Defensive Maneuver Cyber Platform Modeling with Stochastic Petri Nets

William Clay Moody, Hongxin Hu, Amy Apon
School of Computing
Clemson University
Clemson, SC, USA
wcm,hongxih,aapon@clemson.edu

***Abstract*—Distributed and parallel applications are critical information technology systems in multiple industries, including academia, military, government, financial, medical, and transportation. These applications present target rich environments for malicious attackers seeking to disrupt the confidentiality, integrity and availability of these systems. Applying the military concept of defense cyber maneuver to these systems can provide protection and defense mechanisms that allow survivability and operational continuity. Understanding the tradeoffs between information systems security and operational performance when applying maneuver principles is of interest to administrators, users, and researchers. To this end, we present a model of a defensive maneuver cyber platform using Stochastic Petri Nets. This model enables the understanding and evaluation of the costs and benefits of maneuverability in a distributed application environment, specifically focusing on moving target defense and deceptive defense strategies.**

I. Introduction

Multiple institutions in academia, industry, and government have discovered the necessity of parallel and distributed computing in data driven business processes for the solution of complex computational problems. The significant financial investment and operational reliance of these systems create critical infrastructure that is tightly bound to the success of the organization. The security of these platforms is vital to survival of these establishments.

Malicious actors seeking financial or intelligence gains are targeting supercomputers and distributed computing centers at an increasing rate [1], [2], [3]. Their methods and efforts to disrupt the confidentiality, integrity, and availability of the systems require network security professionals and researchers to invest remarkable amounts of time and money into protecting these assets. We argue that one approach to improving system security is the application of military doctrine and tactics to the defense of distributed and parallel applications. In this paper we focus on extending the military concept of maneuver to the defense of cybersystems.

Maneuver is one of the U.S. Army's nine Principles of War [4]. Maneuver includes the application of combat power to maintain an advantage over the enemy. This flexible and dynamic employment of resources ensures success by keeping adversarial conditions imbalanced and thus reduces failures, compromises, and vulnerabilities. The non-military use of the word *maneuver* describes an action that is not random or without purpose, but one that is clever or skillful. In both environments the word maneuver implies deliberate movement and actions taken to achieve a specific purpose. Defensive maneuver in cyberspace operations has been broken down into four basic forms [5]. These forms are: perimeter defense in depth, moving target defense, deceptive defense, and counter attack. In this work, we focus on applying moving target defense and deceptive defense for maneuverability in distributed and parallel applications.

Security and usability have always been at odds with each other in information systems. Adding maneuverable elements and features into distributed and parallel applications affects the performance and operational output of the system. Our goal is to understand and characterize how these additional features can improve security while limiting the impact on operations. Our approach is to develop a model of these processes and systems and to use the model for understanding and evaluating these trade-offs.

Our work with the modeling the defensive maneuver cyber platform provides the following contributions:

- A Stochastic Petri Net (SPN) model of a defensive maneuver cyber platform utilizing moving target defense and deceptive defense tactics.
- An analysis of our SPN model to understand the trade-offs between security and operations in defensive maneuver cyber platform.
- Recommendations for how to use and extend our current SPN model to build prototype systems implementing moving target and deceptive defense in parallel and distributed applications.

The remainder of this paper is organized as follows: Section II provides the background information needed for understanding the system model. Section III describes the design goals of the defensive maneuver cyber platform and its characteristics. We provide analysis of our system in Section IV. We discuss related works and present our conclusions in Sections V and VI, respectively.

II. Background

A. Military Maneuver and Cyberspace

In 2010, William J. Lynn III, former United States Deputy Secretary of Defense, publicly released the details of a previously classified incident of the spread of malicious computer code across the U.S. military's unclassified and classified computer networks [6]. This malware was the product of a foreign intelligence service and spread through the unauthorized movement of USB based flash drives between computing systems with different levels of classification. The efforts of the Department of Defense to counter the attack was known as Operation Buckshot Yankee and was managed at the highest levels of the Department due to the sensitivity and urgency. The highly visible release of information to the public highlighted the scope of daily cyber attacks and intrusions faced by Pentagon and its military. This incident highlighted the need for improved operations and defense in the newest war fighting domain.

Since the mid 1940s, war fighting has been restricted to the traditional domains of Land, Sea, and Air. Recent technological advances have led to adding Space and Cyberspace to the list of operational domains. With the expansion of domains to conduct combat operations, traditional military doctrine must be reevaluated or expanded to provide consistency across the entire military spectrum. This challenge has presented a plethora of opportunities for strategic debate, theoretical analysis, and technological research.

In 2010 the Department of Defense created a new organization, the United States Cyber Command (USCYBERCOM). Tasked as the lead for all military cyberspace actions, this new organization is leading the effort to define and extend military operations into Cyberspace.

Many traditional military topics have been expanded and applied into the cyberspace domain [7]. These include the concept of situational awareness [8], key terrain [9], and defense in depth [10]. A basic military concept that has received extension theoretical review is *maneuver* [5], [11], [12]. Empowered by this theoretical search there has been limited work in actually building systems that exhibit the characteristics and behaviors of maneuver [13], [14], [15].

B. Petri Nets

Petri Nets were created by Carl Adam Petri [16] as a method to study concurrency in parallel and distributed applications. Petri Nets have been applied in many disciplines and extended in multiple facets through the years. Petri Nets are extended in use within the computer science field to model of a distributed computing system in which to study the correctness, concurrency, and synchronization.

A Petri Net is a bipartite, weighted, directed graph composed of two types of nodes called Places and Transitions and edges called Arcs. Places represent preconditions or values of variables in the system and are denoted as circles graphically. Transitions represent actions taken as a result of valid preconditions or values and are represented as rectangles or lines. Since the graph is bi-partite, arcs only exist between places and transitions. There are no arcs between two places or two transitions. Places from which an arc originates are considered input places to the transitions in which the arc terminates. Places in which arcs terminate are considered output places of the transition at the origin of the arc. Tokens, represented by solid dots, indicate preconditions being true or values of variable. Multiple Tokens can be found in places throughout a net. The distribution of tokens over a net represents the configuration of the system and is called the marking. Arcs have weights that represent the number of tokens consumed or produces as a result of transitions "firing". When a transition "fires" tokens from its input places are consumed and tokens are produced in the output places. Transition can only fire when enabled, meaning all input places have the minimum number of required tokens. Transition firings are atomic and nondeterministic. Multiple transitions could be enabled concurrently, but the order in which the fire is not known and the result of a transition firing can result enable or disable other transitions.

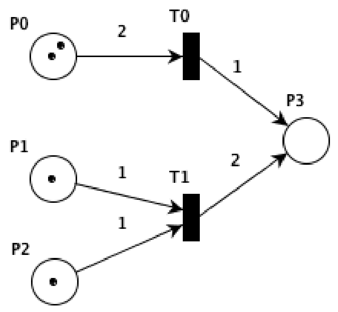

Fig. 1. A basic Petri Net with four places, two transitions and four arcs.

A basic Petri Net with four places and two transitions is show Figure 1. In the current marking of this Petri Net, $P0$ has two tokens, $P1$ and $P2$ each have one token, and $P3$ has no tokens. The weight of two on the arc from $P0$ to $T0$ indicates that two tokens in $P0$ must be present for $T0$ to be enabled to fire. The weight of 2 on the arc from $T1$ to $P3$ indicates that if $T1$ fires that it will place two additional tokens in $P3$. These tokens are added to any tokens that may already be present in $P3$.

A marking of a Petri Net is indicated as M which is a vector of length m which is the number of places in Petri Net. Each value of m_i represents the number of token present in the ith place. Each Petri Net has an initial marking, M_0, which describes the initial state of the system. A marking, M', is said to be reachable if a series of valid transition firings from M_0 results in M'. A way to visualize all reachable states is to create a reachability graph that shows all reachable markings and the transition firing sequence to each state. This graph shows the entire state space for a Petri Net. The reachability graph for the basic Petri Net presented above is shown in Figure 2.

C. Stochastic Petri Nets

The basic Petri Net as describe above has been extended in many ways in the literature [17], [18]. One such extension, is the Stochastic Petri Net (SPN) [19] where transitions have a delay between enabling and firing. This delay is recalculated

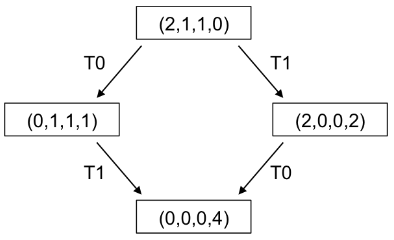

Fig. 2. Reachability Graph for basic Petri Net in Figure 1.

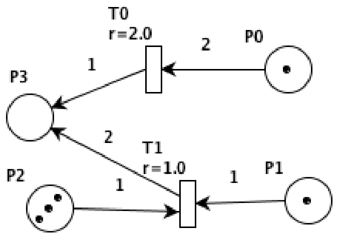

Fig. 3. A Stochastic Petri Net with firing rates for each transition shown

each time the transition becomes enabled and has an exponential distribution where each transition is assigned its own firing rate. When multiple transitions are enabled, the transition with the shortest delay time will fire first. Graphically, the timed transitions are shown as unfilled rectangles, as opposed to solid rectangles or straight lines which are immediate transitions in the standard Petri Net. Figure 3 shown a sample SPN with firing rates $(\lambda_0, \lambda_1, \lambda_2)$.

An SPN in which the number of tokens in a place is finite has been shown to be equivalent to a finite Markov Chain. The Markov Chain can be determined from the reachability graph of the SPN and the initial marking. This makes it possible to solve the steady-state distribution of the SPN. This steady-state distribution allows one to analyze systems described as stochastic Petri Nets to determine probability of the system existing in a given state. We will use this calculation to analyze our Defensive Maneuver Cyber Platform.

Formally, a Stochastic Petri Net can be described as a 6-tuple PN as described in Table I.

TABLE I
FORMAL DEFINITION OF STOCHASTIC PETRI NET

$SPN = (P, T, R, I, W, M_0)$
$P = \{p_0, p_1, ..., p_m\}$ is a finite set of places,
$T = \{t_0, t_1, ..., t_n\}$ is a finite set of transitions,
$P \cap T = \emptyset$,
$\lambda = \{\lambda_0, \lambda_1, ...\lambda_n\}$ is the set of transitions firing rates,
$I = I^- \cup I^+$
$I^- \subseteq (P \times T)$ and $I^+ \subseteq (T \times P)$ are a sets of input and output arcs,
$W : I \rightarrow \{1, 2, 3, ...\}$ is a weight function
$M_o : P \rightarrow \{0, 1, 2, 3, ...\}$ is the initial marking.

D. *Platform Independent Petri Net Editor (PIPE2)*

The Platform Independent Petri Net Editor (PIPE2) [20] began as a graduate course group project at the University of London in 2003. The goal of the project was to design an application that allows Petri Nets to be designed graphically and analyzed.

PIPE2 is written in Java and compatible with any OS capable of running Java programs. The system provides the capability to create standard Petri Nets and Stochastic Petri Net models. The tool allows a user to draw a SPN using drag-and-drop tools on a canvas, then to save the file in an XML file that can be opened in other applications. The tool also has the ability to animate the model with random firing of transitions or interactive user manipulations. The key aspect of the tool is the wide variety of analysis model and the ability for users to design and integrate their own custom modules. One such module is the Steady State Analysis. This module determines reachability graph of a SPN then calculates the steady state distribution.

III. THE DEFENSIVE MANEUVER CYBER PLATFORM MODEL

In an effort to provide improved cybersecurity to a distributed and parallel application, we have designed a model of a defensive maneuver cyber platform using Stochastic Petri Nets. In the model, we represent individual nodes that can operate in specific modes, along with the collection of nodes that constitute the parallel and distributed application. We specifically apply the concepts of moving target defense and deceptive defense to provide increased security. Individual nodes transition between operational, idle, and deceptive modes. Additionally, overall controls of the application seek to ensure that a minimum number of nodes remain operational and deceptive as dictated by the current operational and threat environment. In the event of increased adversarial activity or indications and warnings of threat activity, the system parameters can be adjusted to provide a more secure environment at the cost of operational output. Likewise, in low threat environments, system settings can allow improved productivity at the cost of security. It is understood that these tradeoffs in a system of this type are both necessary and beneficial to system administrators and executive leadership.

In our model, a distributed and parallel system is composed of multiple individual nodes controlled by a separate central management system. We assume that individual nodes are worker nodes of the overall system and can be in one of three states, operational, deceptive, or idle. We further assume that a node does not move directly from an operational state to a deceptive state, but goes to an idle state first, and vice versa. The individual nodes can join or leave the cluster with minimal operational overhead, but otherwise ensure the survivability of the system. One such example of a systems that survives in this manner is JUMMP [21].

Each individual node is comprised of three places (operational, idle, deceptive) and 4 transitions (operational-to-idle, idle-to-operational, deceptive-to-idle, and idle-to-deceptive). There are two universal places (minimum computation, minimum deceptive). There are eight intranode arcs between the places and transitions for each node and an additional six external arc from the control places into the node for management. A single token exists within each node and the place in which it is found determines the state of the node, as described below. The number of nodes is configurable by the system depending on the needs of the application designers.

Formally, a node can be describe as a Stochastic Petri Net as shown in Table II.

A. Assumptions

We have placed some assumptions on our system to aid in modeling. These assumptions are based on realistic expectations of the system in the real world. We assume that the system has a centralized controller that is not a target of an attacker in this system. The centralized controller is responsible for directing the node transitions between states, monitoring the status of all nodes, and ensuring the rules and parameters of the system are obeyed. We assume the system control is aware of the threat conditions and can make intelligent decisions based on this knowledge. We also assume the system management is willing to lessen the operational throughput when threat conditions exists that increases the probability of adversary action exceeds a predetermined threshold.

TABLE II
DESCRIPTION OF SINGLE NODE SPN

$$P = \{p_o, p_i, p_d\}$$
$$T = \{t_{oi}, t_{io}, t_{di}, t_{id}\}$$
$$\lambda = \{1.0, 1.0, 1.0, 1.0\}$$
$$I^- = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$
$$I^+ = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$W = \{1, 1, ..., 1, 1\}$$
$$M_o = \{(1,0,0)|(0,1,0)|(0,0,1)\}$$

B. Individual Node States

When a node is in the operational mode, it is an active member of the parallel and distributed application. In an operational state the node is storing data, processing data, or transferring data to other operational nodes in the system. In the instance of the application Hadoop, these nodes would be engaged datanodes and task trackers performing MapReduce calculations. Operational nodes are listening to known TCP or UDP ports, creating network connections to other nodes listening to these same ports, or transferring data on previously connected flows. Operational nodes are susceptible to attacks from malicious actors, but each is providing an operational contribution as part of the purpose of the parallel and distributed application. Operational nodes, when directed by the central management system, can transition to the idle state. When this transition occurs we assume that the application can continue to operate as a whole working system but with a loss of operational capability.

An idle node is one which is available to be promoted to becoming either an operational or a deceptive mode. In some instances, this node could be part of a larger shared computing resource cluster and available to other users or applications. We assume that idle nodes are non-utilized resources that are available but not susceptible to targeting by an attacker. When directed by the central manager, the node transitions to becoming either an operational node or a deceptive node.

The third mode in which an individual node can exist is the deceptive mode. In the deceptive mode a node is indistinguishable from an operational node to an outside observer. An attacker that scans the network or enumerates the environment will be just as likely to target a deceptive node as it would an operational node. The deceptive nodes are listening to the same TCP / UDP ports, creating connections with other deceptive nodes, and transferring data between each other. Additionally, we assume that decoy nodes have honeypot programs [22] that alert the central management system to the existence of intruders on the network. These triggers can be used to increase the maneuver rate of the nodes or adjust the ratio of computational to deceptive nodes.

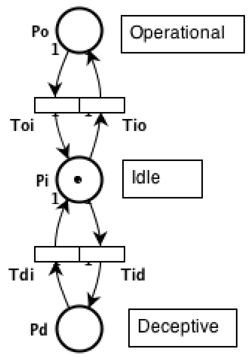

Fig. 4. Petri Net of Individual Node which can transition between three modes of operation.

In the Petri Net model, each mode (operational, idle, and deceptive) is represented by a place (P_O, P_I, P_D, respectively). There is a single token which, by its place location, represents the mode of the node. Timed transitions allow the token to move from operational to idle, idle to operational, deceptive to idle, and idle to deceptive. These transitions also are fed by arcs from the centralized control places (P_{No}, P_{Nd}) to ensure the system maintains the minimum number of operational and deceptive nodes. Figure 4 shows a single node system with the places labeled with names. The modes for the node are represented on the far right side of the node. The current marking has the token in the idle place, indicating its current state.

C. System Parameters

The Defensive Cyber Maneuver Platform is composed of one or more individual nodes. The model has a set of configurable parameters that allow exploration of the trade-offs between different system designs. Parameters include the

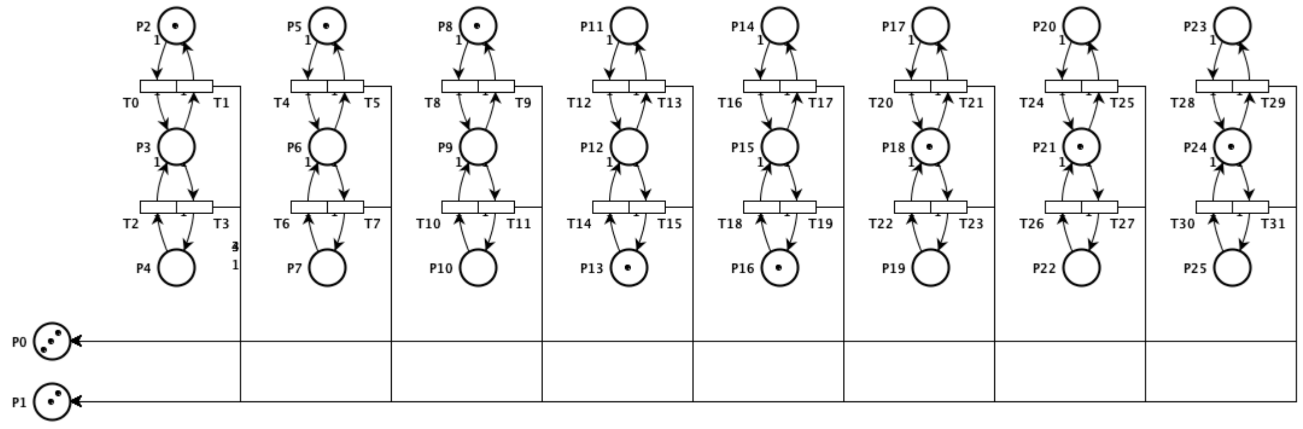

Fig. 5. An eight node Defensive Maneuver Cyber Platform with three operational nodes, two deceptive nodes, and three idle nodes.

total number of nodes in the system, the number of nodes that are operational, deceptive, and idle in the initial system configuration, and the upper and lower bounds of the number of nodes for each of these states. The parameters can be used to compare tradeoffs of different configurations.

The system has a total number of N nodes. Each node is a single whole node as described above. The system has a minimum number of nodes that must remain in the operational and deceptive states, these are described with the variables O and D. Since Petri Net transitions are atomic and nodes can not transfer directly between operational and deceptive states but must move to the idle state, then $N > O + D$ must hold true for the system in all configurations.

In order to ensure the stability of the maneuver system, the minimum number of operation and deceptive nodes is managed by the system controls with the two places P_{No} and P_{Nd}. These places are initialized with O and D tokens, respectively. An input arc exists between each control place and the transition to the idle state. This arc is weighted to $O+1$ and $D+1$ with an output arc back to the control place with weight O and D, respectively. This ensures that a node that transitions into the idle state leaves at least the minimum nodes in its former state, thus not violating the minimal levels set by the system designers. Additional, any transition from idle to operational or deceptive states will deposit a token in the control place. The control place acts as a counter of nodes in its state.

D. Maneuver Platform Builder

The defensive cyber maneuver system is modeled as a Stochastic Petri Net. We use the Platform Independent Petri Net Editor (PIPE2) [20] to view and evaluate the system. PIPE2 provides a graphical user interface for drawing and building Petri Nets. These nets can then be saved in a Petri Net Markup Language compatible file for use in other PNML tools. PNML is an extension of XML as defined by the standard ISO/IEC 15909 Part 2 [23].

In order to aid in the creation of Petri Nets of our maneuverable system, we designed and developed a maneuver platform builder that automates the creation of the nets. The builder extends the standard XML libraries that currently exist in Python. This custom software tool, called `buildDMCP` takes command line parameters of (N, O, D). The output of the program is a PNML file that can be opened in PIPE2 or any other PNML compatible tool.

The `buildDMCP` tool takes an object oriented approach by defining the building blocks for Petri Nets in general. Functions to build places, transitions and arcs are developed with optional parameters. Using these building blocks, the code builds a single node of our maneuverable system with the respective places, transitions, and arcs. Additionally, the system control are created and connected to the node at the specified locations. Finally, depending on the user provided parameters, the total number of nodes and minimal levels for computational and deceptive nodes are built.

The resultant Petri Net has all the places, transitions, and initial markings specified by the user. Additionally, these items are organized and placed on the canvas in a visually appealing manner. Animations and analysis of the nets are done within the PIPE2 application. Figure 5 shows defensive cyber model with $N = 8$, $O = 3$, and $D = 2$ created with `buildDMCP` and visualized in PIPE2. $P0$ and $P1$ in this figure represent the centralized control mechanism of the system (P_{No} and P_{Nd}). These places maintain a count of the current number of operational and deceptive nodes. When a node transitions from idle to either operational or deceptive states, a token is deposited into these places. In order for a node to transition from either operational or deceptive, there must exist $O+1$ or $D+1$ tokens in these places, respectively. In the SPN, this is modeled with output arcs of weight $O+1$ or $D+1$ and input arcs of weight O or D.

IV. Analysis

To understand the trade-offs between security and operations in our system, we provide some analysis of the Defensive

Maneuver Cyber Platform. Understanding the characteristics and behaviors of the system under different configurations will allow system designers and administrators to choose implementations to meet their operational needs. Additionally, this improved understanding of the system will improve our future efforts to build prototype software packages base on this model. We will focus our analysis on enumerating the potential states of the system and studying how adjusting the transition firing rates affects the state distribution probability.

A. *Enumerating and Categorizing State Space*

Initially, we want to study how the parameters of the configuration affect the state space in which the system can exist. The Defensive Maneuver Cyber Platform is specified with three global parameters, N, O, and D. The current state of the system can be described as a vector C of length N where each element C_j represents the state (S_o, S_d, S_i) of node j^{th} in the system. The total number of nodes in each state can be represented by the variables (o, d, i). For each state or marking with equal counts of nodes in each state can be in multiple combinations. It can be shown that the total number of states or markings, M, of the Petri Net is defined by Equation 1

$$|M| = \sum_{d=D}^{N-O} \sum_{o=O}^{N-d} \frac{N!}{o!\, d!\, i!} \qquad i = N - o - d \tag{1}$$

This equation is a summation over the range of two variables that describe the count of computation and deceptive nodes. The equation is similar to the computation of multistate combinations, since with have N nodes that can be either operational, deceptive or idle. The number of operational nodes ranges from the system specified minimum to all remaining nodes after preserving the minimal level of deceptive nodes. Likewise, the deceptive node count ranges from the defined minimal level to all other nodes minus the minimal operational nodes.

The size of the state or marking space of the Defensive Maneuver Cyber Platform is affected by the three variables of the system. It can be shown the maximum state space for a given N is indirectly proportional to the minimum levels of operation and deceptive nodes (O and D). This allows the summation to have a larger range, thus resulting in the maximum state space. The minimal size of the state space is calculated when you maximize O and minimize D or vice versa. This results in only one possible value of o and d in the system.

Figure 6 plots the minimum and maximum values of the state space over the range $8 \leq N \leq 128$. As the graph shows, the maximum is on the order of $\exp n$ where the minimum is on the order of $\log n$.

These states can be broken down into groups that determine the maneuverability of the cluster. The maneuverability of the cluster is a measure of the amount that nodes can change from one state to the next. An idle node is always eligible to change state, where operational and deceptive nodes can only maneuver if the current count of nodes is greater than the minimum levels. When the current state of the system has $o = O$ and $d = D$ the system is considered minimally maneuverable because only the $N - O - D$ idle nodes can maneuver. When the system has more of both operational and deceptive node than the minimum configuration ($(o > O)$ and $(d > D)$) then the system is considered fully maneuverable since all nodes can maneuver at the next moment.

The reachability graph includes in and out edges between markings. These edges are related to the transitions and arc of the SPN. Quantifying the number of edges in the reachability graph is required to calculate the entire state space of the system. It can be shown that the number of edges is greater than the number of states. Calculating all out edges is equal to the edges of the system since all edges can be considered in and out edges. Minimally, maneuver states have the minimum number of out edges. This is equal to the $N-O-D$. The state with the most adjacent states is when $o = O+1$ and $d = D+1$. This is a fully maneuverable state with maximum idle nodes. Thus the number of adjacent states is $2N - O - D - 2$ since each idle node can maneuver to operational or deceptive, while each operational and deceptive node can maneuver to idle. Without specific definition, it can be stated that the number of edges in a graph is on the order of $n * |M|$. This is an important distinction when solving the steady state probability distribution for the SPN.

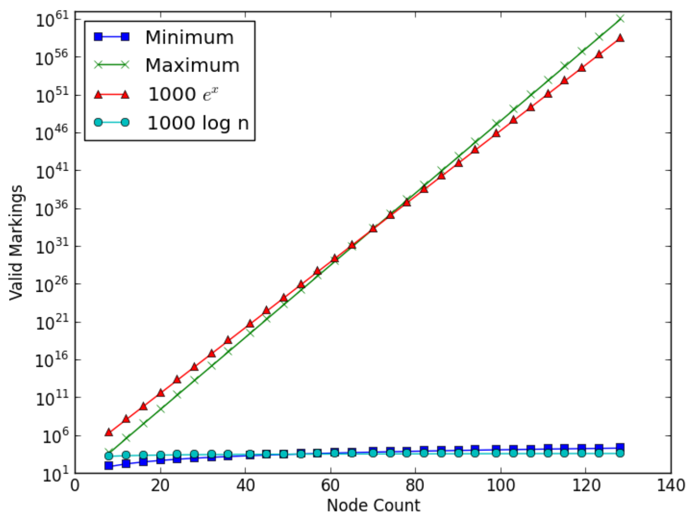

Fig. 6. Scale of N

B. *Transition Firing Rate Impact*

Stochastic Petri Nets have transitions that fire with an exponentially distributed random variable firing delay. Each transition j has a rate λ_j used to calculate the delay. We have studied the effect that the firing rate has on the probability distribution of the steady state of the system.

Since an SPN is isomorphic to a continuous time Markov chain, the steady state distribution π of an SPN can be solved by using the transition rate matrix Q, where Q is the rate between states of the reachability graph and diagonals are the negative sum. As shown earlier, Q is a very sparse matrix and its dimensions grow very fast with N.

PIPE2 provides a module for determining the reachability graph and conducting the steady state analysis. We use this tool to study the probability distribution for our system and the firing rate impact on the distribution. It can be shown with all transition firing rates equal, the Defensive Maneuver Cyber Platform markings are equally likely.

We began by grouping the transitions in each node into pairs. One group (T_{io}, T_{di}) maneuvers the individual towards operational while the second group (T_{oi}, T_{id}) maneuvers towards deceptive. Varying the rate of the firing rates of the two groups changes the state space probability distribution of the entire cluster. We grouped each possible state by the number of operational nodes (the marking of P_{No}) and calculated their combined probabilities. This way we are able to see how the transition firing rates impact the probably the system has various levels of computational state.

Using an 8-3-2 Defensive Maneuver Cyber Platform as an example, the reachability graph of the SPN has 2478 nodes, or valid markings. The range of the number of computational nodes is $(3, 6)$ while we can have $(2, 5)$ deceptive nodes. Figure 7 shows probability over the range of operational node counts when the ratio of the maneuver rate to the deceptive rate is varied. The same analysis of the deceptive count of the system is left as future work.

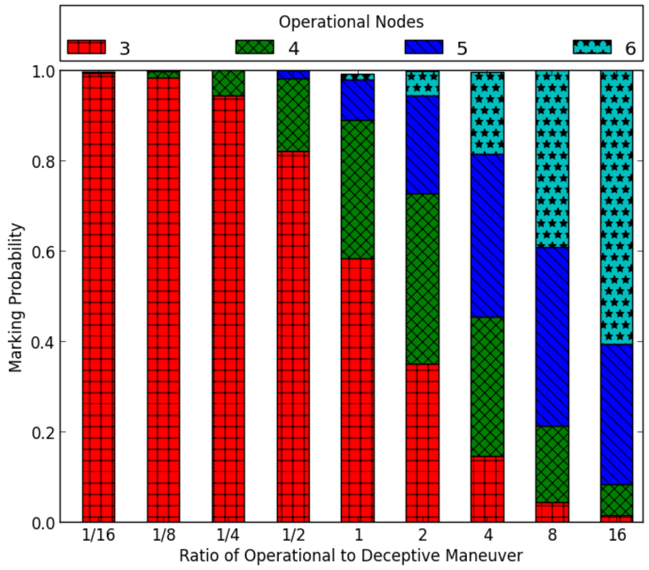

Fig. 7. Maneuver Rate effect on Probability of Computational State

V. Related Works

Moving target defense (MTD) has recently received a great deal of attention [24], [25], [26], [27], [28], [29], [30]. In [24], the authors addressed basic research challenges and how MTD can be deployed using asymmetric cost techniques that are advantageous for the defenders and disadvantageous for the attackers. In [25], the authors explored the effectiveness of MTD protection mechanisms. A model for dynamic diversity defense is proposed. In [26], an OpenFlow Random Host Mutation (OF-RHM) scheme is introduced to use OpenFlow to efficiently assign different addresses to hosts and protect against internal and external scanning. In [27], the authors developed a Moving Target IPv6 Defense (MT6D) that leverages the immense address space of IPv6 to hide and rotate IPv6 assignments by implementing MT6D tunneled packet. The two goals of MT6D is to maintain user privacy and protecting against targeted network attacks. A similar functionality in the form of a Linux hypervisor is provided by [31]. In [28], the authors described an approach to inject artificial diversity into system for use as cyber maneuvers in a moving target defense, which employs uses control theoretic principles. The maneuvers discussed in this work include memory randomization, IP address randomization and applying a new state machine with random extra states for protocols like DHCP. In [29], a concept of Random Route Mutation (RRM) is introduced and algorithms are defined to achieve optimal path randomization between a source and a destination. In [30], a basic design schema of a moving-target network defense system is presented and a simulation-based study is conducted to explore the degree to which proactively changing a network's various parameters can decrease an adversary's chance for success.

Some research efforts have been devoted to security modeling and analysis using SPN [32], [33], [34]. In [32], an approach using generalized stochastic Petri nets (GSPNs) to model and analyze attack trees is proposed with the ultimate goal of automating the analysis using simulation tools. The results of this simulation and analysis can then be used to further refine the attack tree or to develop corresponding countermeasures. In [33], a vulnerability assessment framework is proposed to systematically evaluate the vulnerabilities of SCADA systems at three levels: system, scenarios, and access points using a generalized stochastic Petri net model. The proposed method is based on cyber systems embedded with the firewall and password models, the primary mode of protection in the power industry. In [34], the authors investigated the use of Petri nets for modeling coordinated cyber-physical attacks on the smart grid. A hierarchical method is provided to construct large Petri nets from a number of smaller Petri nets that can be created separately by different domain experts for analyzing cyber-physical attacks.

VI. Conclusion

In this paper, we have introduced a model of a defensive maneuver cyber platform utilizing Stochastic Petri Nets. By utilizing moving target defense and deceptive defense tactics, we have shown that we can increase the complexity and composition of a parallel and distributed application by increasing the ratio of deceptive to operational nodes and increasing the rate in which nodes transition between states. We have theorized these characteristics can present a more defendable platform due to the changing attack surface. Lastly, we have introduced an automated tool for building configurable Petri Nets for use in tools compatible with the Petri Net Markup Language.

In the future, we will continue to refine our model and add an attacker model to provide improved analytic functions. By using Hadoop as a sample distributed and parallel application, we will investigate building a prototype system on our campus that behaves as indicated by our model. Our current work mainly focuses on distributed and parallel applications. We may generalize our model to support other platforms, even traditional networks. With a prototype system and human and automated penetration testers, we desire to deploy our system into a sandbox to study real world survivability of a system built on this model. The Defensive Maneuver Cyber Platform could be used as a defensive mechanism when the threat condition changes. Using these results, we can further explore how a prototype system could be built that updates the maneuver rates during run-time to build a more deceptive or computation cluster as conditions dictate. We are encouraged by these results and believe they have presented introductory results to encourage further exploration into understanding the behaviors of a system that maneuvers individual nodes between operational, deceptive and idle states while continuing to provide a computational resource to the users and community.

References

[1] "Indicted man pleads guilty to hacking government supercomputers." [Online]. Available: http://www.scmagazine.com/indicted-man-pleads-guilty-to-hacking-government-supercomputers/article/309445/

[2] R. Chirgwin and . M. 2014, "Kiwis unplug supercomputer after intrusion." [Online]. Available: http://www.theregister.co.uk/2014/05/26/kiwis_unplug_niwa_super_after_intrusion_spotted/

[3] L. Nixon, "The stakkato intrusions: What happened and what have we learned?" in *Sixth IEEE International Symposium on Cluster Computing and the Grid, 2006. CCGRID 06*, vol. 2, May 2006, pp. 27–27.

[4] U. S. Army, "Field manual 3-0: Operations," 2008.

[5] S. Applegate, "The principle of maneuver in cyber operations," in *2012 4th International Conference on Cyber Conflict (CYCON)*, 2012, pp. 1–13.

[6] W. J. Lynn, "Defending a new domain: The pentagon's cyberstrategy," *Foreign Affairs*, 2010. [Online]. Available: http://www.jstor.org/stable/20788647

[7] R. Parks and D. Duggan, "Principles of cyberwarfare," *IEEE Security Privacy*, vol. 9, no. 5, pp. 30–35, Sep. 2011.

[8] J. Dressler, W. Moody, J. Koepke, and C. Bowen, "Operational data classes for establishing situational awareness in cyberspace," in *Cyber Conflict (CyCon), 2014 6th International Conference on*, June 2014, pp. 1–8.

[9] D. Raymond, G. Conti, T. Cross, and M. Nowatkowski, "Key terrain in cyberspace: Seeking the high ground," in *Cyber Conflict (CyCon), 2014 6th International Conference on*, June 2014.

[10] W. Tirenin and D. Faatz, "A concept for strategic cyber defense," in *IEEE Military Communications Conference Proceedings, 1999. MILCOM 1999*, vol. 1, 1999, pp. 458–463 vol.1.

[11] D. B. Farmer, "Do the principles of war apply to cyber war?" Tech. Rep., May 2010.

[12] P. K. Singh, "Maneuver warfare in cyberspace," DTIC Document, Tech. Rep., 1997. [Online]. Available: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA525962

[13] P. Beraud, A. Cruz, S. Hassell, J. Sandoval, and J. Wiley, "Cyber defense network maneuver commander," in *2010 IEEE International Carnahan Conference on Security Technology (ICCST)*, Oct. 2010, pp. 112–120.

[14] P. Beraud, A. Cruz, S. Hassell, and S. Meadows, "Using cyber maneuver to improve network resiliency," in *MILITARY COMMUNICATIONS CONFERENCE, 2011 - MILCOM 2011*, Nov. 2011, pp. 1121–1126.

[15] S. Hassell, P. Beraud, A. Cruz, G. Ganga, S. Martin, J. Toennies, P. Vazquez, G. Wright, D. Gomez, F. Pietryka, N. Srivastava, T. Hester, D. Hyde, and B. Mastropietro, "Evaluating network cyber resiliency methods using cyber threat, vulnerability and defense modeling and simulation," in *MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012*, Oct. 2012, pp. 1–6.

[16] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[17] C. Ramchandani, "ANALYSIS OF ASYNCHRONOUS CONCURRENT SYSTEMS BY TIMED PETRI NETS," Massachusetts Institute of Technology, Cambridge, MA, USA, Tech. Rep., 1974.

[18] K. Jensen, "Coloured petri nets," in *Petri Nets: Central Models and Their Properties*, ser. Lecture Notes in Computer Science, W. Brauer, W. Reisig, and G. Rozenberg, Eds. Springer Berlin Heidelberg, Jan. 1987, no. 254, pp. 248–299.

[19] M. Ajmone Marsan, G. Conte, and G. Balbo, "A class of generalized stochastic petri nets for the performance evaluation of multiprocessor systems," *ACM Trans. Comput. Syst.*, vol. 2, no. 2, p. 93122, May 1984.

[20] N. J. Dingle, W. J. Knottenbelt, and T. Suto, "PIPE2: a tool for the performance evaluation of generalised stochastic petri nets," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 4, p. 3439, Mar. 2009.

[21] W. Moody, L. Ngo, E. Duffy, and A. Apon, "JUMMP: Job uninterrupted maneuverable MapReduce platform," *In Proceedings of 2013 IEEE International Conference on Cluster Computing*, Sep. 2013.

[22] L. Spitzner, *Honeypots: tracking hackers*. Addison-Wesley Reading, 2003, vol. 1.

[23] J. Billington, S. Christensen, K. van Hee, E. Kindler, O. Kummer, L. Petrucci, R. Post, C. Stehno, and M. Weber, "The petri net markup language: Concepts, technology, and tools," in *Applications and Theory of Petri Nets 2003*, ser. Lecture Notes in Computer Science, W. van der Aalst and E. Best, Eds. Springer Berlin Heidelberg, 2003, vol. 2679, pp. 483–505.

[24] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving Target Defense*. Springer, 2011.

[25] D. Evans, A. Nguyen-Tuong, and J. Knight, "Effectiveness of moving target defenses," in *Moving Target Defense*. Springer, 2011, pp. 29–48.

[26] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 127–132.

[27] M. Dunlop, S. Groat, W. Urbanski, R. Marchany, and J. Tront, "Mt6d: A moving target ipv6 defense," in *MILITARY COMMUNICATIONS CONFERENCE, 2011-MILCOM 2011*. IEEE, 2011, pp. 1321–1326.

[28] J. Rowe, K. Levitt, T. Demir, and R. Erbacher, "Artificial diversity as maneuvers in a control-theoretic moving target defense," in *Moving Target Research Symposium*, 2012.

[29] E. Al-Shaer, Q. Duan, and J. H. Jafarian, "Random host mutation for moving target defense," in *Security and Privacy in Communication Networks*. Springer, 2013, pp. 310–327.

[30] R. Zhuang, S. Zhang, S. A. DeLoach, X. Ou, and A. Singhal, "Simulation-based approaches to studying effectiveness of moving-target network defense," in *National Symposium on Moving Target Research*, 2012.

[31] J. Yackoski, H. Bullen, X. Yu, and J. Li, "Applying self-shielding dynamics to the network architecture," in *Moving Target Defense II*. Springer, 2013, pp. 97–115.

[32] G. Dalton, R. F. Mills, J. M. Colombi, and R. A. Raines, "Analyzing attack trees using generalized stochastic petri nets," in *Information Assurance Workshop, 2006 IEEE*. IEEE, 2006, pp. 116–123.

[33] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for scada systems," *Power Systems, IEEE Transactions on*, vol. 23, no. 4, pp. 1836–1846, 2008.

[34] T. M. Chen, J. C. Sanchez-Aarnoutse, and J. Buford, "Petri net modeling of cyber-physical attacks on smart grid," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 741–749, 2011.