Optimizing resource allocation for secure SDN-based virtual network migration

Fabien Charmet, Gregory Blanc, Christophe Kiennert

Optimizing Resource Allocation for Secure SDN-based Virtual Network Migration

Fabien Charmet, Gregory Blanc, Christophe Kiennert

Optimizing Resource Allocation for Secure SDN-based Virtual Network Migration

Fabien Charmet[1], Gregory Blanc[1], and Christophe Kiennert[1]

[1]Samovar, CNRS UMR 5157, Télécom SudParis, Institut Polytechnique de Paris, Evry, France
[1]{fabien.charmet,gregory.blanc,christophe.kiennert}@telecom-sudparis.eu

***Abstract*—Recent evolutions in cloud infrastructures allowed service providers to tailor new services for demanding customers. Providing these services confronts the infrastructure providers with costs and constraints considerations. In particular, security constraints are a major concern for today's businesses as the leak of personal information would tarnish their reputation. Recent works provide examples on how an attacker may leverage the infrastructure's weaknesses to steal sensitive information from the users. Specifically, an attacker can leverage maintenance processes inside the infrastructure to conduct an attack. In this paper, we consider the migration of a virtual network as the maintenance process. Then we determine the optimal monitoring resources allocation in this context with a Markov Decision Process. This model takes into account the impact of monitoring the infrastructure, the migration process and finally how the attacker may chose particular targets in the infrastructure. We provide a working prototype implemented in Python[1]**

***Index Terms*—Markov Decision Process, Optimization, SDN, Virtual Network Migration, Security, Resource Allocation**

I. Introduction

Cloud infrastructures have become a major component of modern businesses as they are an affordable solution to IT outsourcing. Nowadays, businesses rely on the cloud to support the operations of front-end and back-end solutions for a wide range of activities. This leads these businesses to also store their sensitive data inside the cloud. Over the past years, a number of data breaches has occurred in major companies, making it clear that security should be a concern for both the clients and the company. In this regard, cloud providers have to provide an adequate security level for their customers' information.

Cloud Infrastructures have greatly benefited from a new network paradigm, Software Defined Networking (SDN), that offer a comprehensive view of cloud resources and access network availability [1]. SDN is the decoupling of the data plane and the control plane. Network devices are left with packet forwarding while the network control is done in a centralized entity named "SDN controller". The network control allows the operator of the network to route the traffic in his infrastructure so it can follow specific paths or meet specific requirements in terms of Quality of Service (QoS) or security. The programmability of the network has given developers a lot of flexibility to design new services and network applications and more specifically network virtualization.

Network virtualization focuses on sharing network resources among several users while maintaining a certain level of isolation between each user. The seminal work on this approach using SDN is FlowVisor [2], in which the authors have split their production network to have both experimentation and production traffic on the same physical equipment without incurring any interference between both. Even though there has been numerous studies in the literature about the security of SDN [3], [4], [5], there are some specific aspects of SDN operations that have received little attention security-wise, namely the migration of virtual networks.

The migration is a problem studied as part of the Virtual Network Embedding (VNE) in which the infrastructure provider has to determine which set of physical resources is best suited to support the operations of the virtual network. We further refer to this set of physical resources as the destination substrate. Once the destination substrate has been determined, the virtual resources (i.e., network nodes configuration) will be migrated. If an attacker can impersonate the network hypervisor he will be able to leverage the migration process to setup an environment in which he will be able to exfiltrate sensitive data belonging to the virtual network owner. In order to do so, he will have to deploy specific configuration rules inside several network nodes allowing him to duplicate the sensitive traffic and redirect

[1]https://github.com/FabienCharmet/MDPRA

it towards an "extraction point" in the infrastructure (typically a virtual machine inside the cloud).

In order to protect the infrastructure against such attacks, network monitoring resources must be deployed on network devices to detect attacks on the migration. However, such monitoring incurs a financial cost and a performance impact on the general operation such that it makes it impossible to deploy security measures inside the whole infrastructure. Therefore, determining an optimal distribution of the monitoring resources on the network equipments will provide an adequate security level while meeting the budget's requirements.

In this paper, we propose to model the migration of a virtual network inside the cloud infrastructure as well as the attacker's strategy. To do so, we have designed a Markov Decision Process (MDP) that will let the defender choose iteratively where to deploy the monitoring resources based on the attacker's methodology. Once the MDP is generated, we solve it using the MDPToolbox proposed by Chades et al. [6]. Finally, based on the MDP solutions, we propose an *a priori* deployment of the monitoring so the infrastructure can be optimally protected.

The remainder of this paper is structured as follows: Section II describes the problem statement and Section III the considered attack model. Section IV describes the assumptions made for this problem. Section V introduces the basics of Markovian Decision Processes and Section VI describes our proposition of an MDP to answer the problem statement. We then provide a numerical application of our scenarios in Section VII and discuss some findings and limitations of the model in Section VIII. We present the related work in Section IX. We finally conclude this paper in Section X.

II. Problem statement

We consider an infrastructure provider, referred to as "defender" for the rest of this paper, offering a network virtualization service to the users. Network virtualization, similarly to legacy host virtualization, is prone to physical failures and attacks. To address these problems, the affected virtual network is migrated and new physical resources (destination substrate) are allocated to support the operations of the migrated virtual network. The migration occurs as the configuration rules are sequentially [7] deployed into the destination substrate. The attacker may exploit this process by altering the configuration of specific nodes to exfiltrate sensitive information from the migrated virtual network and route it toward an "extraction" point. The goal of the defender is then to improve the security of his infrastructure against attacks on the migration process. In order to do so, he will deploy monitoring resources that will be used to collect information about events in the infrastructure. However, while virtual machines are essentially clustered inside one physical server, virtual networks are a collection of distributed resources that can not be monitored at a single physical point. Therefore, the monitoring has to be physically distributed as well. Since the defender is limited by the financial cost of such deployment and the performance impact incurred by the monitoring, he aims at finding a cost-effective strategy to select which nodes should be monitored in the infrastructure in order to provide the optimal coverage.

Determining which nodes should handle the monitoring inside the infrastructure requires to model several aspects. The first aspect is the system supporting the virtual network operations and its internal elements. This system is composed of physical nodes, using the Software Defined Networking paradigm to provide an adaptive solution for network virtualization. These nodes are connected together and both links and nodes compose what we will further refer to as "the physical infrastructure". The second aspect is the migration of the virtual network to a new physical substrate, thus having a system evolving independently of the security measures deployed and the potential attacks happening. Regarding the security aspect of the operations, the goal is to determine which nodes the defender should monitor to detect attacks on his infrastructure, but also take into account the strategy of the attacker. The last parameter to account for is the cost of the security deployment since it represents a financial cost for the defender as well as a performance impact for the end user.

III. Attacker model

Fig. 1 depicts the evolution of the infrastructure once the attacker makes part of it unavailable. At first, the virtual network is running on a healthy physical substrate; but once the attack is launched, the substrate becomes unavailable and the virtual network must be migrated quickly to reduce the end user's service interruption. The success of the attacks on the migration relies on two main aspects: the ability to affect the configuration of SDN nodes as well as to retrieve the exfiltrated data. While the latter can be easily solved by owning virtual machines in the infrastructure, the former requires to be able to alter nodes configuration. This has been proven possible in [8], [9], [10]. Precisely, the attacker is able to spoof the identity of the network hypervisor, and thus is able to inject malicious flow rules inside the nodes to create the data exfiltration path. However, he is not able

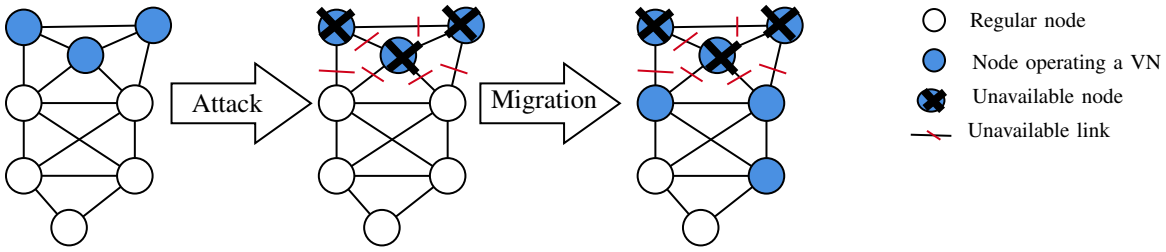

Fig. 1: Migration triggered by the attacker

to be designated as the original network hypervisor in the nodes' configurations. This can be explained because it requires advanced configuration privileges. Moreover, a physical node missing from the legitimate hypervisor's topology view is easily detectable, in comparison to malicious flow rules injected inside the physical nodes. Even though the virtualization infrastructure hosts several virtual networks and end users, we limit the scope of the attacker to a unique target virtual network. He has been able to determine which nodes to attack to trigger the migration thanks to prior scanning and information gathering. Nevertheless, he has no exact knowledge about which nodes will be selected as the destination substrate and he will discover it by doing further scanning and fingerprinting while he is attacking the infrastructure. Even if the attacker may target all nodes in the infrastructure, he has no incentives to attack nodes that will not contribute to exfiltrate data from his victim's network.

We can find a description of such techniques in [11], [12]. This information gathering is not depicted in this paper and we consider that the attacker will choose his targets as presented in Section VI-C. From the point of view of the defender, it is impossible to accurately know which node will be attacked. The attacker may own several virtual machines inside the infrastructure, thus has several sources to launch an attack. However, he can only attack one node at a time. We suppose that each node will always be attacked from the same source. Because of the short time interval considered for the migration, we suppose that the attack will always take the same path. This path will be considered to determine the global detection probability of the attack.

IV. Assumptions

In this section we describe the assumptions we make to address the problem statement.

A. Migration

All nodes in the original substrate have been fully compromised by the attacker and thus are not considered as candidates for the destination substrate. This assumption is reinforced by the fact that forcing all the resources to be reallocated could be leveraged by the attacker in an attempt to have the target virtual network relocated closer to his virtual machines. Virtual machines are already subject to such attacks, as presented by Atya et al. in [13], [14]. The migration will deploy in the destination substrate all the flow rules necessary to operate the virtual network properly. The migration of the virtual nodes is sequential [7], thus all the nodes will be migrated one at a time. We suppose that both virtual network and physical infrastructures are static (i.e., the topology does not change over time).

B. Monitoring

The deployment of the monitoring on the nodes impacts the defender financially and the infrastructure's performance. Based on the work of Ismail et al. [15], we consider the monitoring cost proportional to the intrinsic value of the nodes, (e.g., CPU time on a powerful machine is more expensive compared to a smaller one). Each node on the path of an attack has the same probability to detect it, e.g., there is no node more efficient than another.

C. Targeting nodes

During the migration, the attacker may target nodes to construct the path that will support the exfiltration of the information. We make the assumption that substrate nodes are more likely to be attacked since at least one must be part of the path leading to the exfiltration point. The attacker's strategy for choosing which nodes he attacks is based on the information gathering he performs while attacking. The details of such activity is considered out of the scope of this paper. Similar work on cloud environments for virtual machines colocation has been proposed in [16], [17]. Johnson et al. propose in [18] a real time metric that determines the node that is the most likely to be the next target of an attack. If the attacker was able to establish the full path then we consider that the global attack was successful.

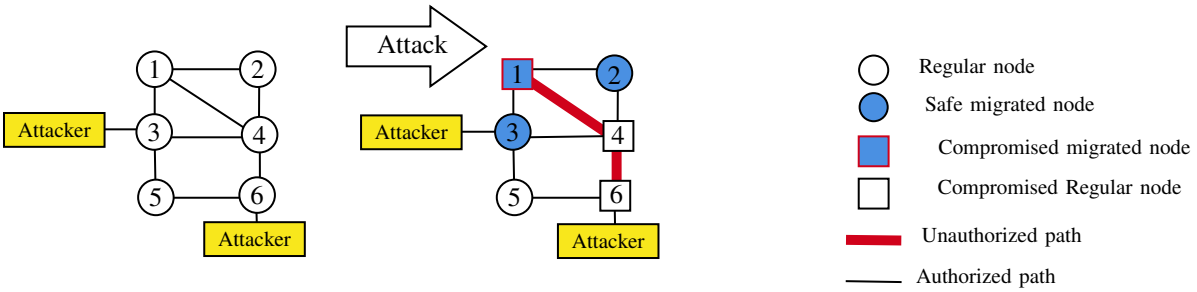

Fig. 2: Exfiltrating information via unauthorized routing

V. Background of Markov Decision Processes

In this section we give a formal definition of a Markov Decision Process (MDP) and present a common algorithm to solve it.

A. Definition

An MDP is a formalism used to represent the evolution of a system based on the decisions made by an agent. These decisions may reward the agent and cause the system to evolve, according to a certain transition probability function. Formally, a Markov Decision Process is defined by a 4-tuple $< S, A, P(s,s',a), R(s,a,s') >$ where S is a finite set of states, A is a finite set of actions, $P(s,s',a)$ is a transition probability function to go from state s to s' when choosing action $a \in A$ and finally $R(s,a,s')$ is the immediate reward gained by choosing a and transitioning from state s to s'. $R(s,a,s')$ is sometimes simplified to $R(s,a)$ in the literature. Actions may reward the agent and can lead the system to evolve from its current state to another. The aim of an MDP is to determine an optimal policy, defining for each state which action will maximize the reward gained by the user while accounting for the consequences of an immediate choice.

B. Solving an MDP

The solution of an MDP consists in determining the action that will maximize the immediate and future rewards for each existing state.

1) The Bellman equation: The Bellman equation [19], shown in Fig. 3 lays the groundwork for solving MDPs using dynamic programming.

This equation states that the optimal utility expected from state s is the sum of the reward obtained when entering state s and the expected discounted sum of probable utilities of the neighbours of s. γ represents the discount factor for the future reward.

The solution of a MDP is the optimal policy $\pi(s)$, which is a vector defining the optimal action for each possible state in the MDP. We propose to determine

$$U(s) = \max_a \left\{ R(s,a) + \gamma \sum_{s'} P(s,a,s')U(s') \right\}$$

Fig. 3: Bellman's equation

the optimal policy using the well known Value Iteration algorithm.

2) Value Iteration Algorithm: This algorithm takes an iterative approach to determine the value of each state in the MDP with the Bellman equation. Value Iteration works as follows:

1) Initialize a value vector $U_0(s) = 0, \ \forall s \in S$
2) Determine the utility of each state iteratively
$$U_{k+1}(s) = \max_a \left\{ R(s,a) + \gamma \sum_{s'} P(s,a,s')U_k(s') \right\}$$
3) Repeat step 2 until $U_k(s)$ converges
4) Compute the optimal policy $\pi(s)$
$$\pi(s) = \arg\max_a \left\{ R(s,a) + \gamma \sum_{s'} P(s,a,s')U(s') \right\}$$

VI. Model

In this section, we describe our MDP model to address the problem of optimal defense resource allocation for virtual network migration.

A. States

The states in the MDP represent the evolution of the system, the progress of the migration, the remaining budgets as well as the compromised nodes in the infrastructure. The defender has two different budgets: b_f, the financial budget for setting and maintaining monitoring, and b_c the global computational power available for the monitoring. Precisely, b_c corresponds to the overall performance impact caused by the monitoring. This represents the amount of resources that can be spent to perform monitoring instead of operational tasks.

We describe the state of the system as the following tuple: $s =< b_f, b_c, Mi, Mo, At >$:

- n: The number of nodes in the infrastructure.
- $\mathbf{N} = \{1, .., n\}$ is the set of nodes in the infrastructure.

- $Mi^s \subset \mathbf{N}$ is the set of currently migrated nodes for state $s \in S$.
- $Mo^s \subset \mathbf{N}$ is the set of currently monitored nodes for state $s \in S$.
- $At^s \subset \mathbf{N}$ is the set of compromised nodes for state $s \in S$.
- b_f^s is the remaining financial budget of state s.
- b_c^s is the remaining computational power of state s.

An absorbing state is a state where all actions transition back to itself.

B. Actions

The defender can either add monitoring on a particular node in the infrastructure, remove this monitoring, or choose to do nothing. The option of doing nothing prevents counter-productive options like forcing undesired actions (e.g., unjustified unmonitoring). We note m_j the action of setting up monitoring on node j. Similarly, we note u_j the action of removing monitoring on node j. Finally we note d the action of doing nothing.
We define $A = \{m_1, .., m_n, u_1, .., u_n, d\}$ as the set of actions available at each state.

C. Probabilistic target determination

We consider the path taken by the attack as a set of nodes. We define L_j the path leading the attacker to node j. We note $\mathbf{Z}$ the set of non compromised nodes that are currently embedding the migrated virtual network (i.e., $\mathbf{Z} = Mi^s \cap \overline{At^s}$). Similarly, we note $\mathbf{T}$ the set of non compromised nodes that are directly connected to a compromised node and are not part of $\mathbf{Z}$, i.e., they are potential candidates for the path between attacker and victim. Alg. 1 is computed at each state as the sets $\mathbf{T}$ and $\mathbf{Z}$ are changing.

Algorithm 1: Probabilistic target determination

Input : $\mathbf{Z}$, $\mathbf{T}$, current state s, n nodes
Output: n probabilities to attack each of n nodes
initialization;
foreach *node* j *in* $\{1, .., n\}$ **do**
  if $j \in \mathbf{Z}$ *and* $Mi^s \cap At^s = \{\emptyset\}$ **then**
    $q(j) = \alpha \frac{3}{3|\mathbf{Z}|+|\mathbf{T}|}$
  else if $j \in \boldsymbol{T}$ **then**
    $q(j) = \alpha \frac{1}{3|\mathbf{Z}|+|\mathbf{T}|}$
  else
    $q(j) = 0$

We note the probability of a specific node being attacked as the combination of the probability for an attack to be launched (i.e., α) and the probability of the node being chosen among all the nodes in $\mathbf{T}$ and $\mathbf{Z}$. We also use a coefficient (here 3) to give more weight to the nodes in $\mathbf{Z}$, referring to the assumption made in IV-C. Once a substrate node has been compromised, the attacker will finalize the exfiltration set-up by completing the path between his VM and the victim's network.

D. Transitions

We describe the coherence of the MDP states with the following transition constraints:

1) If there is no b_f budget left, a state cannot transition to another state (absorbing state)
2) Choosing an action without the required b_c budget will only cost b_f budget with no reward
3) As long as there are nodes to be migrated, each transition will include the migration of one node.
4) A state can only transition to states that preserve the coherence of parameters (budgets, etc.)
5) If there is an action too expensive for b_f it will consume the remaining b_f without any reward
6) Choosing an action twice (i.e., monitoring a node already monitored) only consumes b_f with no reward

Each node $j \in \mathbf{N}$ is characterized by an intrinsic value $V_j \in \mathbb{R}$ that can be seen as the financial value of the node, its computing capacities and its function inside the virtualization infrastructure. Considering that the monitoring cost for each node is not uniform, we note k_f and k_c the atomic monitoring costs respectively associated to budgets b_f and b_c. Then we define the monitoring costs c_f^j and c_c^j for node j as follows:

$$\forall j \in \mathbf{N}, c_f^j = k_f V_j,\ c_c^j = k_c V_j \tag{1}$$

When in state s and after choosing an action $a \in A$, the system can transition to $|\mathbf{Z}| + |\mathbf{T}| + 1$ states, whether an attack happened on one of the nodes or no attack was launched. We define $S' \subset S$ the set of states to which the state s can transition to with a non null probability. We note $s'_{a,j} \in S'$ the state depending on which action a has been chosen and which node will be attacked (index j), and if no attack was launched we note $s'_a \in S'$. To ease the reading we simplify $s'_{a,j}$ and s'_a to s' for the rest of this paper. We define the state modifications once action $a \in A$ has been chosen.

State modifications for action m_i***:*** when choosing action m_i at state s, we compute the budget impact and set changes for state $s', \forall j \in \mathbf{N}$.

$$s \longrightarrow s' = \begin{cases} b_f^{s'} = b_f^s - c_f^i \\ b_c^{s'} = b_c^s - c_c^i \\ Mo^{s'} = Mo^s \cup \{i\} \\ At^{s'} = At^s \cup \{j\} \text{ if there is an attack} \end{cases} \quad (2)$$

State modifications for action u_i*:* when choosing action u_i at state s, we compute the budget impact and set changes for state $s', \forall j \in \mathbf{N}$.

$$s \longrightarrow s' = \begin{cases} b_f^{s'} = b_f^s - c_f^i \\ b_c^{s'} = b_c^s + c_c^i \\ Mo^{s'} = Mo^s \backslash \{i\} \\ At^{s'} = At^s \cup \{j\} \text{ if there is an attack} \end{cases} \quad (3)$$

State modifications for action d*:* when choosing action d at state s, we compute the budget impact and set changes for state $s', \forall j \in \mathbf{N}$.

$$s \longrightarrow s' = \begin{cases} b_f^{s'} = b_f^s - c_d \\ At^{s'} = At^s \cup \{j\} \text{ if there is an attack} \end{cases} \quad (4)$$

We define the transition probability with α and $q(j)$ presented in Section VI-C.

$$P(s, s', a) = \begin{cases} 1 - \alpha \text{ if there is no attack} \\ q(j) \text{ if node } j \text{ is attacked} \end{cases} \quad (5)$$

E. Rewards

The value of the reward for transitioning takes into accounts three criteria: the intrinsic value of the nodes, the overall progress of the attacker, and the probability of detecting an attack. The probability of an attack occurring is already accounted for in the transitions. If no attack was launched, the reward is based on the value of all the nodes in the infrastructure. If an attack was launched, there are two cases: whether the attacker has reached his ultimate goal or not. If he has, we deduct the value of all the compromised nodes. If he has not, we only deduce the value of the attacked node. We note $\Pi(j) = 1 - (1-p)^{|L_j \cap Mo|}$ as the probability of at least one node detecting the attack on node j. Therefore, we define the following reward functions:

$$R(s,a) = \begin{cases} \sum\limits_{i \in \mathbf{N}} V_i, \text{ if no attack} \\ \sum\limits_{i \in \mathbf{N}}^{n} V_i - \sum\limits_{k \in At^s} \overline{\Pi(k)} V_k, \text{ if finalized attack} \\ \sum\limits_{i \in \mathbf{N}} V_i - \overline{\Pi(j)} V_j, \text{ if partial attack} \end{cases} \quad (6)$$

VII. Use Case

In this section, we instantiate our MDP and perform several tests with varying input parameters. We outline specific behaviors shown in the MDP optimal policy. We have generated the MDP using the topology depicted in Fig. 2. We have considered two scenarios: a) the attacker is located at node 6 only, b) the attacker is located at nodes 3 and 6.

We make the simulation computationally tractable by setting all financial and computational costs equal for all nodes, i.e., $\forall i, j \in \mathbf{N},\ c_f^i = c_f^j = c_c^i = c_c^j = c_d = 10$. For eased reading, we note the cost c_a. Simply put, with $c_a = 10$ and $b_c = 40$ there will be a maximum of 4 nodes monitoring the infrastructure. We set the discount factor of the MDP to 0.9 since the consequences of an attack are well-defined and can be precisely evaluated with risk assessment techniques. We summarize the numerical values of the parameters in Table I.

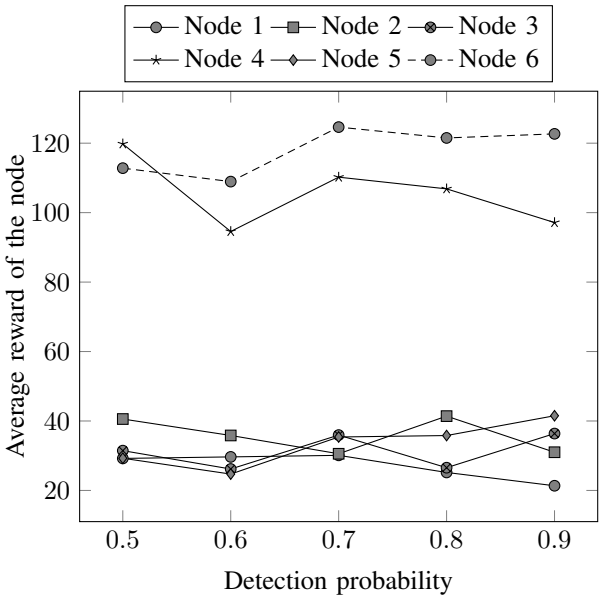

Fig. 4: Nodes impact - Scenario a)

A. Numerical results

We have run the MDP using different budgets and detection probabilities. Nodes are migrated in the following order: 1, 2 and 3. The ordering of the nodes impacts the result sets of Algorithm 1, thus which nodes may be attacked at each transition. We have extracted the monitoring set of each absorbing state, and evaluated the overall reward of each monitoring set. We define the reward of a monitoring set as the weighted mean of the reward of each corresponding absorbing state. The weighted mean uses the stationary distribution of the

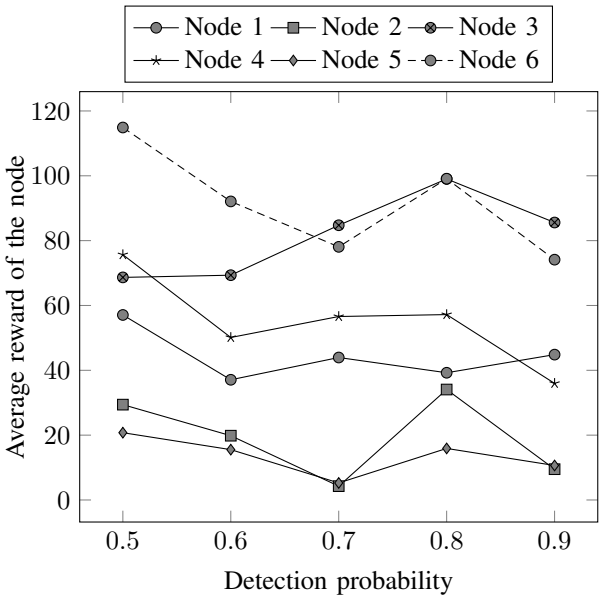

Fig. 5: Nodes impact - Scenario b)

Markov Chain corresponding to the optimal policy. The results are shown in Fig. 4 for scenario a) and in Fig. 5 for scenario b).

Based on the paths taken by attacks and the ordering of the migration, we can categorize the nodes into three categories: source, intermediate and border nodes. The scenario a) sets the source to node 6, nodes 4 and 5 as intermediate nodes and finally nodes 1,2 and 3 as border nodes. The scenario b) sets the source to nodes 3 and 6, nodes 1,4 as intermediate and nodes 2 and 5 as border nodes. The first observation is that node 6 is globally the most rewarding node in both cases. This is explained as it is the source of most of the attacks, and the exfiltrated data is redirected there. In both scenarios the importance of nodes is separated according to the our categorization. This implies that the more the nodes will be on the path of attacks the more they get rewarded. This is observation is reinforced in scenario b) where nodes 4 and 5 are close to attack sources while nodes 1 and 2 are further away.

In both scenarios, the detection rate does not have a significant impact on the ranking of the nodes, compared to each other. The main trend in scenario a) is from $p = 0.6$ node 4 does not overcome node 6 in reward. All other nodes remain closely grouped, and no intermediate or border nodes is standing out. In scenario b) the higher the detection rate the higher the reward of node 3, the secondary source of attacks. The steadiness in the evolution of each node shows that the performance of the detection is not a major factor in determining which

b_f	b_c	p	c_a	γ
[30,40]	[10,20,30,40]	[0.5,0.6,0.7,0.8,0.9]	10	0.9
$\{V_1, V_2, V_3, V_4, V_5, V_6\}$				
[10,10,10,5,5,5]				

TABLE I: Parameters summary

nodes are best suited for the monitoring. We formulate some hypotheses in Section VIII.

Detailed examination of the optimal policy for each budget also shows that the action m_j is never used to redeploy resource elsewhere in the infrastructure. Instead of unmonitoring nodes, the MDP chooses the action d to preserve the global detection probability. m_j actions are only chosen when the unmonitored node does not detect the next attack, thus having the same impact as action d. These corner cases only represent a small percentage of the global solution where very few attacks occurred.

B. A priori deployment

When solving a problem using an MDP, the solution is a dynamic proposition to choose actions as the system evolves. However, from a technical aspect, the defender needs to have the nodes already monitoring the infrastructure before starting the migration process. It becomes necessary to translate the dynamic answer of the MDP into a static *a priori* deployment. After determining the individual importance of each node, we propose to determine the optimal set of monitoring nodes.

The main difference is that each node was evaluated based on all the possible budget combinations, whereas what is defined here is a particular answer for a specific budget. For each budget, the maximum reward is $\frac{b_f}{c_a} \sum_{i \in \mathbf{N}} V_i$ which corresponds to the corner case where the attacker never launched an attack, and we can evaluate the efficiency of the monitoring nodes thanks to the associated reward. Even if a particular monitoring set achieves close to the maximum reward, it is also because the set is tailored to a subset of all possible attacks. We propose to determine the optimal monitoring state for each budget by weighting the reward they achieve with their occupation of the total solution space.

We note S_{abs} the set of absorbing states, $S_{\text{abs}}^{\text{Mo}}$ the set of absorbing states with a common monitoring set Mo, $\rho(Mo)$ the percentage of presence of set Mo in the solution space and $R(Mo)$ the reward of monitoring set Mo. Then we propose to choose the optimal monitoring set Mo^* with:

$$Mo^* = \underset{\text{Mo} \in S_{\text{abs}}}{\arg\max} \left\{ \sum_{s \in S_{\text{abs}}^{\text{Mo}}} \rho(Mo^s) R(Mo^s) \right\} \quad (7)$$

We present the results for scenario a) in Table II. We observe that nodes 4 and 6 are always chosen in the

monitoring, which corresponds to Fig. 4. Node 1 also often appears as a good candidate for a fourth node if it is not already chosen third. With $p = 0.7$ we observe that third and fourth nodes do not coincidate between (30,30) and (40,40) budgets. This suggests that the combining two nodes increases their individual performance. (Node 1 is surrounded by node 2 and 3 in the topology).

detection probability	(b_f, b_c)	Mo^*
0.5	(30,30)	2,4,6
	(40,40)	1,2,4,6
0.7	(30,30)	1,4,6
	(40,40)	2,3,4,6
0.9	(30,30)	4,5,6
	(40,40)	1,4,5,6

TABLE II: Optimal monitoring set - Scenario a)

VIII. Discussion

In this section, we propose to discuss some of the limitations and findings of our approach. The main limitation of the model is the size of the numerical use case. Since the MDP set of states and transitions are generated recursively, the bigger the topology and the budgets the bigger the computation time. Because of the use of the m_j action as an equivalent to action d, we assume that the relocation of monitoring resources would happen in bigger use cases, but those cannot be generated due to combinatory explosion.

In Section VII-B we proposed a method to determine for each budget what was the optimal monitoring set. This method supposes that the ordering of the monitoring deployment does not impact the rewards obtained. While this is not true when considering individually each path from the starting state to an absorbing state, the aggregation of the results toward monitoring sets reduces the impact of this assumption. In addition to that, we have defined an attacker model in which the target decision is not based on which security measures are already deployed on the infrastructure, thus making the target decision space independent from the monitoring.

We have observed the importance of the detection probability in our use case and concluded that it was not an impacting parameter in determining the optimal monitoring set. However, topologies where important nodes could be reached from several paths and where there could be multiple data exfiltration paths could lead to choosing intermediate nodes over border nodes.

IX. Related Work

The migration of a virtual network is a subclass of the Virtual Network Embedding (VNE) problem, which consists in allocating a new physical substrate adequate to support the operations of the virtual network. Migrating a virtual network in an SDN environment consists in deploying flow rules in the destination substrate in order to implement the routing behavior of the original substrate. Migration may occur because the service provider wants to transfer the virtual network toward a substrate with a different QoS, because of a failure of the network devices, or because the original substrate has been compromised by an attacker. In [7], [20] Ghorbani et al. leverages SDN virtualization to offer a seamless migration of virtual elements by cloning and aggregating network configurations across the infrastructure. From the security perspective, [21], [22], [23] determine the appropriate destination substrate to embed the virtual network with regard to the client's required security level.

Linear Programming (LP) is particularly fit to solve Resource Allocation problems since LP determines the optimal way to spend limited resources. LP is a method used to maximize a reward function constrained by a set of linear equations. Network communications have always been subject to resource constraints since there can be an overwhelming number of users on the same physical equipment. The physical constraints of wired/wireless communications may affect the proper behavior of the system as well. In [24], the authors study resource sharing in cellular communications and the underlying noise and interference problems when devices communicate together. However, the resource sharing problem they present is non linear and a division of this complex problem into two simpler ones is proposed and solved using LP. Similarly, in [25], Awad et al. determine the optimal resource sharing in two-hop communications relay networks, where users will communicate cooperatively with a base station and relay stations to alleviate the load on the base station. Their results show that LP can achieve a near optimal resource sharing with a low computation complexity. Early work related to defense is presented in [26], in which the authors use LP to determine the capacities of a set of sensors to track their target. More precisely, the authors determine the maximum detection range each sensor should have in order to ensure the proper tracking of each threat. Providing a safe embedding for virtual networks is studied by Bays et al. [27] as they model the security properties of the resources they are allocating. Similarly

to previous works, LP helps determining the optimal partitioning for network resources while ensuring the required security level is always met. More details on LP, its uses and related algorithms can be found in [28].

While LP works with a system where the optimization is based solely on the point of view of the system owner, there are other scenarios where the solution requires to take into account the perspective of several actors. Specifically, in network security, considering the possible actions an attacker may do to compromise a system will help in providing a better defense against him. In this regard, game theory has proven useful to represent both of these points of view.

The formalism of game theory in computer security could describe the interactions between two (or more) players: attackers and defenders. Players will either attack or defend a particular system, and depending on the choice of their opponent they will be rewarded. Game theory allows to highlight the compromises made by the players to optimize their personal reward while accounting for their opponent's strategies. Games can be classified based on how players choose their actions: either they choose simultaneously (static game) or they take turns (dynamic game). One of the first work on resource allocation using a static game is presented by Kodialam et al. [29] where a model used for drug interdiction is adapted to describe a link sampling optimization. They determine the sampling rate of each link in the infrastructure to optimize the detection of attacks. Another approach with static games is proposed by Chen et al. [30]. For each node in the infrastructure, attacker and defender will be offered the choice of respectively attacking or defending the node or not. Each player has a constrained budget and will have to choose which targets will maximize their reward. Solving the game gives the defender a list of valuable nodes that will be targeted by the the attacker in priority, as well as the optimal defense resources to allocate on each of these nodes. In addition, the authors show that a rational attacker will have no incentive to attack an unprotected node because he will have other protected targets worth the risk of attacking. Stochastic games are a subclass of dynamic games, in which the actions of players have an impact on the state of a system. After both players make their move, the system may transition to another state according to a probability function, and the payoffs for the players may evolve. The dynamic aspect of network configuration using a stochastic game is studied by Zhu et al. [31]. They describe a zero-sum stochastic game where the defender will have to select a set of detection tools that will be deployed inside the infrastructure, each with their own cost of deployment. Meanwhile, the attacker will have to select an attack that can only be detected by a certain tool. If the attack was not detected, the system will transition from a healthy state to a compromised state.

While a stochastic game represents the interactions of two players on the same system, the Markov Decision Process introduced by Bellman [19] is the reduction of this kind of game to one player (agent). Precisely, the agent will be able to choose, at any state the system is in, an action that may lead the system toward another state. Each action may result in a reward or a cost, based on the model. Originally, this approach was used by the industry to determine whether a machine should be repaired or left to manufacture faulty products.

An MDP could be leveraged to tackle core problems in cloud infrastructures, such as energy consumption, resource sharing or service availability, where it could be used to alleviate workloads or organize resources more efficiently. For example, Wang et al. proposed in [32] a Continuous Time Markov Decision Process to optimize the revenue generated by the cloud infrastructure by determining the optimal CPU power depending on client requests and energy consumption. In [33], the authors propose a discrete time MDP where, at each time point, the user will have to choose whether he wishes to keep using the current networking technology or switch to another (e.g., from WLAN to 4G).

X. Conclusion

In this paper, we proposed a Markovian Decision Process in which a Cloud infrastructure provider will deploy monitoring resources on networking nodes in order to protect a network virtualization service from attacks. The attacks are targeting a particular aspect of network virtualization, namely the migration of virtual networks. We also provided an experimental prototype[1] of MDP generation, solving and results analysis. Results show that we can determine which nodes provide the best security with regard to current attacks, as well as how the dynamic aspect of the optimal policy can be translated into an *a priori* deployment of the monitoring resources on the nodes. When the attacker can launch attacks from several sources, the impact of the nodes on the monitoring is much more differentiated and gives a better understanding of their role in the infrastructure. As a future work, we will improve the model by adding uncertainty in the attacker's location thus alleviating the hypothesis that the attacker's location is known beforehand. We will also investigate the correlation

between the different parameters to determine which could be leveraged by the attacker to improve his results. Specifically, the attacker could improve the positioning of his attack sources or which nodes should be prioritized targets.

REFERENCES

[1] S. Azodolmolky, P. Wieder, and R. Yahyapour, "SDN-based cloud computing networking," *International Conference on Transparent Optical Networks*, no. JUNE, pp. 1–4, 2013.

[2] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "FlowVisor: A Network Virtualization Layer," *OpenFlow Switch Consortium, Tech. Rep*, pp. 1–13, 2009.

[3] D. Kreutz, F. M. V. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *arXiv preprint arXiv:1406.0440*, pp. 1–62, 2014.

[4] D. B. Rawat and S. R. Reddy, "Software Defined Networking Architecture, Security and Energy Efficiency: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 1, pp. 325–346, 2017.

[5] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN security: A survey," *SDN4FNS 2013 - 2013 Workshop on Software Defined Networks for Future Networks and Services*, 2013.

[6] I. Chadès, G. Chapron, M. J. Cros, F. Garcia, and R. Sabbadin, "MDPtoolbox: A multi-platform toolbox to solve stochastic dynamic programming problems," *Ecography*, vol. 37, no. 9, pp. 916–920, 2014.

[7] S. Ghorbani, C. Schlesinger, M. Monaco, E. Keller, M. Caesar, J. Rexford, and D. Walker, "Transparent, live migration of a software-defined network," *Proceedings of the 5th ACM Symposium on Cloud Computing, SOCC 2014*, 2014.

[8] J. Hizver, "Taxonomic Modelling of Security Threats in Software Defined Networking," *Blackhat Conference*, 2015.

[9] A. Bokani, M. Hassan, S. Kanhere, and X. Zhu, "Optimizing HTTP-Based Adaptive Streaming in Vehicular Environment Using Markov Decision Process," *IEEE Transactions on Multimedia*, vol. 17, no. 12, pp. 2297–2309, 2015.

[10] B. E. Ujcich, U. Thakore, and W. H. Sanders, "ATTAIN: An Attack Injection Framework for Software-Defined Networking," *Proceedings - 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2017*, pp. 567–578, 2017.

[11] S. Hong, L. Xu, H. Wang, and G. Gu, "Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures," *Proceedings 2015 Network and Distributed System Security Symposium*, no. February, pp. 8–11, 2015.

[12] M. Dhawan, "SPHINX: Detecting Security Attacks in Software-Defined Networks," *Ndss '15*, no. February, pp. 8–11, 2015.

[13] A. Atya, A. Aqil, K. Khalil, S. V. Krishnamurthy, and T. F. L. Porta, "Stalling Live Migrations on the Cloud," no. Llc.

[14] A. Atya, Z. Qian, S. V. Krishnamurthy, T. L. Porta, P. McDaniel, and L. Marvel, "Malicious co-residency on the cloud: Attacks and defense," *Proceedings - IEEE INFOCOM*, 2017.

[15] Z. Ismail, C. Kiennert, J. Leneutre, and L. Chen, "A Game Theoretical Model for Optimal Distribution of Network Security Resources," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10575 LNCS, pp. 234–255, 2017.

[16] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds," in *In Proceedings of the 16th ACM conference on Computer and communications security*, pp. 199–212, 2009.

[17] Y. Zhang, M. Li, K. Bai, M. Yu, and W. Zang, "Incentive compatible moving target defense against VM-colocation attacks in clouds," *IFIP Advances in Information and Communication Technology*, vol. 376 AICT, pp. 388–399, 2012.

[18] J. R. Johnson and E. A. Hogan, "A graph analytic metric for mitigating advanced persistent threat," *IEEE ISI 2013 - 2013 IEEE International Conference on Intelligence and Security Informatics: Big Data, Emergent Threats, and Decision-Making in Security Informatics*, pp. 129–133, 2013.

[19] R. Bellman, "A Markovian Decision Process," *Indiana University Mathematics Journal*, vol. 6, no. 4, pp. 679–684, 1957.

[20] S. Ghorbani and P. B. Godfrey, "COCONUT: Seamless scale-out of network elements," *EuroSys 2017*, pp. 32–47, 2017.

[21] Y. Wang, P. Chau, and F. Chen, "A Framework for Security-Aware Virtual Network Embedding," 2015.

[22] F. Boutigny, S. Betgé-Brezetz, H. Debar, G. Blanc, A. Lavignotte, and I. Popescu, "Multi-provider secure virtual network embedding," *2018 9th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2018 - Proceedings*, vol. 2018-Janua, pp. 1–5, 2018.

[23] L. R. Bays, R. R. Oliveira, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "A heuristic-based algorithm for privacy-oriented virtual network embedding," *IEEE/IFIP NOMS 2014 - IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World*, pp. 1–8, 2014.

[24] A. Moubayed, A. Shami, and H. Lutfiyya, "Wireless Resource Virtualization With Device-to-Device," vol. 61, no. 4, pp. 734–740, 2015.

[25] M. K. Awad and X. Shen, "OFDMA based two-hop cooperative relay network resources allocation," *IEEE International Conference on Communications*, no. ii, pp. 4414–4418, 2008.

[26] J. M. Nash, "Optimal allocation of tracking resources," in *IEEE Conference on Decision and Control including the 16th Symposium on Adaptive Processes and A Special Symposium on Fuzzy Set Theory and Applications*, pp. 1177–1180, 1977.

[27] L. R. Bays, R. R. Oliveira, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Security-aware optimal resource allocation for virtual network embedding," *2012 8Th International Conference on Network and Service Management (Cnsm) and 2012 Workshop on Systems Virtualiztion Management (Svm)*, pp. 378–384, 2012.

[28] J. J. J. &. S. H. D. Bazaraa, M. S., *Linear programming and network flows.* 2011.

[29] Murali Kodialam and T. Lakshman, "Detecting network intrusions via sampling: a game theoretic approach," vol. 00, no. C, pp. 1880–1889, 2004.

[30] L. Chen and J. Leneutre, "A game theoretical framework on intrusion detection in heterogeneous networks," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 2, pp. 165–178, 2009.

[31] Q. Zhu and T. Başar, "Dynamic policy-based IDS configuration," *Proceedings of the IEEE Conference on Decision and Control*, pp. 8600–8605, 2009.

[32] Y. Wang, S. Chen, H. Goudarzi, and M. Pedram, "Resource allocation and consolidation in a multi-core server cluster using a Markov decision process model," *Proceedings - International Symposium on Quality Electronic Design, ISQED*, pp. 635–642, 2013.

[33] K. S. Anupama, S. S. Gowri, B. P. Rao, and T. S. Murali, "An Intelligent Vertical Handoff Decision Algorithm for Heterogeneous Wireless Networks," *Advances in Intelligent Systems and Computing*, vol. 248 VOLUME, no. 2, pp. 331–339, 2014.