IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON SOFTWARE DEFINED NETWORKS FOR ENERGY INTERNET AND SMART GRID COMMUNICATIONS

Received April 8, 2019, accepted April 25, 2019, date of publication June 26, 2019, date of current version July 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2924968

A Bayesian Game-Theoretic Intrusion Detection System for Hypervisor-Based Software Defined Networks in Smart Grids

RUMAISA AIMEN NIAZI AND YASIR FAHEEM
Department of Computer Science, COMSATS University Islamabad, Islamabad Campus, Islamabad 45550, Pakistan
Corresponding author: Yasir Faheem (yasir.faheem@comsats.edu.pk)

ABSTRACT The future smart grids (SGs) require advanced capabilities in terms of automation, processing, monitoring, and communication. The most crucial component in the successful sustainability of SGs is communication management. In the vSDNs, a hypervisor is implemented between a physical infrastructure and a control plane that abstracts the underlying SDN infrastructure into multiple isolated virtual slices, i.e., we can have multiple vSDNs each with its controller. For that purpose, the virtualized SDNs offer a promising solution as they offer better network management, programmability, and virtualization. However, vSDN-based SGs are prone to many security issues. To disturb operations of the SGs, the security of the vSDN can be compromised by manipulating the jeopardized switches in the DDoS attacks to repress the resources of vSDN controllers. To prevent the exploitation of a vSDN-based SG architecture and preserve its limited resources, this paper formulates the strategic interaction between a hypervisor monitoring its vSDN controllers and the source of new flow requests potentially launching a DDoS attack, via compromised switches, as a non-cooperative dynamic Bayesian game of intrusion detection. Our game model enables a hypervisor to distribute its limited resources to monitor guest vSDN controllers optimally. The performance evaluation via simulations shows that our game model enables a hypervisor not only to increase the probability of detecting distributed attacks and minimize false positives but at the same time, its monitoring costs get reduced as the allocation of resources to monitor vSDN controllers depends upon its belief about the source of the attacks that it forms based on its observation.

INDEX TERMS Software-defined networks, smart grids, DDoS attacks, hypervisor, Bayesian game theory.

I. INTRODUCTION

A Smart grid (SG) is a critically important infrastructure designed to replace the conventional power grid. The dynamic change in the demand of users requires uninterrupted availability of communication [1]. Thus, the SG environment accommodating several utilities with huge data storage does not only calls for a common platform for data center virtualization. But, also a platform which provides minimal network and power management efforts [2].

The utilization of the Cloud Computing (CC) model can significantly contribute to the SG's computational requirements [3]. However, a grid communication network is a vital component for reliable and efficient transmission of real-time data generated by deployed devices. Emerging cloud-based SDNs can provide better networking opportunities as SDN introduces the abstraction and centralized control, which along with CC can provide dynamism and controllability [4].

The integration of SDNs and cloud computing can provide splendid service delivery platforms. Irrespective of enormous opportunities, a cloud network faces significant network challenges due to the flexible configuration of appliances, policy enforcement complexities, and topology independence. In the meantime, the SDN paradigm contributes significantly to address these problems. SDNs introduce a centralized controller, software-based traffic analysis established on machine learning and statistical techniques, and the dynamic update of forwarding rules. The programmability offered by the SDNs provides support for the accumulation of intelligence from Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs) [5]. Virtualization in the SDN-based SG

The associate editor coordinating the review of this manuscript and approving it for publication was Mubashir Husain Rehmani.

has been introduced to provide network centralization and virtualization [5].

The adaption of vSDNs can significantly enhance the flexibility of SGs. Generally, a SG infrastructure is comprised of thousands of devices to assist in monitoring and control of resources, which further automates the process of operations in a grid. For that purpose, the Supervisory Control and Data Acquisition (SCADA) - distributed system is responsible for the management and monitoring of automated processes and components of a grid. However, to optimally operate SCADA, efficient resource management of the underlying communication network is required. Although a vSDN in a SG will support the resilience system, it is necessary to highlight that the above-discussed loopholes in the CC and the SDNs concurrently transcend to virtualized SDNs based SGs. Above all, it is also essential to realize the successful attainability of a single point management failure in vSDN.

A vSDN provides dynamic network architecture and relatively easier detection of the DDoS attacks due to the network-wide knowledge. It can build secure policies to monitor traffic patterns for attack detection [6]. On the contrary, it exposes the network to many security problems. These threats are categorized based on whether the target of an attack is the application layer, the control layer, or the infrastructure layer [7]. For this reason, the security threats in the vSDN-based SGs can belong to the three main classes: a) devices in a vSDN or applications on a vSDN controller get compromised, b) devices of the SG get compromised [8], and c) a hypervisor in a vSDN either gets compromised or becomes resource-constrained. Nonetheless, the online connectivity of the future smart grids poses cyber-physical threats which can contribute to the potential disruption of power supply by the SGs.

One significant threat to the virtualized SDN-based SGs is a DDoS attack. It is mostly a botnets driven attack, where a handler under the control of an attacker directs its' distributed automated malicious bots to launch an attack on a targeted cyber-physical system [9]. In the vSDN-based SGs, hypervisors and vSDN controllers are the potential targets of the DDoS attacks. These multiple distributed bots can synchronously flood SDN switches with packets resulting in numerous packet-in messages sent to a vSDN controller for flow rules [10]. Similarly, Openflow switches also get compromised and participate in DDoS attacks. A DDoS attack aims to cease services of the authenticated users by flooding a server with fake requests to exhaust resources of the victim controller. Hence, the overutilization of a single vSDN controller affects the hypervisor layer, which degrades the overall performance of several or all its vSDN tenants. As a consequence, the performance of SG operations also degrades. This intrusion makes services unavailable for a specific time resulting in the loss of revenue and additional cost for mitigation in SGs [11]. For instance, from [8], we infer that a compromised switch of a vSDN in a sub-station will have a much severe impact in comparison to a vSDN switch in any Vehicles-to-Grid (V2G) network.

Initially, it was thought that the SDNs could provide a better defense mechanism against the DDoS attacks. However later, the vulnerabilities of the SDNs were discovered which need to get addressed by the research community. In this paper, we consider a scenario in which a DDoS attack source and a hypervisor strategically interact with each other. A DDoS attacker launches attacks on the vSDN controllers via compromised SDN switches to incur maximum damage to vSDN controllers and as a result the SG applications. Contrarily, at the same time, a DDoS attack source wants to remain undetected during monitoring by a hypervisor. On the other hand, a hypervisor aims to optimally distribute its limited monitoring resources over the vSDN controllers it hosts in a way that maximizes the probability of detecting an intrusion if any occurs. However, at the same time, the hypervisor wants to minimize its monitoring cost. We present an optimal resource allocation solution against the detection of DDoS attacks affecting the control plane. Ultimately, we advocate the use of vSDNs for timely and reliable resource management for SG systems in case of DDoS attacks.

The aforementioned strategic interaction between a hypervisor and a possible attack source via comprised switches can be ideally formulated as a game. Algorithmic game theory is a mathematical tool to formulate, analyze and solve the strategic situations of conflict or cooperation in which multiple agents interact in a way that the payoffs that they get are interdependent on the actions of each other. We present a dynamic Bayesian game-theoretic intrusion detection model against the DDoS attacks on the control layer to ensure flexible operations of a vSDN-based SG networks. The proposed game model enables a hypervisor to optimally allocate its resources to monitor the vSDN controller it hosts to detect an intrusion, while at the same time, mitigating severe damage to the functionality of vSDNs.

A. CONTRIBUTIONS

In our proposed work, we ensure to meet the SG's power system resource requirements by incorporating a mechanism which considers the rational behavior of a hypervisor and an attacker which is not only aware of the strategies of its opponent i.e., the defending hypervisor but incorporates a mixed strategy to counter its actions. We present a game-theoretic Bayesian Nash Equilibrium solution, which addresses the issue mentioned above by distributing the available resources to monitor the vSDN controllers hosted by a hypervisor. Given the limited amount of resources available for detection, a hypervisor H monitors each of its tenants by specifying a threshold over the policy composition overhead (as in the case of Compositional hypervisors discussed in Section-II). Based on excessive requests from individual vSDN controller hypervisor optimally distributes detection resources. This mechanism works by forming belief toward the type of switch. A hypervisor establishes this belief while observing the policy formation requests from each of the tenants, which may exceed their pre-defined limits due to legitimate or

illegitimate reasons. Furthermore, the main contributions of this work are summarized as follows:

- A hypervisor based dynamic Bayesian game-theoretic IDS is modeled to build trust towards the source $packet_in$ messages. To the best of our knowledge, this model is the first approach in virtualized SDNs which addresses the problem of limited resources allocated to a hypervisor for intrusion detection.
- It is a resource-aware detection model that enables a hypervisor to efficiently monitor its hosted vSDN controllers by distributing monitoring resources over them and provides maximum detection without overspending the security resources on monitoring. This model addresses the realistic approach of a malicious entity, which via a compromised switch aims to minimize its detection by deviating its behavior between a normal and a malicious entity. In the proposed model, the decision of a hypervisor to optimally distribute its monitoring resources over its vSDN controllers depends upon: 1) worth of each of the hosted vSDN controllers, 2) resource consumption level, 3) cost of monitoring, 4) cost of attack, 5) its belief about the maliciousness of a controller, 6) false alarm rate and 7) the detection rate.
- Performance of the proposed game model is evaluated extensively via simulations. We assess the impact of the following parameters on the strategies of both the players: a) financial worth of the controllers, b) cost of monitoring, c) gain cost ratio, d) detection rate and e) false alarm rate.

II. BACKGROUND

This section provides a brief overview of: a) the architecture of smart grids and their related issues, b) virtualization in the software defined networks, and, c) the impact of DDoS attacks in vSDNs.

A. SMART GRID ARCHITECTURE AND RELATED ISSUES

The traditional grid supports four tasks of power generation, transmission, distribution, and management of produced electricity. In contrast, the future SG is an amalgam of multiple technologies [12]. Whereas, SG resilience hinges on its timely and reliable delivery service. They constitute a layered bi-directional communication network from which the data is collected to be delivered to the main control center through the Wide Area Network (WAN). SCADA systems are considered the main components of smart grids. They are responsible for control, monitoring, and acquisition of data from substations and equipment for energy delivery. This system further comprises the core control center called Mass Terminal Unit (MTU) and widely distributed substations; MTU is the main component to gather information and to send commands to substations.

SDN-based communication infrastructure provides a viable solution to transport massive amounts of data generated by SG devices towards the servers. The SG environment accommodates several utilities with massive data storage which does not only call for a common platform for data center virtualization but, also a platform which provides minimal network and power management efforts [2]. The utilization of the CC model can significantly contribute to the computational requirements of smart grids [3]. However, a communication network is a vital component of the SG environment. Emerging cloud-based SDNs can provide better networking opportunities as SDN introduces the abstraction and centralized control, which along with CC will provide dynamism and controllability [4]. The advancements in digital communication have provided high data transfer rates, P2P communication, programmability, and now virtualization benefits contributed by vSDN. However, SDNs are also prone to DDoS attacks. The cyber-resilient SG requires dynamism, which calls for flexibility, availability, integrity, and confidentiality of data.

B. HYPERVISORS IN VIRTUALIZED SOFTWARE DEFINED NETWORK:

An SDN network can be virtualized by incorporating a hypervisor between the control plane and the SDN data plane. Hypervisors were initially developed in CC to monitor the hosted virtual machines and to allocate resources to them. Analogously, virtualization in SDN-based clouds for virtual networking adapts a hypervisor to monitor virtual networks and to allocate network resources. The operations of multiple virtual networks demand an infrastructure with sufficient resources. A virtualized network in SDNs requires the implementation of a hypervisor between the physical infrastructure and controllers. As illustrated in the Fig 1, the hypervisor gives each virtual SDN (vSDN) controller, the perception of being connected directly with the vSDN network. It provides different levels of abstraction to its tenants, which depends on the characteristics of the underlying network. One such abstraction of the resources of physical nodes includes CPU and flow table resources [13].

One of the design challenges of hypervisor functions in the control plane virtualization is the confinement and protection of the tenants. The isolation is provided for the reliable operation of tenants, whereas, protection contributes to the independent performance of each tenant without network degradation. Isolation is a much needed requirement in the control plane which is highly influenced by the resources available to a hypervisor from their hosting platforms. For instance, computational resources can significantly influence the packet processing performance. Likewise, the storage capacity also restrains the control plain buffering. For this reason, a vSDN requires the confinement and the protection features at both the data and the control planes. Since the hypervisor layer lies between the vSDN controllers and the physical infrastructure, it is pertinent to provide these features for the hypervisor processing resources among multiple tenants.

C. DDOS ATTACK IN COMPOSITIONAL HYPERVISORS

Compositional Hypervisor (CH) is a policy-based hypervisor responsible for constructing Openflow rules based on

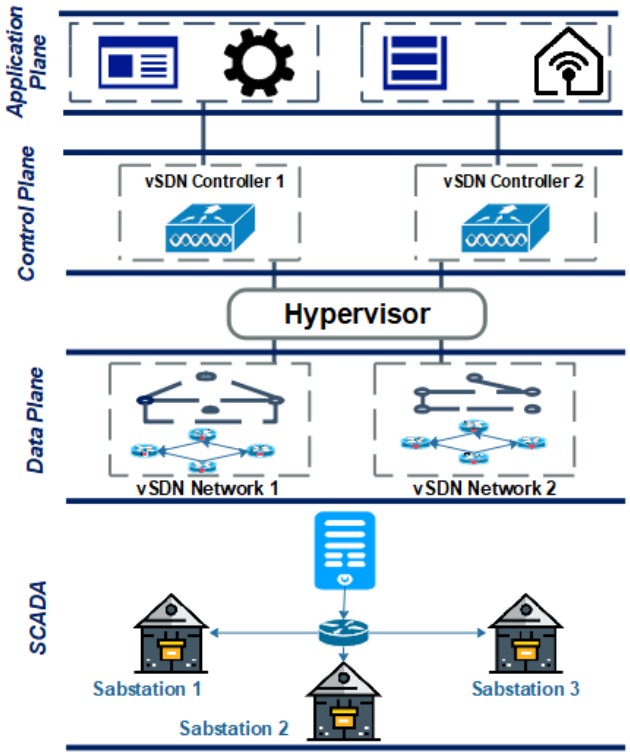


FIGURE 1. Illustration of a virtual software defined networking based smart grid.

inputs received from multiple distinct network applications. The CH focuses on the policy consumption overhead, which includes the computation and rule update based overheads. The computational overhead is due to the time required for calculating a new table, and the rule update overhead is due to the number of control messages sent in order to obtain a new flow table. If any of the hosted controllers get bombarded with new packets, each will send a request for a new flow table, and CH will need to compute a new policy by integrating individual policies which will ultimately increase the policy composition overhead [14].

If Openflow is the protocol employed for an SDN controller, the DDoS attack substantially affects the working of the vSDN controller. In the normal working of an SDN, in case of a new packet with no policies available, the data plane sends queries to the control plane to obtain flow rules [15]. That is, either a packet header or a complete packet is transmitted. Thus, the vast network traffic transmitting complete packets to the controller consumes bandwidth and controller's processing capacity [7]. Due to the high volume of the incoming packets, flow tables will get filled with fake flows, and this behavior will consequently have an impact on the working of CH as CH is responsible for composing policy based on received individual policies of hosted vSDN controllers.

To prevent a DDoS attack from affecting virtualized SDN based SGs, we consider an attack that exploits a switch for generating low-traffic flows to trigger packet-in messages to the controller. This attack will ultimately overburden the hypervisor with policy composition tasks by over-consuming its resources. Thus, ultimately exhausting the processing and storage resources of a hypervisor shall lead to the unavailability of the services. The virtualized SDNs is a growing research area for which the ever-growing threat of DDoS attacks has been largely neglected. In the following section, we discuss the DDoS attack detection mechanisms for SDNs available in the literature.

III. RELATED WORK

The detection of the compromised switches in SDN is a longstanding challenge. Potentially compromised SDN switches can be manipulated to conduct several attacks such as black-hole attacks, man-in-the-middle attacks, and DDoS attacks. The vulnerabilities causing DDoS threats to vSDN infrastructure naturally arise from SDN. Several techniques have been proposed in the literature to detect DDoS attacks in the SDNs. This section discusses the literature on DDoS attacks from two different perspectives. Firstly, as this work addresses the DDoS attacks against the control plane, we review the detection mechanisms of compromised switch-based attacks in the control layer. Secondly, we review the efforts made from the game-theoretic perspective for controller assignment based solutions.

A. DETECTION OF COMPROMISED SWITCH-BASED ATTACKS

In this section, we discuss several approaches introduced to detect compromised switches via which DDoS attacks get launched. We categorize these approaches into statistical solutions, machine learning-based solutions, and techniques that require an additional module for compromised switch detection.

1) STATISTICAL BASED MECHANISMS

These solutions rely on the compilation of statistical features of the normal traffic, which are compared against new incoming traffic to detect an anomaly. In [16], malicious switch monitoring and detection system is proposed for OpenFlow protocol. Using this technique, two kinds of issues, i.e., packet swapper and packet dropper are investigated. For that purpose, two additional functional blocks are added to a SDN controller. One for malicious switch detection and prevention, and another for a policy block containing rules against an identified malicious switch. However, the proposed solution can be extended to other forms of abnormal behavior. Moreover, the packet dropping detection mechanism relies on the statistics reports of the ports received from switches. Hence, the architecture overlooks the impact of falsified information received from dishonest switches.

To address the limitation mentioned above of incorrect information of flow statistics from the faulty switches, a Byzantine model-based prototype is proposed in the [17]. The objective of this model is to tolerate faulty switches automatically. A proxy layer is implemented, which is a

TABLE 1. Summary of related work on detection of compromised switch-based attacks.

Mechanism	Paper	Technique	Strength	Limitations	Attack Detection
Statistical	[16]	A malicious switch monitoring and detection system is proposed for OpenFlow protocol.	Malicious switch detection block and policy blocks containing rules against faulty switch are added for OpenFLow Protocol.	Two additional blocks of limited functionality are added in SDN controller. Detection mechanism relies on port statistics reports received from compromised switches.	Packet swapper, Packet Dropper
	[17]	Prototype system based on Byzantine model.	Resolves the issue of false flow statistics information collection.	A separate proxy layer is implemented. The only objective is to tolerate faulty switches.	DDoS
	[18]	SDN secure controller algorithm.	Addresses ineffective distribution of resources and identifies attackers location.	Algorithm relies on switch port statistics information from a compromised switch.	DDoS
	[10]	Sequential Probability Ratio Test	Detects the compromised interfaces. Reduces false positive and negative error rates.	Monitoring cost of the mechanism is not considered.	Low-rate DDoS
	[19]	Backup controllers audit the network update events information received from controllers and switches.	Detects compromised SDN devices, i.e. switches and controllers.	Suspected devices are designed to support the collection of analyzed information.	Switch based DDoS
Machine Learning	[20]	Hidden Markov Model	Baun-Welch, an unsupervised algorithm is used to train the model whereas Viterbi algorithm is used to predict the status of the observed sequence. A total of twelve typical features of four kinds of attacks are collected.	The model can be improved for more complex scenarios and the parameters can be optimized to provide low error rate.	Flooding attack, a compromised switch based attack, ARP and network scanning attack.
Module Based Mechanisms	[21]	SPHINX is built on a flow graph model.	Provides accurate real time verification of the network behavior. Intercepts Openflow messages between the controller and the switch to build updated flow graphs and to validate network.	Attack detection time is delayed by ingress queues and long time processing of packet in messages. It fails to identify malicious ingress and egress switch.	Switch Based DDoS
	[22]	The middlebox management architecture is proposed to enables the deployment of programmable middleboxes.	Each switch detects and prevents malignant activities locally and reports to the controller. The controller collaborates with the switches to form the final intrusion decision.	A compromised switch can deliberately report false alerts to confuse the controller.	Switch based DDoS

separate function layer, only to resolve the issue of flow statistics information collection. Nevertheless, the question of receiving inaccurate information from the faulty switch is still un-addressed.

To address the limitation of the ineffective distribution of resources between legitimate users and an attacker, and to identify the location of the attackers, a SDN secure controller algorithm is presented in [18]. This algorithm also relies on the information obtained from switch port statistics. Since the controller has a global network view, it gets hold of the location of the source of the attack. However, this mechanism is counterproductive against DDoS attacks, as well as their claim of addressing the problem of a compromised switch is opaque due to the collection of flow statistics information from the compromised switches.

A statistical tool named Sequentially Probability Ratio Test (SPRT) is proposed to detect the compromised interfaces causing low-traffic flow-based DDoS attacks in [10]. The introduced mechanism reduces false positive and negative error rates. However, the monitoring cost of the procedure is not well considered.

To detect the problem of compromised SDN devices, i.e., controllers and switches [19], backup controllers are employed to identify unexpected behaviors among primary controllers and switches. The backup controllers audit the information of network update events received from controllers and switches. Whereas suspected devices are designed to support the collection of analyzed information, yet the rational behavior of compromised devices is not taken into account.

2) MACHINE LEARNING BASED MECHANISMS

Machine learning-based techniques encompass methods based on statistics and data mining. However, there is a subtle difference as they involve performance optimization based on past results. In [20], a Hidden Markov Model (HMM) is introduced for security awareness and quantification of the network based on the provided sequence of observed flow features. A total of twelve typical features of four kinds of attacks are collected that include a flooding attack, a compromised switch-based attack, ARP, and network scanning attack. These features are extracted by building a feature extractor and HMM is used in an assessor to develop a quantification method. Baun-Welch, an unsupervised algorithm is used to train the model, whereas Viterbi algorithm is used to predict the status of an observed sequence. However,

TABLE 2. Summary of related work on game theory based approaches for controller assignment.

Paper	Technique	Strength	Limitations
[23]	Stackelberg's game theoretic model.	IDS presents the identification of bots based on the signature match. Based on reward-punishment model the bandwidth is degraded to an attacker for a limited period of time.	The game does not incorporate the cost of attack and monitoring of the participating players.
[24]	A two phased dynamic mechanism is proposed to minimize the controller response time and traffic overhead. This problem is formulated as a stable matching problem.	Improves the response time which is reduced up to 86%.	The model does not consider the rational nature of the players.
[25]	Zero-sum based decision making approach to alleviate the performance of the overloaded controller.	The switches are traded among competing controllers to increase their payoffs using load balancing.	Fails to incorporate the communication cost among controllers.
[26]	Bargaining game is modelled to obtain multi-objective optimal placement of controllers based on: a) the communication overhead and latency between switches and a controller, and, b) between controllers and guaranteed load balancing between controllers.	A cooperative Nash bargaining game model is used to and a unique solution that satisfies maximum utility for both players.	Causes huge communication overhead between controllers.

the model can be improved for more complex scenarios, and the parameters can be optimized to provide a low error rate.

3) MODULE BASED MECHANISMS

Apart from the frameworks discussed above, we present techniques that incorporate additional modules either in a controller or in an infrastructure layer. Authors in [21], address the network topology and the data plane forwarding attacks mounted by jeopardized switches and hosts. SPHINX is built on a flow graph model which provides accurate real-time verification of the network behavior by focusing on the following four types of messages to get the metadata: 1) the *packet in*, 2) *features reply*, 3) *flow mod* and 4) *stats reply*. The SPHINX intercepts Openflow messages between the controller and the switch to build updated flow graphs and to validate the network. However, the attack detection time is delayed by ingress queues and long processing time of the *packet in* messages. Also, it fails to identify malicious ingress and egress switch.

A middlebox management architecture is proposed in [22], to extend the functionality of the Openflow switches. It enables the deployment of programmable middleboxes to support the intrusion detection and the intrusion prevention systems. The Openflow extends its support for ClickOS VMs which serve as the isolated middelboxes in the switches. These middleboxes are configured by a controller and are executed in a distributive manner in VMs of the switches. Each switch detects and prevents malignant activities locally and reports to the controller. The controller receives multiple alerts of the intrusion from switches and collaborates with them to form the final intrusion decision. However, the compromised switches can deliberately report false alerts to confuse the controller. In this case, the collaboration of alerts prevents false alarms. Furthermore, compromised switches are eliminated to reduce the false positive and negative rate of alert collaboration.

From the above-discussed literature, we infer that most of the mechanisms rely on the statistical flow-based information obtained from suspected switches which are not a reliable source of information; an infected switch can provide false reports and may variate its behavior to minimize its chances of detection. Moreover, some of the techniques introduce an additional layered approach in SDNs due to which we claim that SDN virtualization can provide an effective and efficient mechanism over an additional layer with limited functionality.

B. GAME THEORY BASED APPROACHES FOR CONTROLLER ASSIGNMENT

Game theory provides a mathematical tool to formulate, analyze and solve the situations of strategic interactions in which the payoffs the decision makers get are interdependent on their actions. For instance, in the attack-defense scenario, the payoff to an attacker increases when it causes maximum damage, but without being caught by the defender to avoid penalties. Similarly, the payoff of the defender increases when it maximizes the chances of detecting an intrusion if it happens, but by allocating as minimum monitoring resources as possible because of the associated costs. Such non-cooperative strategic situations of conflict can ideally be modeled using game theory. Realizing its importance, various problems related to SDNs such as controller assignment, mitigation, and decision making have been modeled as games. In this sub-section, we highlight some of the game-theoretic controller assignment based approaches proposed for the security issues of the control layer in SDNs. In the following discussed literature, a game is played between various components of SDN infrastructure as players to maximize their expected utilities.

In [23], authors incorporate dynamic Stackelberg's game model and countermeasure selection. Since the administrator has control over resource optimization, the framework provides a reward and punishment model for network

bandwidth consumption such that it degrades the bandwidth to an attacker for a limited duration. The IDS presents the identification of bots based on the signature match. In order to implement a punishment mechanism to restrict the attacker for future malicious activities, authors have implemented the Nash Folk theorem, whereas the framework is implemented on top of an ODL controller which functions through the northbound API and evaluations are performed using Mininet.

A two-phased dynamic SDN controller assignment mechanism is proposed to minimize the controller response time and traffic overhead in [24]. This problem is formulated as a stable matching problem and coalitional formation game. The stable matching guarantees the improvement in response time, which is reduced by up to 86 percent. Whereas, a group of switches associated with the controller are perceived as a coalition that can negotiate to improve their response time.

A zero-sum based decision-making approach is introduced by [25] to elect the master controller for migrated switches. This mechanism is designed to alleviate the performance of the overloaded controller. The migrated switches are commodities while controllers are the players. The competing controllers use load balancing to trade commodities in order to increase their payoffs.

In [26], a bargaining game is modeled to obtain multi-objective optimal placement of controllers based on: a) the communication overhead and latency between switches and a controller, and, b) between controllers and guaranteed load balancing between controllers. Due to the conflicting objectives, a cooperative Nash bargaining game model is used to find a unique solution that satisfies maximum utility for both players.

From the above discussion, we can deduce that the objective of all the players is to maximize their payoffs. Nevertheless, none of them address the varying type of concerned players. In conclusion, SDNs lack game-theoretic IDSs for DDoS, and most of the approaches focus on optimizing the response time or performance of overloaded controllers.

Conclusion: This section provided a comprehensive overview of the existing detection and mitigation schemes against the DDoS attacks in the control plane of SDNs. From it, we conclude that these solutions do not consider the limited controller resources for optimal performance, as well as, they are based on the false assumption that the compromised switch will always behave maliciously. In addition, the literature lacks a vSDNs based solution for the problem of compromised switches aiding DDoS attack. While no solution exists to resolve the rational behavior of compromised switches i.e., being a malicious agent, it will try to hide its presence to induce maximum damage. Hence, we provide a hypervisor-based dynamic bayesian game-theoretic IDS for vSDN, which eliminates the need for an additional layer of the limited functionality [17] and allocates monitoring resources to vSDN controllers efficiently. Overall, the existing methods seek to maintain the performance of the overloaded network by adding more to the consumption of resources. Our work is the first in the domain of vSDNs to provide optimal load distribution for the detection of the compromised switches participating in the DDoS attack.

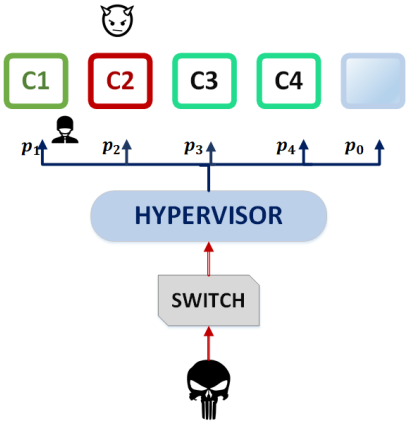
TABLE 3. Parameters.

Notations	Significance
H:	Hypervisor
D:	DDoS Attacker
k_i:	i^{th} controller
a_j:	Action attack on j^{th} vSDN controller
m_i:	Action monitor for i^{th} vSDN controller
w^i:	Financial worth of the i^{th} vSDN controller
R:	Total resources available to H for resource distribution
c_e^j:	Cost of Penalty on detection by DDoS Attacker
c_m^i:	Cost of monitoring i^{th} vSDN controller by **H**
c_a^j:	Cost of Attack on j^{th} vSDN controller by DDoS Attacker
p_i:	Probability of monitoring i^{th} vSDN controller
q_j:	Probability of attacking j^{th} vSDN controller
U_D:	Utility of DDoS Attacker
U_H:	Utility of Hypervisor
μ_0:	Prior Belief of **H** about the type of Switch

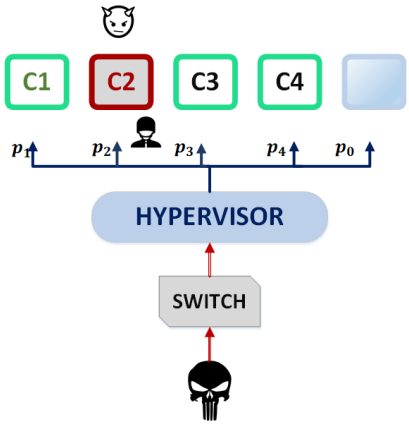
IV. GAME MODEL

In this section, we formulate the strategic interaction between a hypervisor H defending its set of vSDN controllers against a DDoS attack on them, launched via a compromised switch (S), by an attacker *D*. A hypervisor aims to maximize its payoff by detecting any intrusion if it occurs, however, at the same time, it wants to minimize the costs it incurs due to the allocation of its limited resources to monitor vSDN controllers. On the other hand, via a compromised switch, the aim of a DDoS attacker *D* is to maximize its utility by exhausting, as much as possible, the resources of the vSDN controllers hosted by *H* to interrupt SG applications. Contrarily, it wants to remain undetected during monitoring by *H* for which it has to reduce its intensity of the attack. This is a strategic interaction of incomplete information as the hypervisor *H* does not know whether it is monitoring a set of vSDN controllers that are experiencing resource exhaustion due to legitimate requests from un-compromised switches, or, due to an attack launched via a compromised switch. We formulate this scenario as a non-cooperative reward and penalty based dynamic Bayesian game of intrusion detection between a Hypervisor *H* and a switch which might be compromised by a DDoS attacker *D* to launch attacks.

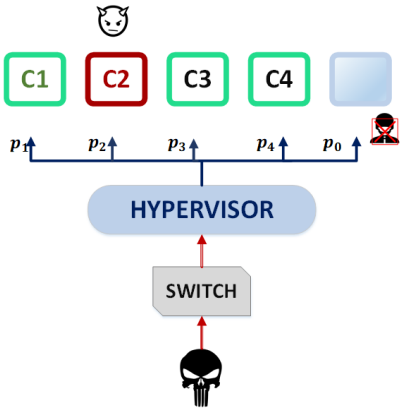
A DDoS attacker allocates its resources to exhaust resources of the hosted controllers and to compromise the normal functioning of the vSDN infrastructure. This process gives this malicious entity *D*, a payoff which depends on worth w^i of the compromised vSDN controller k_i, as well as the cost of attack c_a which depends on the resources the



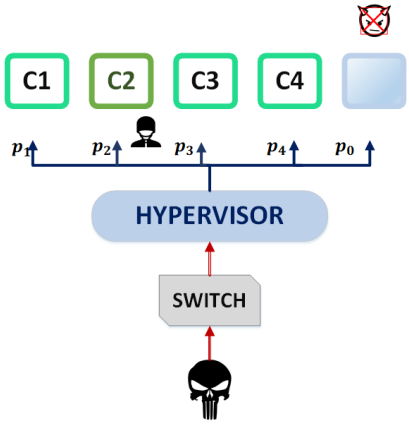
(a) (m_i, a_j): H monitors the controller k_i & D attacks k_j, where i≠j



(b) $(m_{i=j}, a_j)$: H monitors k_i & detects an intrusion by D which attacks the same controller



(c) (m_0, a_j): H does not monitor any controller and an intrusion occurs on a controller k_j by D



(d) (m_i, a_0): H monitors a controller k_i and D takes the action *not-attack*

FIGURE 2. **Strategic interaction between a hypervisor *H* and an attacker *D*.**

attacker D allocates to k_i to carry out the attack. A successful attack benefits the attacker to propagate its attack surface area to the SG infrastructure making its services either unavailable or damages their QoS satisfaction. Whereas, the defender aims to monitor hosted vSDN controllers *packet in* messages to prevent disruption in SG operations. Since the hypervisor H allocates these resources to the hosted vSDN controllers, it monitors each vSDN controller directly from the infrastructure; it is not possible for the vSDN controller to hide resource exhaustion. Nevertheless, the hypervisor may decide to remain idle by not monitoring any of the vSDN controllers. Similarly, a vSDN controller which is not entirely compromised yet provides information to the hypervisor about the compromised switch sending queries for flow rules.

We consider a scenario where a hypervisor H is hosting a set K of N number of vSDN controllers. By $K = \{k_1, k_2, \ldots, k_N\}$, we represent the set of vSDN controllers protected by the H, whereas R is the total resources available to H for distribution among the N hosted vSDN controllers. Moreover, the probability with which a hypervisor monitors any controller k_i is presented by p_i, while the probability with which D attacks any controller k_j is represented by q_j. Similarly, we represent the hypervisor's probability of not monitoring any of the hosted controllers by p_0, and the attacker's probability of not attacking any controller by q_0.

The fig-2 depicts the four possible example scenarios of the strategic interaction between a Hypervisor H and an attacker D which launches attacks on the controllers via a compromised switch. The fig-2a depicts a scenario in which the hypervisor monitors the controller k_i, and the attack goes undetected because D launches an attack on a different controller k_j, i.e., $i \neq j$; this results in the loss of revenue and resources to the hypervisor. On the other hand, fig-2b depicts the scenario where the hypervisor monitors the same controller that is under attack, i.e., $i = j$, resulting in successful detection of the intrusion by the hypervisor. Moreover, fig-2c presents a scenario where a hypervisor does not monitor any controller with the probability p_0, while D attacks a

TABLE 4. Payoff matrix for the game between a hypervisor and an attacker of type $T_S = 1$.

S/H	Monitor	Idle
Attack	$-c_e^j - c_a^j, -\epsilon R - c_m^i$	$c_a^j + w^i, -w^i - R$
Not Attack	$0, -c_m^i$	$0, 0$

TABLE 5. Payoff matrix for the game between a hypervisor and a regular player of type $T_S = 0$.

S/H	Monitor	Idle
Not Attack	$0, -c_m^i$	$0, 0$

controller k_j with the probability q_j. Lastly, fig-2d portrays the scenario, where D does not attack any of the controllers with probability q_0, whereas, the hypervisor monitors a controller k_i with the probability p_i.

Note that the proposed model is based assuming that a threshold is defined over the number of *packet in* messages from each switch and a list received at t_1 and t_2 is aggregated. If the count of the received requests exceeds a pre-defined threshold, the hypervisor notifies the affected controller to isolate the infected switch. However, it is out of the scope of this paper; the focus of our methodology is to identify malicious intruders through a Bayesian game model and to provide an optimal load distribution mechanism for uninterrupted SG operations.

From the assumption mentioned above, we define a penalty c_e^j to the attacker once the hypervisor detects the malicious switch. This penalty presents the loss the attacker suffers from the immediate isolation of an infected switch by the vSDN controller. c_e^j may represent the cost an attacker may have borne to infect the switches or any other associated costs such as to infiltrate the vSDN or the bots via it launched the DDoS attack. Moreover, the resource exhaustion at the vSDN may occur temporarily due to legitimate requests from a legitimate user or malicious requests from a compromised switch. Hence, the hypervisor keeps updating its belief for a particular switch using the proposed dynamic Bayesian game model.

In the following subsections, we discuss the payoff matrices in the Tables 4 and 5 presenting the reward and the penalty of the players, which results due to their strategic interaction. It is pertinent to note that the hypervisor does not know whether it is monitoring a set of vSDN controllers which have been compromised via a DDoS attack, or, it is monitoring the set of vSDN controllers on which no attack is launched. Thus, the hypervisor is not sure whether it is playing the game against a compromised vSDN as depicted in the table 4, or, the one depicted in the table 5. For this reason, we have modeled this strategic interaction as a non-cooperative Bayesian game; the hypervisor associates a belief μ_0 about the game the being played; with the probability μ_0, it assumes the game presented in the table 4 is being played, and with the probability 1-μ_0 it, it assumes the game depicted in the table 5 is played. The hypervisor will update its belief based on its observations. Next, we calculate the expected utilities to the players.

A. EXPECTED PAYOFF OF AN ATTACKER

The objective of a DDoS attacker D exploiting a compromised switch is to distribute its resources optimally on the hosted set of vSDN controllers managed by a single hypervisor such that it can minimize its detection probability by the hypervisor, and to consequently flood the hosted vSDN controllers managing SG applications with unnecessary requests. Hence, the attacker will assign each compromised device the task of flooding *packet in* messages to each vSDN controller. The attack succeeds when it drains the resources of the attacked vSDN controller, which ultimately affects the performance of the hypervisor to allocate resources to the rest of the hosted vSDN controllers and the SG applications.

We now discuss the payoffs of an attacker illustrated in a payoff matrix in Table 4. They represent the payoffs which include the reward, cost and the penalty an attacker may get due to its strategic interaction with the hypervisor in a non-cooperative game. An attacker will gain a payoff of w^i as the worth of the successfully attacked vSDN controller k_i along with the gain R due to successful resource consumption of the hypervisor, minus the cost c_a^i of the attack; this is shown in the fig 2a. However, if the malignant switch gets identified for an attack, as in the case, when the belief of H towards its type increases, the attacker receives a penalty of c_e^i together with the cost of the attack, as depicted in the fig 2b. This penalty is significantly higher than the reward the attacker achieves in the case of a successful attack. In other words, the identified malevolent switch gets permanently blocked from sending the flow rule requests to any vSDN controller will cause a significant loss c_e^i to the manipulative attacker.

Let q_0 be the probability with which an attacker D does not overload vSDN controller with unnecessary requests, while with probability q_j, it attacks the vSDN controller k_j, such that, $1 \leq j \leq N$. The total set of actions available to D are $A = \{a_0, a_1, a_2, \ldots, a_N\}$, where a_0 depicts the action *not attack* by the DDoS attacker on any of the controllers. Likewise, the actions $\{a_1, a_2, \ldots, a_N\}$ represent *attack* action on j^{th} vSDN controller. The action of the attacker which results in obtaining a negative payoff is limited to the scenario where the attacker gets identified for its malicious behavior and gets isolated from sending any form of requests to the vSDN controllers hosted by the hypervisor. Thus, diminishing the consequences of a successful DDoS attack in vSDN based SG. We now determine the expected utility of the attacker D based on its type and strategic behavior.

$$U_D(a_j) = \mu_0[U(a_j, m_{i=j}) + \sum_i^N U(a_j, m_{i\neq j})] + (1 - \mu_0) * 0$$

$$U_D(a_j) = \mu_0[U(a_j, m_{i=j}) + \sum_i^N U(a_j, m_{i\neq j})] \quad (1)$$

By replacing the above utilities $U(a_j, m_{i=j})$ and $U(a_j, m_{i\neq j})$ with payoffs in tables 4 and 5, we obtain the expected utility of the attacker for action *attack* as:

$$U_D(a_j) = \mu_0[q_j p_{i=j}(-c_e^j - c_a^j) + q_j \sum_{i\neq j}^{N} p_{i\neq j}(w^j + R - c_a^j) + q_j(1 - \sum_{i}^{N} p_i)(w^j + R - c_a^j)] \quad (2)$$

where k_i is the vSDN controller monitored by the hypervisor such that $1 \leq i \leq N$.

In this paper, we address the scenario where, in each round, a hypervisor monitors one of the vSDN controllers it hosts, and an attack is launched by a DDoS attack source on only one of the vSDN controllers as well. Hence in each round, the payoff of the attacker would be the worth of an undetected vSDN controller; one such scenario is depicted in fig 2. The summation in equation (1) represents the utility of an attacker in a scenario where the monitored vSDN controller is not the one which gets attacked, i.e., $i \neq j$. Moreover, the summation also includes the scenario when the attack does not get identified due to the hypervisors decision of remaining *idle*. Whereas, $U(a_j, m_{i=j})$ in the above equation 1 demonstrates the payoff of an attacker for a scenario when it attacks the vSDN controller k_j, and it gets caught by the hypervisor because it monitors the same controller. So happens when $i = j$. However, the attacker does not suffer any loss in the case of normal behavior. Thus the utility of the attacker for the action *not-attack* is given as;

$$U_D(a_0) = \mu_0[0] + (1 - \mu_0)(0)) = 0 \quad (3)$$

We now calculate expected utility of the DDoS attacker by adding utilities obtained in equation 2 and 3 for all actions of *attack* and *not attack* on the hosted vSDN controllers such that;

$$U_D = \sum_{j=1}^{N} U_D(a_j) + U_D(a_0) = \mu_0 \sum_{j=1}^{N}[-p_j q_j\{c_e^j + w^j + R^j\} + q_j\{w^j + R^j - c_a^j\}] \quad (4)$$

B. EXPECTED PAYOFF OF A HYPERVISOR

In order to optimally distribute the limited monitoring resources, the hypervisor H has to choose a mixed strategy. The set of actions available to a hypervisor are $M = \{m_0, m_1, \ldots, m_N\}$, where m_0 is the action not to monitor any hosted controller. Likewise, m_i represents the action to monitor the guest controller k_i. The payoff the hypervisor obtains for the successful detection of a compromised switch S is the value of the consumption resources that have been left i.e., $R(1 - \epsilon)$ as well as, the worth of the attacked vSDN controller w^i, illustrated in fig 2b. On the other hand, as depicted in fig 2a and 2c, it has to suffer the loss of vSDN controllers worth w^i along with the available resources in case it has been monitoring a vSDN controller other than the one attacked i.e., $\sum_j^N U_H(m_i, a_{j\neq i})$, or, it has not monitored any vSDN controller at all. Hence, any action of a hypervisor in this static Bayesian game will ultimately impact the service delivery of the vSDN based SGs.

As each vSDN controller has different worth and resource consumption, we take into account the action to monitor each hosted controller as an independent action of the defender. Let p_0 be the probability that a hypervisor does not monitor any of the hosted vSDN controllers and with probability p_i, it monitors vSDN controller k_i, where $1 \leq i \leq N$. We now calculate the expected payoff of the hypervisor by determining the utilities for each of its actions. These utilities are the result of a static Bayesian game played between a hypervisor and an attacker.

$$U_H(m_i) = \mu_0[\sum_{j=1}^{N} U(m_i, a_j) + U(m_i, a_0)] + (1 - \mu_0)[U_H(m_i, a_0)] \quad (5)$$

where j is the vSDN controller which has been attacked by the malicious switch such that $1 \leq j \leq n$. Hence, if H monitors the same vSDN controller which D has attacked, i.e. $i = j$, or if the attacked vSDN controller j is not monitored by the hypervisor, i.e. $i \neq j$, then the expected utilities are formulated as;

$$U_H(m_i) = \mu_0[U_H(m_{i=j}, a_j) + \sum_{j}^{N} U_H(m_i, a_{j\neq i})] + (1 - \mu_0)[U_H(m_i, a_0)]$$

By replacing above utilities w.r.t each scenario with their respective payoffs as presented in Tables 4 and 5;

$$U_H(m_i) = \mu_0 p_i[q_{i=j}(w^{i=j} - \epsilon R - c_m^i) + \sum_{j\neq i}^{N}\{q_j(-w^j - R - c_m^i)\} + (1 - \sum_{j=1}^{N} q_j)(-c_m^i)] + (1 - \mu_0)p_i[-c_m^i] \quad (6)$$

The expected utility of a hypervisor when it does not monitor any of the guest vSDN controllers is calculated as:

$$U_H(m_0) = \mu_0[\sum_{j=1}^{N}\{U_H(m_0, a_j)\} + U_H(m_0, a_0)] + (1 - \mu_0)[U_H(m_0, a_0)] \quad (7)$$

We now input the expected payoffs as illustrated in Table 4:

$$U_H(m_0) = \mu_0 p_0[\sum_{j=1}^{N}\{q_j(-w^j - R)\} + 0] + (1 - \mu_0)p_0 * 0 = \mu_0 p_0[\sum_{j=1}^{N}\{q_j(-w^j - R)\}] \quad (8)$$

From equations 6 and 8, we obtain the expected payoff of H w.r.t each of the possible actions. The final utility of H is obtained by combining the above-formulated equations as:

$$\begin{aligned} U_H &= \sum_{i=1}^{N} U_H(m_i) + U_H(m_0) \\ &= \sum_{i=0}^{N} p_i[\mu_0 q_{i=j}\{2w^{i=j} - R(1+\epsilon)\} \\ &\quad + \mu_0 \sum_{j\neq i}^{N} q_j\{-w^j - R\}] - \sum_{i=1}^{N} p_i(c_m^i) \end{aligned} \tag{9}$$

In this section, we formulated the total expected utilities of both the hypervisor and the DDoS attacker. These payoffs are obtained based on a static Bayesian game played between them. However, in reality, this strategic interaction between a hypervisor and a DDoS attacker shall occur repeatedly. H builds these strategies based on its observation of the action profile of its opponent i.e., history. In the following section, we will discuss a dynamic Bayesian game model that sequentially updates the belief of a hypervisor against the true nature of the attacker.

V. DYNAMIC BAYESIAN GAME OF INTRUSION DETECTION

We extend this extensive form of the game from one-stage to a multi-stage dynamic game model, where a one-stage game is played repeatedly in each time slot t_n where $n\epsilon\{1, 2, \ldots.\}$. This extension resolves the problem of assigning an accurate probability towards the type of the player and accommodate defender by updating its belief as the game evolves. We incorporate the Bayes' law as the defenders' belief updating mechanism.

The proposed Bayesian game is an incomplete-information extensive-form game. In our scenario, the hypervisor does not have exact information about the type of the switch. Whereas, the switch is well aware of its type T_D, which is a private information $T_D = \{0, 1\}$, where $T_D = \{0\}$ implies the switch is compromised by an attacker D and $T_D = \{1\}1$ implies it is not. On the contrary, a hypervisor (defender) has only one type, i.e. *regular*, which is a common knowledge $T_H = \{1\}$. The belief of the defender is updated at the end of each stage, which depends on the history profile of the behavior strategy of its opponent $h_D^H(t_n)$, the prior belief $\mu_0(a_D(t_n)|T_S, h_D^H(t_n))$ about the type of the opponent and action of the player at a stage game t_n. In the equation 10, $h_H^D(t_n)$ represents the action history profile of the defender H at the game stage t_n.

$$h_D^H(t_n) = \{a_D(t_0), a_D(t_1), \ldots, a_D(t_{n-1}))\} \tag{10}$$

The belief is updated by determining the conditional probabilities $P(a_D(t_n)|T_D, h_D^H(t_n))$. In the conditional probabilities, we incorporate the impact of the detection rate α and the false alarm rate β, where $1-\alpha$ and $1-\beta$ represent false negative and true negative rates respectively. These conditional probabilities help in determining the actions the opponent is about to take given the history profile and the type of an attacker in the previous round, which are later incorporated in Baye's law to estimate the belief of H towards the type of a compromised switch. Following are the equations which help to build a belief system:

$$P(a_D(t_n) = a_0|T_D = 1, h_D^H(t_n)) = \alpha * \sum_{i=1}^{N} q_j + \beta * (1 - \sum_{i=1}^{N} q_j) \tag{11}$$

$$\begin{aligned} P(a_D(t_n) &= a_0|T_D = 1, h_D^H(t_n)) \\ &= (1-\alpha) * \sum_{i=1}^{N} q_j + (1-\beta) * (1 - \sum_{i=1}^{N} q_j) \end{aligned} \tag{12}$$

$$P(a_D(t_n) = a_j|T_D = 0, h_D^H(t_n)) = \beta \tag{13}$$

$$P(a_D(t_n) = a_0|T_D = 0, h_D^H(t_n)) = 1 - \beta \tag{14}$$

We use the Bayes' law to construct the belief update system by the hypervisor, in order to update its belief from the stage game t_n to t_{n+1}. Hence, the belief about the type of S is updated at the end of the game as follows:

$$\begin{aligned} &\mu_H(T_D|a_D(t_n), h_D^H(t_n)) \\ &\quad = \frac{\mu_H(T_D|h_D^H(t_n))P(a_s(t_n)|T_D, h_d^H(t_n))}{\sum \mu_H(T_D|h_D^H(t_n))P(a_s(t_n)|T_D, h_D^H(t_n))} \end{aligned} \tag{15}$$

The above equation (15) represents the posterior belief of the hypervisor. $\mu_H(T_D|h_D^H(t_n))$ exhibits the prior belief of H which provides information about the type of D based on the history profile $h_D^H(t_n)$. Whereas, the second factor represents the probability of choosing an action $a_D(t_n)$ by D against H given the type and the history profile at the stage game t_n.

In this section, we formulated the dynamic Bayesian game model of intrusion detection between a hypervisor H and an attack source D. This model offers a realistic approach as it allows a hypervisor to update its belief towards the type of its opponent in an incomplete information game. This belief update allows each player with the opportunity to optimize their strategies by continuously varying their actions. This implies that both players are rational, which is an obvious assumption in an attacker-defender scenario. The hypervisor will prefer to maximize its payoff by not overspending its detection resources. At the same time, an attacker via a compromised switch will play a strategy that minimizes its chances of detection, while at the same time depleting the resources of vSDN to sabotage SG applications.

In the next section, we determine the Bayesian Nash Equilibrium of the proposed game model and calculate the best response strategies for both the players.

VI. BAYESIAN NASH EQUILIBRIUM

To gain maximum payoff, both players have to best respond to each others' actions by optimizing probability distributions over their available actions. A hypervisor wants maximize the probability to detect an intrusion if one occurs on any of its hosted vSDN controllers, while at the same time it wants to minimize monitoring costs; a hypervisor can conserve some

of its resources by not monitoring any of its hosted vSDN controllers or by monitoring one of them. Similarly, an attacker will prefer to attain maximum payoff without being detected. In order to identify the optimal utilities of both the players, we analyze the extensive form of our Bayesian game. The development of a dynamic Bayesian game also requires us to determine its Bayesian Nash Equilibrium (BNE). For that purpose, we need to find the probabilities of p_i and q_j w.r.t each vSDN controller, such that neither of the players have a strategy by individually deviating to which they can increase their payoff.

A. PURE STRATEGY EQUILIBRIUM ANALYSIS

Before we begin with the one-stage Bayesian game equilibrium analysis, we first determine the pure strategies Nash equilibrium. Since the type of the attacker is its private information, the pure strategies of σ_S are dependent on its true nature. We categorize each action of a sender σ_D based on its type such that $\sigma_D = \{(a_1\ ifmalicious, .., a_N ifmalicious, a_0\ ifregular), a_0\}$, where $a_1, a_2, a_3, \ldots, a_N$ represent action the *attack* on the Nth vSDN controller hosted by a hypervisor. For hypervisor, the strategy set is defined as $\sigma_H = \{m_1, m_2, \ldots, m_N, m_0\}$. In order to perform our analysis for the dynamic Bayesian game Nash equilibrium, we first figure out the pure strategy Nash equilibrium (PSNE). We consider that there exists a belief threshold μ_0', such that for $\mu_0 > \mu_0'$ no pure strategy Nash equilibrium exits. We hereby analyze the following cases.

Case-I: $\sigma_D = \{(a_j\ if\ malicious, a_0\ if\ regular), a_0\}$ for the attacker while $\sigma_H = \{m_{i\neq j}\}$. The expected payoff in this scenario is:

$$\begin{aligned} U_H(m_{i\neq j}) &= \mu_0[-w^{j\neq i} - R - c_m^i] + (1-\mu_0)[-c_m^i] \\ &= \mu_0[-w^{j\neq i} - R] - c_m^i \end{aligned} \tag{16}$$

If $\sigma_H = \{m_{i=j}\}$, the expected payoff is,

$$U_H(m_{i=j}) = \mu_0[w^{i=j} - \epsilon R] - c_m^{i=j} \tag{17}$$

For$\sigma_H = \{m_0\}$, the expected payoff is,

$$U_H(m_0) = \mu_0[-w^j - R] \tag{18}$$

Now if (17)>(16) the dominant strategy for the hypervisor is to monitor. However, the best response for the attacker is to not attack. Hence, $(\sigma_D, \sigma_H) = \{(a_j\ if\ malicious, a_0\ if\ regular), m_{i=j}\}$ is not a Bayesian Nash equilibrium under the condition $\mu_0 < \frac{c_m^{i=j} - c_m^{i\neq j}}{w^{i=j} + w^{i\neq j} - \epsilon R^i + R^j)}$.

If (18) > (17) or (16) > (17), the best response for the attacker is to attack. Hence, $(\sigma_D, \sigma_H) = \{(a_j\ if\ malicious, a_0\ if\ regular), m_0\}$ and $(\sigma_D, \sigma_H) = \{(a_j\ if\ malicious, a_0\ if\ regular), m_{i\neq j}\}$ is a Bayesian Nash equilibrium under the conditions $\mu_0 = \frac{c^{i=j}}{w^{i=j} + w^j + R^j - \epsilon R^i}$ and $\mu_0 > \frac{c_m^{i=j} - c_m^{i\neq j}}{w^{i=j} + w^{i\neq j} + R^j - \epsilon R^i)}$. However, this does not depict a realistic scenario, as it will require H to remain either idle or not to monitor the attacked vSDN controller.

Case-II: $\sigma_D = \{a_0\}$ for the attacker while $\sigma_H = \{NotMonitor\}$. If the hypervisor decides to remain idle the best response of the attacker is to attack; this will lead us to the case when (16)>(17) or (18) >(17). Therefore, there is no pure strategy Nash equilibrium in either of the above cases.

Above discussed cases depict that due to uncertainty towards the type of the player, no pure strategy Nash equilibrium exits. Hence, it is desirable to figure out the mixed strategy Nash equilibrium.

B. MIXED STRATEGIES BAYESIAN NASH EQUILIBRIUM

This section determines the mixed strategy Nash equilibrium (MSNE) of the proposed game model. We also derive the optimal probability distributions of both the players over their actions at the MSNE. The MSNE enables the associated players to optimize their strategies based on probability distribution to maximize their payoffs.

1) PROBABILITY DISTRIBUTIONS OF AN ATTACKER OVER ITS ACTIONS

In order to figure out mixed strategy Nash equilibrium (MSNE), we first present the probability distributions of a malicious source of *packet in* messages q_j and q_0 when the hypervisor plays its best response strategy. Thus, neither of the players deviate from their response as none of them can enhance their payoffs.

Let θ represents the negative payoff of a hypervisor, which depends on the financial worth of the attacked vSDN controllers along with the total resource consumption R. For MSNE, the hypervisor must be indifferent between choosing its actions, irrespective of the different characteristics of hosted vSDN controllers. Thus the utility of monitoring any hosted vSDN controller is equivalent to others.

$$U_H(m_N) = U_H(m_{N-1}) =, \ldots, U_H(m_1) = U_H(m_0) \tag{19}$$

Since all utilities are equivalent, we consider

$$U_H(m_i) = U_H(m_0) \tag{20}$$

From equation 6 and 8, we know that;

$$\begin{aligned} mu_0[q_{i=j}(w^{i=j} - \epsilon R - c_m^i) + \sum_{j\neq i}^{N}\{q_j(-w^j - R - c_m^i)\} \\ + (1 - \sum_{j=1}^{N} q_j)(-c_m^i)] + (1-\mu_0)[-c_m^i] \\ = \mu_0[\sum_{j=1}^{N}\{q_j(-w^j - R)\}] \end{aligned} \tag{21}$$

Solving the above equation results in:

$$\begin{aligned} \mu_0[q_{i=j}(w^{i=j} - \epsilon R - c_m^i) - \sum_{j\neq 1}^{N} q_j(w^j + R) + \sum_{j=1}^{N} q_j(w^j + R)] \\ - c_m^i = 0 \end{aligned}$$

$$\mu_0[q_{i=j}(w^{i=j} - \epsilon R - c_m^i) + q_{i=j}(w^j + R) - \sum_{j\neq i}^{N} q_j(w^j + R) + \sum_{j\neq i}^{N} q_j(w^j + R)] - c_m^i = 0 \quad (22)$$

$$q_{i=j} = \frac{c_m^i}{\mu_0[R(1-\epsilon) + 2w^j]} \quad (23)$$

Equation 23 calculates the probability of the attacker for attacking the vSDN controller k_j. The equation demonstrates that the probability increases with the increase in the cost of monitoring c_m^i. The increase in the risk factor $\theta = \mu_0[R(1-\epsilon)+w^j]$ provides H the means to increase p_i, which enhances the chance of the hypervisor to detect an attack. As a result, the probability of attacking a vSDN controller gradual reduces, which results in the improved performance of the SG applications. As we now know that $\sum_{j=0}^{N} q_j = 1$, and $q_0 = 1 - \sum_{j=1}^{N} q_j$, thus;

$$q_0 = 1 - \sum_{j=1}^{N} \frac{c_m^i}{\mu_0[R(1-\epsilon) + 2w^j]} \quad (24)$$

The equation 24 represents the probability with which the malicious sender does not attack any guest vSDN controllers.

2) PROBABILITY DISTRIBUTIONS OF A HYPERVISOR OVER ITS ACTIONS

We now present the probability distribution p_i's and p_0 for the hypervisor when it plays the best response strategy. To determine the BNE, we consider the indifference condition for the attacker while determining its strategy.

$$U_D(a_N) = U_D(a_N - 1) =, \ldots, = U_D(a_1) = U_D(a_0) \quad (25)$$

For each p_i, we consider $U_D(a_j) = U_D(a_0)$, such that from equations 2 and 3 we obtain;

$$\mu_0[-p_{i=j}\{c_e^{j=i} + w^{j=i} + R\} + \{w^{j=i} + R - c_a^{i=j}\}] = 0$$

$$p_{i=j} = \frac{w^{j=i} + R - c_a^{i=j}}{c_e^{j=i} + w^{j=i} + R} \quad (26)$$

Equation 26 represents the probability of H to monitor i^{th} vSDN controller k_i, such that the probability increases with the increase in parameters, i.e., the financial worth of the hosted vSDN controller w^i and the overall resources available for distribution. Hence, the more valuable the asset greater is the probability to monitor it. Whereas, $\phi = c_e^{j=i} + w^{j=i} + R$ depicts the risk factor of an attacker, which significantly relies on the penalty an attacker has to pay in case its attack gets detected. This penalty will permanently block the device from sending any requests to the hosted vSDN controllers. Hence, if an attacker constantly sends requests, its risk of losing a bot increases. Consequently, the attacker refrains from attacking i^{th} vSDN controller. At the same time, the hypervisor gradually decreases its probability to monitor i^{th} vSDN controller.

Finally, we determine the probability that H does not monitor any of its hosted vSDN controllers. The fact that $\sum_{i=0}^{N} p_i = 1$ implies $p_0 = 1 - \sum_{i=1}^{N} p_i$, from which we obtain:

$$p_0 = 1 - \sum_{i=1}^{N} \frac{w^{j=i} + R - c_a^{i=j}}{c_e^{j=i} + w^{j=i} + R} \quad (27)$$

In this subsection, we determined both the pure strategy Nash equilibrium and the mixed strategies Nash equilibrium. We calculated the mixed strategy probabilities that both the players play over their actions at the Nash equilibrium. We provided a detailed analysis of the proposed dynamic Bayesian game, which contributes to the optimal distribution of load detection by a hypervisor among its hosted vSDN controllers. In the following section, we analyze the performance of the proposed game model.

VII. PERFORMANCE EVALUATION

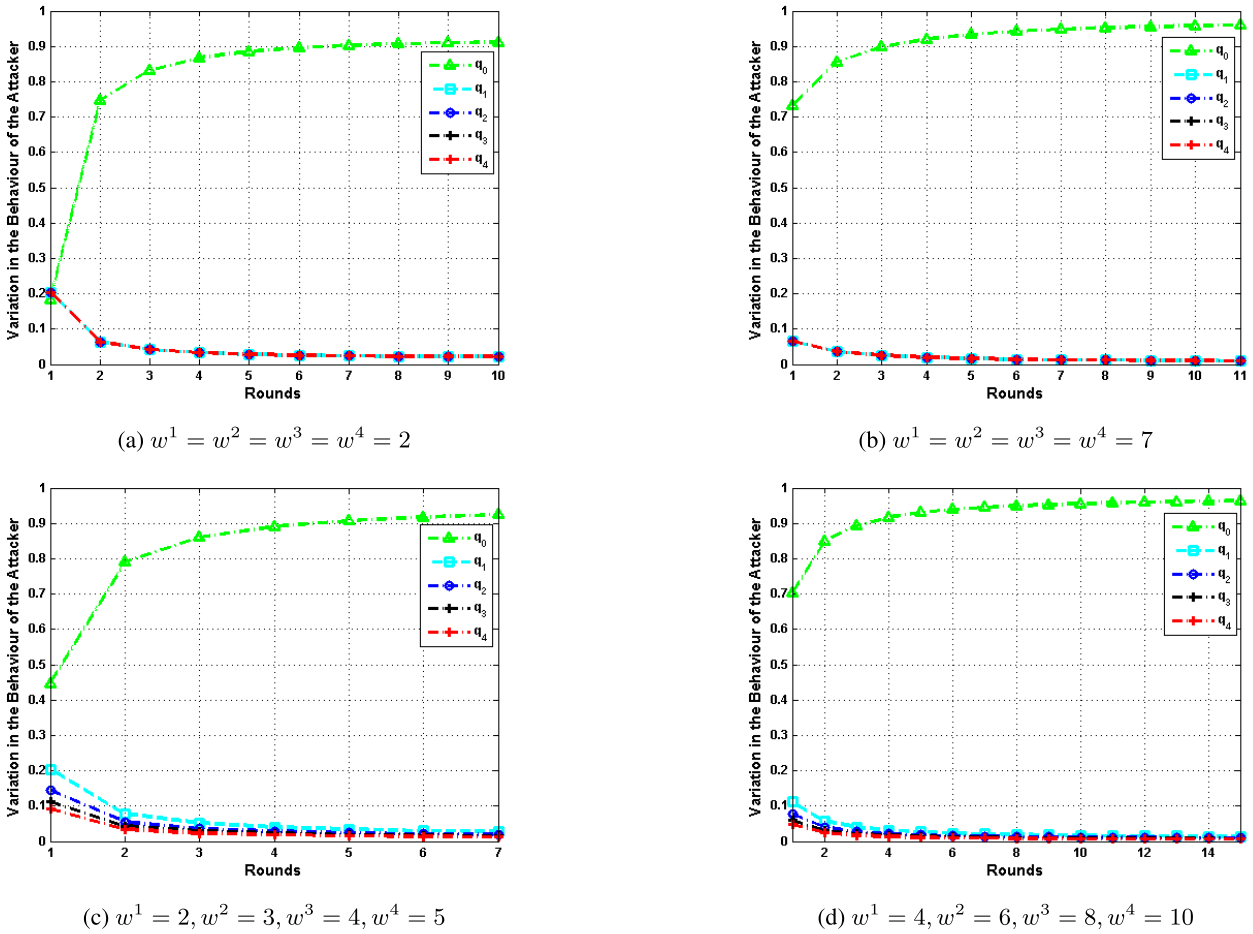
In this section, we validate the performance of our dynamic game model between a Hypervisor and an attack source. The repeated strategic interaction between the two players is studied via the simulations performed in MATLAB. In order to set up the simulation environment, we have considered the number of hosted vSDN controllers to be $N = 4$, where the parameters are assigned default values of $\alpha = 0.1$, $\beta = 0.02$ and $\epsilon = 0.1$. Similarly, the cost of monitoring and attack for all the controllers is initialized with $c_a^j = c_m^i = 0.1$, unless otherwise defined. The purpose of evaluating this model is to observe the impact of parameters over equilibrium strategies of the participating players.

A. IMPACT OF FINANCIAL WORTH

The financial worth of a vSDN controller plays an essential role in determining the probability distribution of an attacker based on the associated attack gain. In the results presented in fig 3, we have fixed the cost of monitoring, and we vary the worth of the vSDN controllers. We obtain plots in fig 3a and 3b while keeping the worth of each vSDN controller fixed at the minimum and maximum values of $w^i = 2$ and $w^i = 7$ respectively.

The plots suggest that the attacker initially distributes its resources equally among all vSDN controllers. However, the behavior converges by stage game 4. The attacker avoids allocating its resources to attack any of the vSDN controllers to avoid the penalty on a successful detection by the hypervisor. Similarly, plot 3b indicates that the belief approaches its maximum value by stage game 3. Moreover, the attack on each vSDN controller is initialized with a lower frequency in comparison to the vSDN controllers of smaller worth.

In plot 3c, we vary the financial worth of each vSDN controller to observe the attacker's distribution of resources, while in 3d, we double their financial worth for a better analysis. From fig 3c, we infer that an attack occurs with a higher intensity, which can be the result of the belief that vSDN controllers with smaller worth shall get monitored with a lower probability. Though, we do notice a gradual decrease in the likelihood to attack vSDN controllers with the updates about belief, which is similar to the results observed in plot 3a.



(a) $w^1 = w^2 = w^3 = w^4 = 2$

(b) $w^1 = w^2 = w^3 = w^4 = 7$

(c) $w^1 = 2, w^2 = 3, w^3 = 4, w^4 = 5$

(d) $w^1 = 4, w^2 = 6, w^3 = 8, w^4 = 10$

FIGURE 3. Impact of the financial worth on the strategy of players.

While the plot in fig 3d resembles the behavior observed in plot 3b, it indicates that the attacker is conscious of attacking vSDN controllers with higher worth.

This behavior of the attacker can be the result of two reasons. Firstly, the attacker is aware of the fact that H will monitor a vSDN controller having more worth with higher probability. The attacker will have to endure the penalty of c_e^j, i.e., $c_e^j > w^i$ such that an attack will not impact the performance of the hosted SG applications. Secondly, the gain received at the first round of the game for vSDN controllers of high worth sufficiently affects the SG applications, which is adequate for the attacker to decrease its distribution for the later rounds. Hence, the substantial attack gain implies a reduction in the number of attacks.

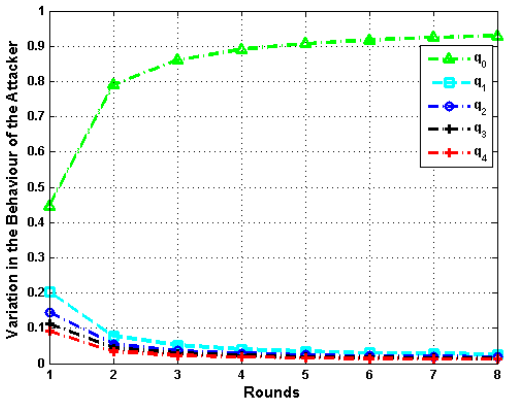
B. IMPACT OF THE MONITORING COST

Fig 4 represents how monitoring cost affects the convergence of the belief of a hypervisor against the strategy of the attacker. In this analysis, we have varied the worth of each vSDN controller, i.e., $w^1 = 2$, $w^2 = 3$, $w^3 = 4$ and $w^4 = 5$, while the cost remains fixed for the plots presented in 4a. From the graphs, we infer that the hypervisor monitors vSDN controllers with higher probability when the monitoring cost is lower. However, when we analyze the obtained results, in comparison to the plots shown in fig 4b, distribution for attack increases. On the other hand, we also notice the belief update results in a lower distribution for the action *not attack* in comparison to the fig 4a. An Attacker does so because the increased monitoring cost restrains the hypervisor for distributing its resources for monitoring.

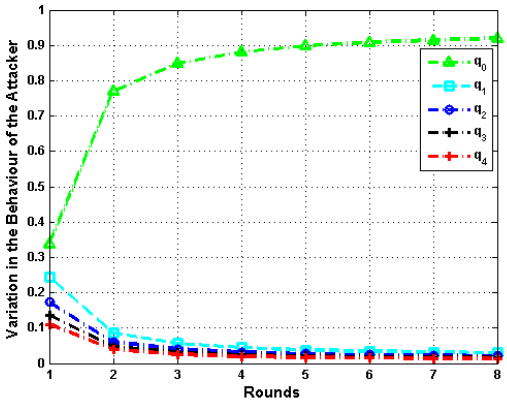
Furthermore, we analyze the behavior of an attacker when the cost to monitor vSDN controllers is different in fig 4c. The distribution for action *not attack* is slightly lower than previously observed results. However, the convergence of the belief of the hypervisor is similar to the fig 4b. Also, the probability distribution to attack reduces at a higher rate, which improves operations of the supported SG applications. The rationale behind this behavior is that the attacker is well aware of the impact of the cost escalation over the hypervisor's monitoring strategy. Nevertheless, the belief of the hypervisor converges by the 5^{th} stage and is updated in the later rounds more frequently.

C. IMPACT OF GAIN/COST RATIO W^I/C_m^I

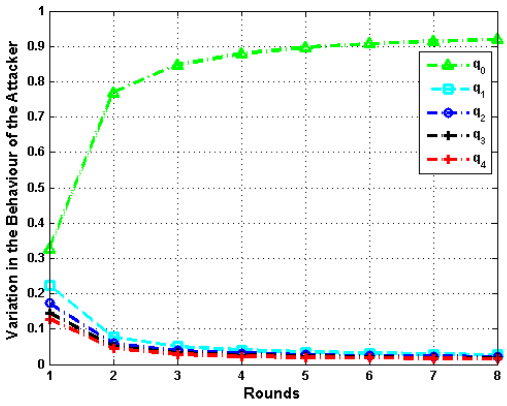
In the fig 5, we analyze how the financial worth to monitoring cost ratio impacts the belief update of a hypervisor w.r.t each vSDN controller. The gain cost ratio represents the difference



(a) $c_m^1 = c_m^2 = c_m^3 = c_m^4 = 0.1$



(b) $c_m^1 = c_m^2 = c_m^3 = c_m^4 = 0.12$



(c) $c_m^1 = 0.11, c_m^2 = 0.12, c_m^3 = 0.13, c_m^4 = 0.14$

FIGURE 4. Impact of monitoring cost on the strategy of players.

between the gain in terms of the financial value of an affected vSDN controller to an attacker in comparison to the monitoring cost the hypervisor needs to pay. This analysis provides a keen approach to analyze the impact of the relationship between the gain and cost on players strategic interaction. We conclude that for a greater value of gain to cost ratio, the belief converges slowly. This is because the intensity of the attacks on the vSDN controllers having more worth will be less in comparison to those which have lower worth. Fig 5 illustrates

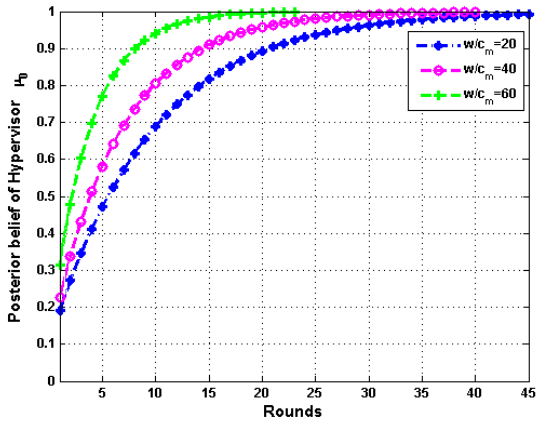


FIGURE 5. Impact of gain cost ratio on the belief system update.

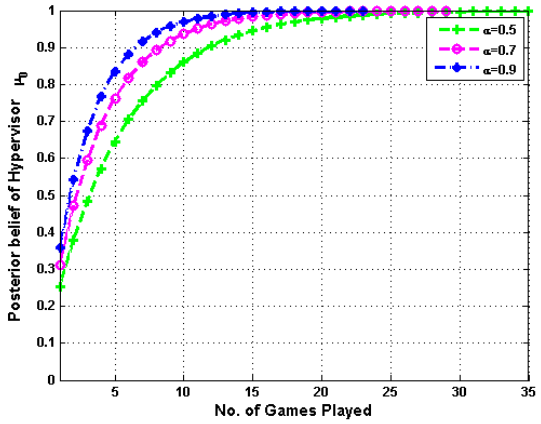


FIGURE 6. Impact of detection rate on the belief system update.

that the smaller the ratio of the hosted vSDN controller to the monitoring cost is the slower the belief converges. On the contrary, high ratio leads to frequent belief convergence as for $w/c_m = 40$, the belief converges by the round 25, while for $w/c_m = 60$, the belief converges by the stage game 15. When the ratio is smaller, the malicious switch needs to send requests more often to be profitable, and vice versa.

D. IMPACT OF THE DETECTION RATE

We determine the impact of varying detection rates on the probability distribution of an attacker, and we fix the false alarm rate, $\beta = 0.09$. As shown in the fig 6, we infer that an attacker will prefer distributing its resources more optimally as the detection rate increases. For the detection rate of $\alpha = 0.5$, the attacker prefers to attack with a higher probability, while the belief of the hypervisor converges at stage 24. On the other hand, as the detection rate increases, we observe a gradual increment in the belief update, i.e., for $\alpha = 0.7$ and $\alpha = 0.9$ the belief converges to 1, at the stage 17 and 14 respectively.

E. IMPACT OF THE FALSE ALARM RATE

Lastly, in this section, we evaluate the impact of the false alarm rate for our model. Fig 7 demonstrates that the increased false alarm rate provides an attacker with the

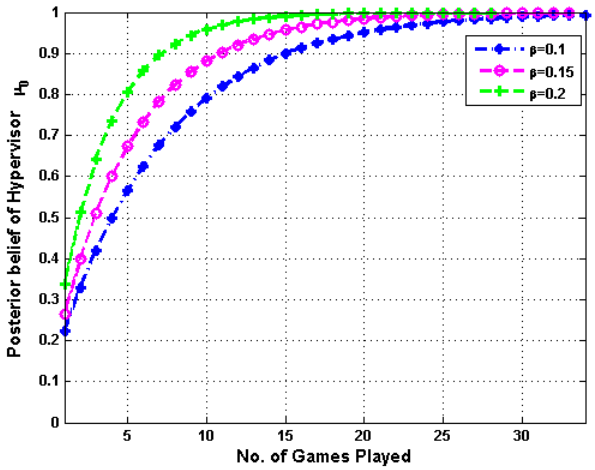


FIGURE 7. Impact of the false alarm rate on the belief system update.

opportunity to take advantage of the situation. Nevertheless, our model presents effective results. For $\beta = 0.2$, the belief of the hypervisor converges to 1 at the game stage 30. While, for $\beta = 0.15$ and $\beta = 0.1$, the belief converges rather quickly at the game stages 21 and 13 respectively.

In this section, we evaluated the performance of the proposed dynamic Bayesian game of intrusion detection against the financial worth of the controllers, the cost of monitoring, detection rates, false alarm rates, and the gain cost ratio. The results show that our model effectively detects compromised switches and minimizes their impact on the operations of the vSDN based SG. It is evident from the results that the restriction of a compromised switch penalty design of the payoff relationship as a Bayesian game mitigates the malicious behavior of a DDoS attacker. At the same time, the efficiency of the system improves due to the optimal distribution of monitoring resources by the hypervisor, by formulating belief towards the type of switch based on its history profile.

VIII. CONCLUSION

To prevent the exploitation of a vSDN-based SG architecture, this work formulated the strategic interaction between a hypervisor monitoring its vSDN controllers and the source of new flow requests sent from switches compromised by a DDoS attacker, as a non-cooperative dynamic Bayesian game of intrusion detection. Our game model enables a hypervisor to distribute its limited resources to monitor guest vSDN controllers optimally. Simulation results showed that our game model enables a hypervisor not only to increase the probability of detecting distributed attacks and minimize false positives, but at the same time, its monitoring costs get reduced as the allocation of resources to monitor vSDN controllers depends upon its belief about the source of the attacks that it forms based on its observation. We presented the best response analysis by determining the dynamic Bayesian game Nash equilibrium which depicts the players' probability distributions over their actions. Furthermore, we evaluated the impact of the following parameters on the strategies of both the players: a) financial worth, b) monitoring cost, c) gain-cost ratio, d) detection rate and d) false alarm rate.

REFERENCES

[1] M. M. Hasan and H. T. Mouftah, "Cloud-centric collaborative security service placement for advanced metering infrastructures," *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 1339–1348, Mar. 2019.

[2] S. Bera, S. Misra, and J. J. P. C. Rodrigues, "Cloud computing applications for smart grid: A survey," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1477–1494, May 2015.

[3] M. Yigit, V. C. Gungor, and S. Baktir, "Cloud computing for smart grid applications," *Comput. Netw.*, vol. 70, pp. 312–329, Sep. 2014.

[4] D. A. Chekired, L. Khoukhi, and H. T. Mouftah, "Decentralized cloud-SDN architecture in smart grid: A dynamic pricing model," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 1220–1231, Mar. 2018.

[5] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, Jan. 2016.

[6] H. Cheng, J. Liu, J. Mao, M. Wang, J. Chen, and J. Bian, "A compatible openflow platform for enabling security enhancement in SDN," *Secur. Commun. Netw.*, vol. 2018, 2018, Art. no. 8392080.

[7] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, Jul. 2013.

[8] M. H. Rehmani, A. Davy, B. Jennings, and C. Assi, "Software defined networks based smart grid communication: A comprehensive survey," 2018, *arXiv:1801.04613*. [Online]. Available: https://arxiv.org/abs/1801.04613

[9] A. Alsirhani, S. Sampalli, and P. Bodorik, "DDoS attack detection system: Utilizing classification algorithms with Apache spark," in *Proc. 9th IFIP Int. Conf. New Technol. Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–7.

[10] P. Dong, X. Du, H. Zhang, and T. Xu, "A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[11] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Software-defined networking security: Pros and cons," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 73–79, Jun. 2015.

[12] M. H. Rehmani, M. E. Kantarci, A. Rachedi, M. Radenkovic, and M. Reisslein, "IEEE access special section editorial smart grids: A hub of interdisciplinary research," *IEEE Access*, vol. 3, pp. 3114–3118, 2015.

[13] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 655–685, 1st Quart., 2016.

[14] X. Jin, J. Rexford, and D. Walker, "Incremental update for a compositional SDN hypervisor," in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 187–192.

[15] J. Lhteenmki, "Scenario based security evaluation: Generic openflow network," Dept. Commun. Netw., Alto Univ., Espoo, Finland, Tech. Rep., 2014.

[16] A. Kamisiński and C. Fung, "Flowmon: Detecting malicious switches in software-defined networks," in *Proc. Workshop Automated Decis. Making Act. Cyber Defense*, 2015, pp. 39–45.

[17] B. Yuan, H. Jin, D. Zou, L. T. Yang, and S. Yu, "A practical byzantine-based approach for faulty switch tolerance in software-defined networks," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 2, pp. 825–839, Jun. 2018.

[18] S. Wang, K. G. Chavez, and S. Kandeepan, "SECO: SDN sEcure COntroller algorithm for detecting and defending denial of service attacks," in *Proc. 5th Int. Conf. Inf. Commun. Technol. (ICoIC7)*, May 2017, pp. 1–6.

[19] H. Zhou, C. Wu, C. Yang, P. Wang, Q. Yang, Z. Lu, and Q. Cheng, "SDN-RDCD: A real-time and reliable method for detecting compromised SDN devices," *IEEE/ACM Trans. Netw.*, vol. 26, no. 5, pp. 2048–2061, Oct. 2018. doi: 10.1109/TNET.2018.2859483.

[20] Z. Fan, Y. Xiao, A. Nayak, and C. Tan, "An improved network security situation assessment approach in software defined networks," *Peer-to-Peer Netw. Appl.*, vol. 12, no. 2, pp. 295–309, Mar. 2019. doi: 10.1007/s12083-017-0604-2.

[21] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "SPHINX: Detecting security attacks in software-defined networks," in *Proc. NDSS*, vol. 15, Feb. 2015, pp. 8–11.

[22] W. Wang, W. He, and J. Su, "Network intrusion detection and prevention middlebox management in SDN," in *Proc. IEEE 34th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Dec. 2015, pp. 1–8.

[23] A. Chowdhary, S. Pisharody, A. Alshamrani, and D. Huang, "Dynamic game based security framework in SDN-enabled cloud networking environments," in *Proc. Int. Workshop Secur. Softw. Defined Netw. Netw. Function Virtualization*, 2017, pp. 53–58.

[24] T. Wang, F. Liu, J. Guo, and H. Xu, "Dynamic SDN controller assignment in data center networks: Stable matching with transfers," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.
[25] H. Chen, G. Cheng, and Z. Wang, "A game-theoretic approach to elastic control in software-defined networking," *China Commun.*, vol. 13, no. 5, pp. 103–109, May 2016.
[26] A. Ksentini, M. Bagaa, T. Taleb, and I. Balasingham, "On using bargaining game for optimal placement of SDN controllers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

RUMAISA AIMEN NIAZI is currently pursuing the M.S. degree in information security with the Department of Computer Science, COMSATS University Islamabad, Islamabad Campus, Pakistan. Her research interests include network security, cloud computing, software defined networks, block chain technologies, and applied game theory.

YASIR FAHEEM received the B.S. degree in computer science from NUCES-FAST, Pakistan, in 2006, the M.S. degree in networks and distributed systems from the Université de Nice-Sophia Antipolis, France, in 2008, and the Ph.D. degree in computer science from the Université Paris Nord, France, in 2012. He is currently an Assistant Professor with the Department of Computer Science, COMSATS University Islamabad, Islamabad Campus, Pakistan. Moreover, he is interested in the applications of the algorithmic game theory to various domains. His current research interests include the Internet of Things, cloud computing, and wireless ad hoc networks (sensor networks, cognitive radio networks, and opportunistic networks). He has served on the technical program committee of the IEEE ICC, the IEEE GLOBECOM, and various other conferences. He is also an Editorial Board Member of the IEEE Internet Policy and the *IEEE Future Directions Newsletters*. He regularly serves as a Reviewer for the *IEEE Communications Magazine* and the *Elsevier Journal of Network and Computer Applications*.