previously observed cyber threats. Then we generate trajectories through the Markov Chain representation to identify previously unknown themes of common vulnerabilities cyber threats can exploit.

This ability to identify previously unknown themes of common vulnerabilities enables a more proactive approach to cyber security. Analysts can study the intelligence provided by our tool to understand what common themes of vulnerabilities adversaries could exploit. Then they can create proactive defenses against them. Furthermore, because we leverage STIX descriptions, as new cyber threats are described the resulting themes uncovered by the analysis will be automatically updated. The result is an analysis methodology that will not become outdated because as new attacks are described, it accounts for the properties of those attacks in its analysis through the Markov Chain model.

The remainder of this paper proceeds as follows. First, we provide background information related to Markov Chains. Next, we describe our approach to creating and sampling a Markov Chain built from existing STIX descriptions of cyber threats. Finally, we employ our model to identify and analyze a previously unknown common vulnerability theme. We conclude our work by summarizing our contributions and providing direction for future research.

Approach

Understanding our approach to modeling documented cyber threats requires an understanding of the Structured Threat Information eXpression language (STIX) for describing

cyber threats and Markov Chain methods. Here we provide an overview of each and discuss how they work in combination to generate and analyze previously unknown common vulnerability themes.

Markov Chains

A Markov chain is a stochastic system with the Markov property. A Markov Chain moves through a series transitions among of random states. However, the Markov property only enables transitions between adjacent states. In other words, the next state of the system depends entirely on the current state. This property makes Markov chains useful for describing systems that follow a chain of linked events, where what happens next depends only on the current state of the system [12].

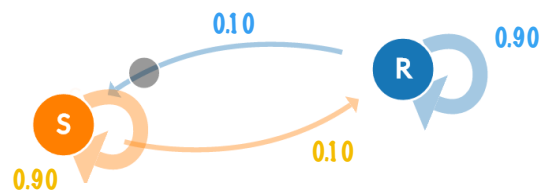


Figure 2. Markov Chain used to simulate the rainy and sunny days in April.

Rainy Days In April An example helps elucidate the structure and use of Markov Chains. Suppose, we want to determine the maximum number of rainy days *in a row* that occur in April. Based on existing weather data from past Aprils we know that, if it's sunny (S) one day, then there is a 90% chance it will be sunny the next day. Similarly, if it's rainy (R) one day, then the next day there is also a 90% chance that it will be rainy. This data enables us to construct a Markov Chain to generate rain predictions that are representative of the month of April. The Markov Chain is shown in Figure 2.

Using the Markov Chain we can generate a possible trajectory for the upcoming April. For example, one possible trajectory is: **S S S S S S S R R R S S S S S S S S R R R R R R R S S S S**. In this trajectory the maximum number of rainy days in a row are the final seven rainy days shown in bold. While this trajectory does not exactly match any previous April data that we collected it is representative of the data set as a whole. It is representative because the trajectory is generated from a Markov Chain where the probability of

transitioning between states was parameterized by the data set as a whole.

Unfortunately, a single trajectory through the Markov Chain does not enable us to answer our question of what is the maximum number of rainy days in a row in April. In order to answer this question we need to repeatedly sample the Markov Chain shown in Figure 2 to generate many possible trajectories of the upcoming April. Figure 3 shows a summary of the distribution of the maximum number of rainy days in a row from all the trajectories resulting from our sampling. There are a range of outcomes for the maximum number of rainy days in a row in April based on the past data. It is possible there could be as few as one rainy day in a row or as many as 10. However, by generating possible trajectories and summarizing them with a histogram we can see that the most likely outcome is that there will be 5 - 6 rainy days in a row in April.

The previous example is not meant to be rigorous, but to illustrate how a Markov Chain is applied to model and analyze possible future trajectories based on past data. Next, we describe our Markov Chain model of cyber threats described in the STIX language. Then we present a case study where the Markov Chain is applied to generate a previously unknown theme of common vulnerabilities cyber threats can exploit.

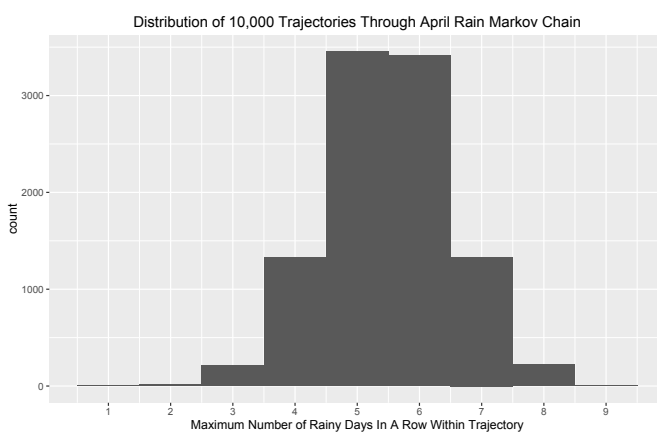


Figure 3. Histogram summarizing the distribution of the maximum number of rainy days in a row.

Markov Chain of STIX Described Cyber Threats

Recall, our goal is to: (1) create a Markov Chain model of cyber threats described in STIX language and (2) analyze possible trajectories through the Markov Chain to identify previously unknown themes of common vulnerabilities. First we describe STIX and then we discuss how we model STIX descriptions using a Markov Chain.

STIX is a collaborative community-driven effort to define and develop a standardized language to represent structured cyber threat information. It provides a common mechanism for analyzing, specifying indicator patterns, sharing information and managing responses related to cyber threats. It does this in a structured fashion to support more effective cyber threat management processes and application of automation.

Furthermore, STIX is flexible and extensible. Existing standardized languages may be leveraged as optional extensions where appropriate and numerous flexibility mechanisms are designed into the language. Finally, STIX is agnostic to any particular technology or type of cyber threat. As a result it is focused on capturing information as opposed to assuming a specific technology or requiring specific components to be included in an attack [6, 7, 8].

The process of constructing a Markov Chain to model possible STIX trajectories begins with identifying the core components of a cyber threat description in STIX. Each component in a STIX description is annotated with a set of attributes. These attributes describe the particular instantiation of the component that is employed in the cyber threat. The eight core components of STIX are:

- *Observables* - stateful properties or measurable events pertinent to the operation of computers and networks. Examples of observables include Information about a file, a registry key value, a service being started, or an HTTP request.
- *Indicators* - information about valid time windows, likely impact, sightings of the indicator, structured test mechanisms for detection, related campaigns and suggested courses of action.
- *Incidents* - discrete instances of Indicators affecting an organization along with information discovered or decided during an incident response investigation. Incidents consist of data such as time-related information, parties involved, assets affected, impact assessment, intended effects, nature of compromise, confidence in characterization and handling guidance.
- *Tactics, Techniques and Procedures (TTPs)* - specific adversary behavior (attack patterns, malware, exploits) exhibited, resources leveraged (tools, infrastructure, personas), information on the victims targeted (who, what or where), intended effects and relevant kill chain phases.
- *Campaigns* - the suspected intended effect of the adversary, the related TTPs leveraged, the related incidents, confidence in the assertion of intent and

characterization of the Campaign, and activity taken in response to the Campaign.

- **Threat Actors** - characterizations of malicious actors (or adversaries) representing a cyber attack threat including presumed intent and historically observed behavior. These characterizations include identity, suspected motivation, suspected intended effect and handling guidance.
- **Exploit Targets** - vulnerabilities or weaknesses in software, systems, networks or configurations that are targeted for exploitation. These include vulnerability identifications or characterizations, weakness identifications or characterizations and configuration identifications or characterizations.
- **Courses Of Action** - specific measures to be taken to address threat whether they are corrective or preventative to address ExploitTargets, or responsive to counter or mitigate the potential impacts of Incidents including cost and efficacy.

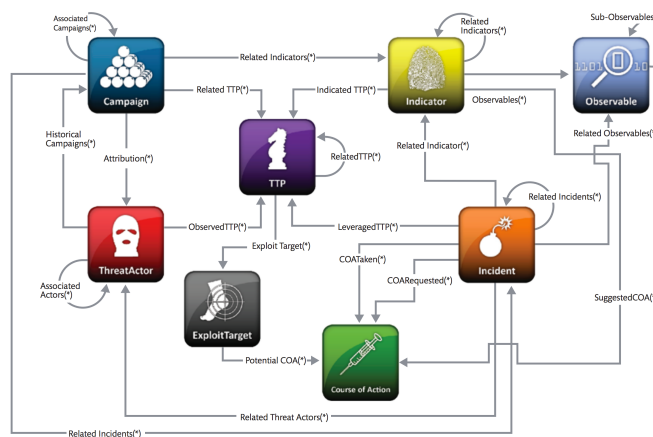


Figure 4. STIX architecture highlighting the relationships among the core cyber threat concepts in STIX. Graphic reproduced from [1].

The interactions among these eight core components is specified by STIX architecture shown in Figure 4. The figure highlights the types of relationships that can exist among these core components via the directed arrows. The bracketed asterisk on each of the arrow labels implies that each relationship may exist zero to many times [1].

The biggest difference between a description of a cyber threat in STIX and the rainy days example is that the core components in STIX can be nested inside one another. For example, each of the cyber threats shown in Figure 5 are valid STIX descriptions even though components are included within other components in each of the three descriptions.

To address this issue we create two algorithms to translate a set of STIX descriptions into a Markov Chain that can generate possible trajectories. Algorithm 1 constructs a Markov Chain Model of a cyber threat given a set of STIX descriptions. Algorithm 2 generates a possible cyber threat trajectory through the model. We describe each of these algorithms next.

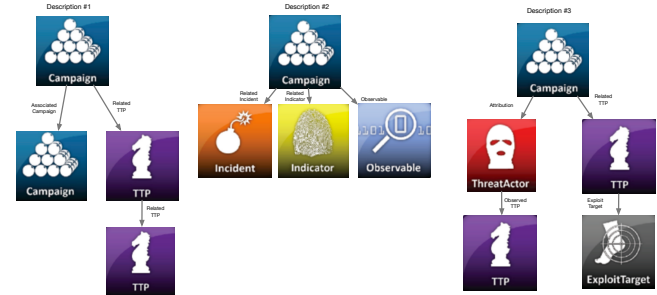


Figure 5. Three different STIX valid compositions of the STIX components.

Algorithm 1 employs the STIX architecture shown in Figure 4 as a graph G . Graph G has 8 vertices ($G.V$) and 23 edges ($G.E$). Each edge has indicators which specify the vertex the edge travels from and the vertex the edge travels to. Also each edge in G is augmented with a queue ($G.E.Q$). The algorithm takes as input the graph G and a set of STIX descriptions S . Each of the STIX descriptions in set S is a tree like those shown in Figure 5. Each tree is composed of the vertices ($s_i.V$) and edges ($s_i.E$). These vertices and edges are a subset of those in G . However, it is important to note that each vertex in $s_i.V$ is annotated with attributes to describe the particular instantiation of the component that is employed in the cyber threat being described.

Algorithm 1 traverses the vertices in each tree in S along the edges. Each time it traverses an edge in the tree it adds the vertex (with annotation) it is traveling to, to the queue for the corresponding edge in G . It also adds a NIL to the queue for each edge in G connected to the vertex it traveled from but was not traversed. Once the entire tree has been traversed, it is encoded in G by: (1) adding the vertices (with annotations) within the tree to the queues for the appropriate edges and (2) encoding those paths not taken in the tree by adding a NIL value to the queues for the appropriate edges. Algorithm 1 is specified in pseudocode below.

Algorithm 2 uses the Markov Chain constructed by Algorithm 1 to print a depth first trajectory. It takes the graph created by Algorithm 1, a starting vertex and a number indicating the maximum number of vertices to include at any level in the trajectory as parameters. Since

Data: G - Graph of STIX Architecture, $S = s_1, \dots, s_n$ - Set of STIX Descriptions

Result: G - The original graph now encoded as a Markov Chain Model of S

```

for each  $s_i$  in  $S$  do
   $currentTree = s_i$ ;
  // start the traversal of the current tree
  for each  $currentTree.e_j$  in  $currentTree.E$  do
     $currentEdgeInTree = currentTree.e_j$ ;
    // get the to and from vertex of the edge being
    // traversed
     $fromVertex =$ 
       $currentEdgeInTree.e_j.from$ ;
     $toVertex = currentEdgeInTree.to$ ;
    // add the toVertex to the queue in G
    // corresponding to the edge being traversed
    add  $toVertex$  to  $G.currentEdgeInTree.Q$ ;
    // now add NIL to the queue for any other edge
    // in G emanating from fromVertex
     $otherEdges = G.E$ ;
    for each  $otherEdges_j$  in  $otherEdges$  do
       $edgeToAddNilTo = otherEdges_j$ ;
      if  $edgeToAddNilTo.from =$ 
         $fromVertex$  and  $edgeToAddNilTo \neq$ 
         $currentEdgeInTree$  then
        | add  $NIL$  to  $G.edgeToAddNilTo.Q$ ;
      end
    end
  end
end

```

Algorithm 1: STIX Markov Chain Construction Algorithm

all STIX descriptions begin with a campaign component, the algorithm is initially called with a campaign component chosen at random from the STIX descriptions in S . The algorithm then recursively finds edges and destination vertices from the current vertex. Each edge and destination vertex is printed as part of the trajectory. If a NIL vertex is selected, then the current path in the trajectory is ended and paths starting at any remaining edges are explored. The trajectory is finished being printed when all paths have been explored and ended with a NIL vertex. The Markov Chain is guaranteed to include a NIL vertex for every edge because one is placed in the queue for each edge every time the edge is not taken during the construction of the Markov Chain in Algorithm 1. Algorithm 2 is specified in pseudocode below.

Case Study

To evaluate our Markov Chain modeling approach we conducted a case study inspired by combining STIX descriptions of two canonical cyber threats: (1) APT1 an advanced persistent threat from one of China's cyber espionage groups and (2) Nitro a coordinated attack to steal information from chemical makers, government offices,

Data: G - Markov Chain Model of STIX Descriptions, CV - The Current Vertex, MaxLevel - The max # of vertices at any level in a trajectory

Result: A printed depth first trajectory through G

```

 $print(CV)$ ;
if  $CV = NIL$  then
  |  $return$ ;
else
   $n = rand(0, MaxLevel)$ 
  for  $0$  to  $n$  do
     $EdgeToSample = pickAtRandom(edge$  from
       $G.E$  where  $edge.from = CV$ );
     $print(EdgeToSample)$ ;
     $NewVertex =$ 
       $pickAtRandom(EdgeToSample.Q)$ ;
     $return GenerateTrajectory(G, NewVertex,$ 
       $MaxLevel)$ ;
  end
end

```

Algorithm 2: STIX Markov Chain Trajectory Generation Algorithm

defense firms, and human-rights groups. Each of these cyber threats have been described in STIX for the purposes of information sharing and gathering cyber intelligence. Combined they contain descriptions of more than 1200 TTPs, 200 exploit targets, 150 threat actors, 50 indicators, 30 incidents, 25 observables and 20 courses of action within 2 campaigns.

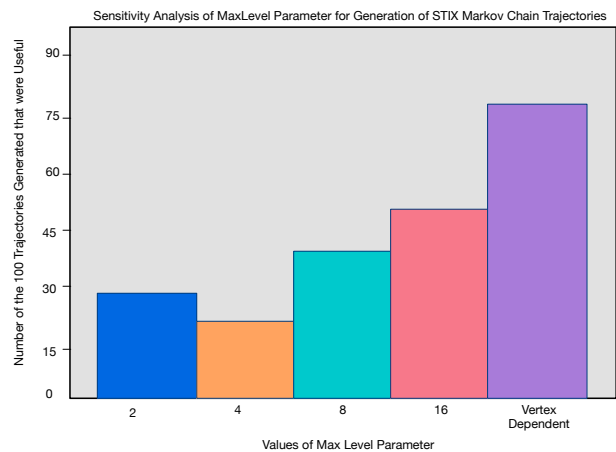


Figure 6. Analysis of how sensitive the usefulness of a generated STIX Markov Chain Trajectory is to the Max Level parameter.

First, we create a Markov Chain by augmenting the graph of the STIX architecture with the STIX description of APT1 and the STIX description of Nitro. Then, we collect different samples from the Markov Chain by generating trajectories. To identify the parameterization of Algorithm 2 that produces the most effective trajectory for this case study we

explored five different values of the *MaxLevel* parameter: 2, 4, 8, 16, *VertexDependent*. The *VertexDependent* value reflects the maximum number of edges for a given vertex among the APT1 and Nitro descriptions. For each of the five different values we generated 100 trajectories through the Markov Chain for the APT1 and Nitro STIX description. Then we manually inspected each one to determine if it was *useful* or not. Here, the term *useful* means that the STIX description reflects a cyber threat that would be of interest to someone charged with securing a cyber system. The number of useful trajectories resulting from each parameterization are shown in Figure 6.

Based on this analysis we employed the version of Algorithm 2 using the *VertexDependent* parameter to generate 10,000 possible trajectories. Of the 10,000 trajectories generated from our case study Markov Chain only $\sim 1/10$ (1,035) of them only featured annotated vertices or edges only found in the STIX description of one of the two cyber threats. The remaining 8,965 trajectories featured annotated vertices and edges from both of the threats. Within the 8,965 trajectories, several trajectory subsets were generated frequently. A histogram of the 25 most commonly generated trajectory subsets containing at least 6 vertices and 10 edges is shown in Figure 7.

The combination of automatically: (1) generating STIX descriptions that feature annotated vertices and edges from multiple cyber threats and (2) isolating the most common subsets reflecting vulnerability themes demonstrates the utility of our approach. Personnel tasked with defending a cyber system can manually inspect each of the common subsets in Figure 7 to identify any areas where they are vulnerable to an attack that combines elements from two constituent threats. Since they are only inspecting the most common (i.e. most likely) trajectories the manual burden on personnel is reduced.

Figure 7 shows that one common subset trajectory is generated significantly more frequently ($\sim 2x$) than any other. This subset is shown in Figure 8. Figure 8 shows how a vulnerability in JAVA is exploited by attackers using several different types of malware in combination to manifest a TTP.

The incident featured in our generated TTP is attacked through a vulnerability in JAVA (CVE-2012-4681) that allows adversaries to bypass security checks and download, install and use malware on a victim host. Within the TTP there are three different types of malware programs (downloader, backdoor and utilities) used in combination. First, GOGGLES, downloads and installs an encoded payload from a remote location and launches it. Next, several backdoors allowing the execution of arbitrary commands

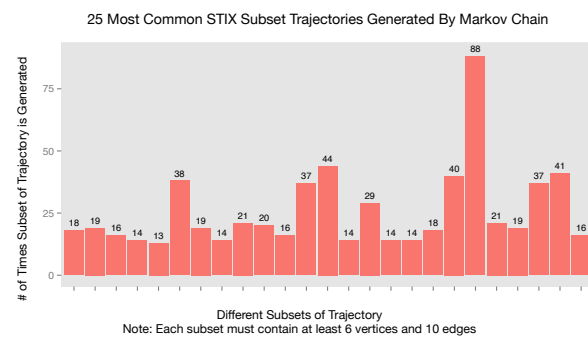


Figure 7. The 25 Most Common STIX Subset Trajectories Generated By Our Markov Chain.

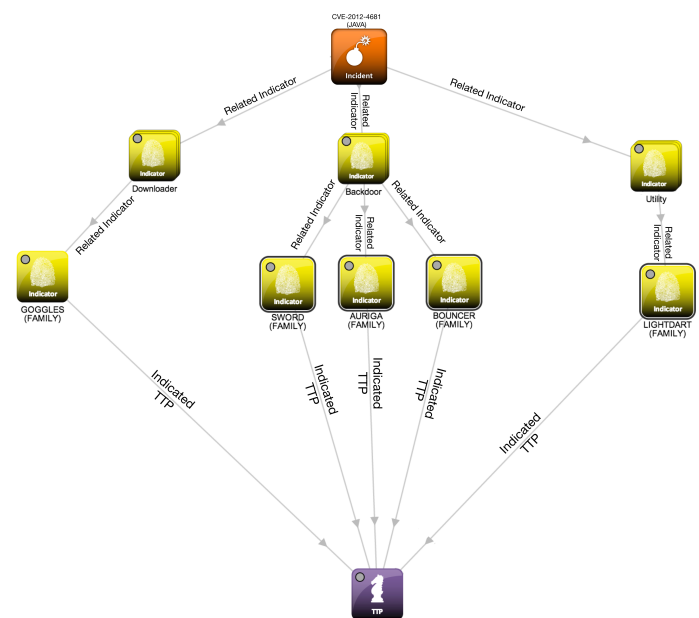


Figure 8. Previously unknown theme of common vulnerability.

through a cmd.exe instance are installed in the victim's machine using BOUNCER and SWORD. In addition the AURIGA malware family of gains access to the file system and registry to create a persistent backdoor for the virus by registering AURIGA as a service on the victim host. Finally, the LIGHTDART utility is used to upload information from the host to the attacker [13, 14].

Once identified those with a cyber security background can see this tactic to exploit the vulnerability will be effective. However, many systems may be vulnerable to it because: (1) it reflects one of the major exploit targets from Nitro, but (2) the strategy and the indicator malware used

in the TTP are representative of APT1. As a result the vulnerability may not be protected against attacks from the indicator malware. However, our Markov Chain identified a common vulnerability theme - cyber threats which exploit Nitro vulnerabilities can attack with APT1 malware. In response, personnel can add rules to their security system which prohibit the indicator malware shown in Figure 8 from interacting with vulnerable application, JAVA.

It is important to note, that uncovering a cyber threat which possessed characteristics of APT1 and Nitro was an expected outcome of our case study. Our Markov Chain provides a structure to combine the description of both threats. However, by generating 10,000 trajectories we are able to show that: (1) this cyber threat is the most common theme of attack that can be generated when combining the two and (2) while this threat has not yet been observed it is possible and could exploit a possible vulnerability in many systems. Furthermore, this information is actionable. For example, it could be included as a test in a benchmark suite that insurance companies use to evaluate the cybersecurity risk of current and prospective policyholders.

Limitations and Assumptions

Our approach is an effective means to generate possible cyber threat trajectories based on a corpus of STIX cyber threat descriptions. Furthermore, because the trajectories are representative of the entire set of descriptions one can analyze them to identify a previously unknown theme of common vulnerabilities. However, there are multiple limitations and assumptions within our approach.

First, our approach can only identify previously unobserved cyber threats vulnerability themes which are a combinations of past threats. However, not all cyber threats are extensions of the past. There is some evidence to suggest that cyber events are generally Byzantine in nature and difficult to predict based on past events [15]. In addition, our approach to constructing a Markov Chain and generating possible trajectories is resource and time intensive. However, in both cases the resources and time being used are computer related as opposed to human related. Making this type of tradeoff has enable formal methods such as model checking and theorem proving to be successful despite compute times measured in days and weeks as opposed to minutes and hours [16, 17]. Unfortunately, our approach still does require some burden on human resources. In our case study, we were able to highlight 25 subsets of trajectories that were the most common. While the generation and isolation of these 25 was automated, the process of going through the 25 subsets

to understand the vulnerabilities and secure the system is manual. This is weakness of our approach. In future work we will explore a means to automate this part of the process.

Related Work

There are a variety of the traditional technologies for detecting and preventing cyber attacks and intrusions. Most attacks start with infecting the user PC or server with malicious code. The major technologies for detecting such malicious code can be divided into the vaccine-based intrusion detection, malicious network traffic monitoring-based detection and policy driven intrusion protection and detection. Here we review all three. Our approach differs from these in that it models cyber threats to be proactive. Instead of waiting to prevent an attack it automatically generates common vulnerabilities that can be exploited by cyber threats based on previous descriptions. This section concludes with a review of other proactive modeling approaches.

Detection of Vaccine-Based Attacks

The vaccine-based intrusion attack detection method inspects the files flowing into or running in a PC or a server to check if they contain malicious code. The vaccine detects malicious code using the specific unique value of the already analyzed malicious code. To create the unique characteristics of the malicious code, the malicious behavior file must be identifiable using the result of the analysis by a skilled malicious code analyst. The unique value of the malicious behavior file is then extracted and made into a signature to be distributed to each user PC [18, 19, 20].

Vaccine-based intrusion attack detection can inspect all known malicious code in a short period. Moreover, it consumes only a small amount of system resources for detection. Note, however, that it can detect malicious code only when the malicious code used for the attack is already known. It will not detect cases such as APT attack, which uses unknown malicious code or attack pattern [21, 22].

Prevention of Intruding Malicious Network Traffic

The network-based prevention of intrusion technology can allow or prevent data similar to the specific condition. It includes firewalls, which can analyze and prevent the data penetrating into the network in real time [21, 23]. Using the specific data of network traffic, intrusion attacks can be detected and prevented. Records of outside IP, information-leaking server IP, and anomalously connecting IP used in

past intrusion attacks can aid in preventing the recurrence of the same attack. Moreover, it can allow or prevent intrusion of the specific service of the traffic to prevent unintended access. By monitoring traffic in and out for a specific period, it can detect anomalously created traffic [24, 22]. Like the detection of vaccine-based intrusion attack, however, its limitation is that it can perform prevention based only on known attack or analyzed data. Although it can detect unknown anomalous traffic, it can detect traffic during a short period of 1-2 days only, but not attacks which occurs over a long period.

Policy Driven Intrusion Protection & Detection Systems

Policy-based prevention of intrusion technology attempts to define the line between benign and malicious applications as a set of rules. These rules specify what an application is allowed to do and attempts to violate the rules are considered intrusions. The rules governing an application define precisely which system resources an application can access and in what way [25]. Some approaches automate this process by deriving application-specific system call behavior model from the application's source code, and checks the application's run-time system call pattern against this model to thwart any control hijacking attacks [26]. Others have moved the approach from a machine's operating system to its web browser. The idea is that a web site can embed a policy in its pages that specifies which scripts are allowed to run. The browser, which knows exactly when it will run a script, can enforce this policy [27].

Proactive Modeling Approaches of Cyber Threats

In recent years several proactive approaches to modeling and detering cyber threats have been discovered. Two approaches are similar to ours. The first reproduces real cyber attacks that have signature performance data which can identify them. Machine learning is used to train a classification algorithm on the performance data resulting from the attack. The result is a technology that has been pre-trained for a variety of cyber attacks so that if any are attempted the resulting performance data will be identified immediately and the privileges of the attacking program will be revoked so that it cannot cause any harm to the system [28]. Another approach leverages the coevolutionary relationship between attackers and defenders to derive a new method for proactive network defense. The approach involves exploiting basic threat information (e.g., from cyber security analysts) to generate synthetic

attack data for use in training defense systems. The result is networks defenses that are effective against both current and near future attacks [29]. Unlike, our approach the approach does not use a formal markup language like STIX to model cyber threats nor does it employ a Markov Chain to ensure that the cyber threats generated are representative of the data set of cyber threat descriptions.

Conclusion and Future Work

Traditional approaches for cyber security focus on understanding and addressing vulnerabilities. While this mindset is necessary it is not sufficient. In order to make informed defensive decisions that optimize the use of limited resource a bettering understanding of the nature of existing and future cyber threats is needed. By applying Markov Chain methods we address this need. Our Markov Chain of cyber threats ingests descriptions of the previously observed cyber threats encoded in the description language STIX. Using the STIX architecture of a cyber threat as a directed graph it creates a mechanism to generate possible cyber threat trajectories that are representative of the collected STIX descriptions. Using analysis tools to summarize the generated threats provides cyber defenders with the ability to identify common themes of vulnerabilities that cyber threats can exploit. In a case study, we explore a previously unknown cyber threat vulnerability, characterized by indicators of two well known cyber threats.

Our future work will focus on automating the analysis of the data generated by our approach to inform cyber threat intelligence and inform cyber insurance scoring. Specifically, we will explore the extent to which insurance companies can use are generated threats to evaluate cybersecurity risk of current, prospective policyholders, as demand for cyber insurance increases. Companies will be screened using our generated threats to determine their cybersecurity risk when they apply for insurance.

Author Biographies

Dr. Ross Gore holds a Doctorate of Philosophy and a Master's degree in Computer Science from the University of Virginia and a Bachelor's degree in Computer Science from the University of Richmond. He has ten years of research experience in problems that lie at the intersection of software engineering and modeling and simulation. His work has yielded authorships on more than 20 conference and journal publications and has been recognized by the Achievement Rewards for College Scientists Society as an impactful and novel research avenue.

Dr. Jose Padilla studies the concept of understanding and how the concept applies to areas such as Modeling and Simulation, Systems Science, and complexity. He is currently conducting research into how the concept of understanding can be used in the process of conceptual modeling and in the creation of autopoietic agents. His interests include Human, Social, Cultural, and Behavior Modeling, Epistemology of Modeling and Simulation, Conceptual Modeling, Systems Engineering and Engineering Management.

Dr. Diallo is a Research Assistant Professor at the Virginia Modeling Analysis and Simulation Center (VMASC) of the Old Dominion University. His research focuses on the theory of interoperability as it relates to model-based data engineering and web services for modeling and simulation applications. He has authored or co-authored over fifty publications including a number of awarded papers and articles in conferences, journals, and book chapters.

Acknowledgements

This material is based on research sponsored by the Office of the Assistant Secretary of Defense for Research and Engineering (OASD(R&E)) under agreement number FAB750-15-2-0120. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. We also gratefully acknowledge the support of our colleagues at the Virginia Modeling, Analysis and Simulation Center at Old Dominion University.

References

- [1] Sean Barnum. Standardizing cyber threat intelligence information with the structured threat information expression (stix). *MITRE Corporation*, page 11, 2012.
- [2] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1:80, 2011.
- [3] Scott Swanson, Craig Astrich, Michael Robinson, et al. Cyber threat indications & warning: Predict, identify and counter. *Journal Article— Jul*, 26(4):59–71, 2012.
- [4] Vipin Kumar, Jaideep Srivastava, and Aleksandar Lazarevic. *Managing cyber threats: issues, approaches, and challenges*, volume 5. Springer Science & Business Media, 2006.
- [5] Frank L Greitzer and Deborah A Frincke. Combining traditional cyber security audit data with psychosocial data: towards predictive modeling for insider threat mitigation. In *Insider Threats in Cyber Security*, pages 85–113. Springer, 2010.
- [6] Jelle De Vries, Hans Hoogstraaten, Jan van den Berg, and Semir Daskapan. Systems for detecting advanced persistent threats: A development roadmap using intelligent data analysis. In *Cyber Security (CyberSecurity), 2012 International Conference on*, pages 54–61. IEEE, 2012.
- [7] Aditya K Sood and Richard J Enbody. Targeted cyberattacks: a superset of advanced persistent threats. *IEEE security & privacy*, (1):54–61, 2013.
- [8] Nikos Virvilis, Dimitris Gritzalis, and Theodoros 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)*, pages 396–403. IEEE, 2013.
- [9] Paul Giura and Wei Wang. A context-based detection framework for advanced persistent threats. In *Cyber Security (CyberSecurity), 2012 International Conference on*, pages 69–74. IEEE, 2012.
- [10] Oscar Serrano, Luc Dandurand, and Sarah Brown. On the design of a cyber security data sharing system. In *Proceedings of the 2014 ACM Workshop on Information Sharing & Collaborative Security*, pages 61–69. ACM, 2014.
- [11] Julie Connolly, Mark Davidson, Matt Richard, and Clem Skorupka. The trusted automated exchange of indicator information (taxii), 2012.
- [12] Walter R Gilks. *Markov chain monte carlo*. Wiley Online Library, 2005.
- [13] Ping Chen, Lieven Desmet, and Christophe Huygens. A study on advanced persistent threats. In *Communications and Multimedia Security*, pages 63–72. Springer, 2014.
- [14] Stevens Le Blond, Adina Uritesc, Cédric Gilbert, Zheng Leong Chua, Prateek Saxena, and Engin Kirda. A look at targeted attacks through the lense of an ngo. In *Proc. 23rd USENIX Security Symposium*, 2014.
- [15] M Uma and G Padmavathi. A survey on various cyber attacks and their classification. *IJ Network Security*, 15 (5):390–396, 2013.

- [16] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.
- [17] Edmund M Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT press, 1999.
- [18] Wei Yan and Erik Wu. Toward automatic discovery of malware signature for anti-virus cloud computing. In *Complex Sciences*, pages 724–728. Springer, 2009.
- [19] Suchita Patil, Pradnya Rane, and Dr BB Meshram. Ids vs ips. *Proceedings of International Journal of Computer Networks and Wireless Communications*, 2, 2012.
- [20] Martin Lee and Darren Lewis. Clustering disparate attacks: mapping the activities of the advanced persistent threat. *Leading Issues in Information Warfare & Security Research*, 26, 2013.
- [21] Peter Mell, Karen Kent, and Joseph Nusbaum. *Guide to malware incident prevention and handling*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, 2005.
- [22] Terence Slot and Frank Kargl. Detection of apt malware through external and internal network traffic correlation. 2015.
- [23] Radoslav Bodo and Michal Kostenec. Experiences with ids and honeypots-best practice document, 2012.
- [24] Ahmed Patel, Qais Qassim, and Christopher Wills. A survey of intrusion detection and prevention systems. *Information Management & Computer Security*, 18(4): 277–290, 2010.
- [25] Suresh N Chari and Pau-Chen Cheng. Bluebox: A policy-driven, host-based intrusion detection system. *ACM Transactions on Information and System Security (TISSEC)*, 6(2):173–200, 2003.
- [26] Lap Chung Lam and Tzi-cker Chiueh. Automatic extraction of accurate application-specific sandboxing policy. In *International Workshop on Recent Advances in Intrusion Detection*, pages 1–20. Springer, 2004.
- [27] Trevor Jim, Nikhil Swamy, and Michael Hicks. Defeating script injection attacks with browser-enforced embedded policies. In *Proceedings of the 16th international conference on World Wide Web*, pages 601–610. ACM, 2007.
- [28] Md Tanzim Khorshed, ABM Shawkat Ali, and Saleh A Wasimi. A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. *Future Generation computer systems*, 28(6):833–851, 2012.
- [29] Richard Colbaugh and Kristin Glass. Proactive defense for evolving cyber threats. In *Intelligence and Security Informatics (ISI), 2011 IEEE International Conference on*, pages 125–130. IEEE, 2011.