Modeling Lateral Attack Containment via Honeypot Architectures in Virtualized Systems

Pierre Charreaux, Yezekael Hayel, Francesco De Pellegrini, Alexandre Reiffers-Masson, Françoise Sailhan

Modeling Lateral Attack Containment via Honeypot Architectures in Virtualized Systems

Pierre Charreaux[1], Yezekael Hayel[2], Francesco De Pellegrini[2], Alexandre Reiffers-Masson[1] and Françoise Sailhan[1]

[1] Lab-STICC Laboratory, IMT Atlantique
[2] Avignon Computer Science Laboratory, University of Avignon

Abstract. Cloud-hosted microservices enable scalable and flexible delivery of online services, but unfortunately, also introduce new attack surfaces. In particular, lateral movement attacks exploit interconnected microservice chains and compromise target service without defenders having time to discover and remediate the breach. In this paper, we explore ways to deploy honeypots as a deception-based defense against lateral movement attacks targeting a microservice oriented architecture. We consider a cost metric corresponding to the expected number of lateral movement steps to perform so that the attack succeed. Using a Markov model, we describe the performance of exploration strategies adopted by an attack to reach its target. The model describes two types of attackers, namely a naive attacker with no knowledge on the system, as well as an experienced attacker with partial information about the distance to travel to reach the target. We evaluate our framework and simulate an attack against a vulnerable microservices chain in a Kubernetes cluster. Evaluations confirm that the honeypot-based defense strategy significantly increases the average time involved to reach the target, providing a baseline for sizing honeypots deployment that effectively slows down attacks.

Keywords: Microservice chains, Honeypots, Cyberdeception, Stochastic processes, Hitting time

1 Introduction

Whilst driving the digital transformation, cloud is also opening up new attack opportunities [10, 12] and security threats for the public and private organizations that rely on it. Cloud-hosted services grant a privileged access to the organization information system and thereby constitute a prominent target characterized by a new attack surface. One of the most critical threats [7] to a cloud environment is lateral movement, an attack which is performed to explore and compromise the sub-systems (i.e., micro-services) of a cloud-hosted service. The lateral movement attack typically starts with a compromised cloud frontend service from which the attacker gradually reaches on-premise back-end service (i.e., the target). In practice, the attacker applies lateral movement(s): the attacker follows the chain

of modular microservices, either deployed in Virtual Machines (VMs) or containers, that form a full-featured service. Defending the cloud against lateral movement attacks remains a challenge to address [2, 14] because attacker takes advantage of the communication paths and trust relationship of the microservice chain [3]. Therefore, attackers move quickly and remain unnoticed.

In this work, we investigate the use of honeypots to lure attackers, divert their efforts, and ultimately isolate them from legitimate services. We study a performance metric that evaluates the efficiency of the deployed honeypots. Traditionally, honeypots serve as a tool to monitor, and support post-attack analysis [8]. However, honeypots can also serve as dynamic shields that protect microservice-based applications by misdirecting and containing adversaries. In our setting, a honeypot is a component (e.g., a microservice, pod, or a container) that corresponds to a decoy emulating a legitimate component, that the attacker cannot distinguish from a legitimate one. A honeypot is accessible from genuine microservices. In practice, the honeypot is configured to ensure that any outgoing traffic is redirected to itself or to another honeypot, effectively trapping the attacker within a deceptive environment. Despite their benefits, the deployment of honeypots at scale, is challenging due to their resource (CPU, storage, memory...) usage and associated operational costs.

Recent approaches [11, 4, 6, 13, 5] optimize the dynamic placement of honeypots to minimize resource consumption. Given a certain number of available honeypots, honeypots are placed so that the expected success chance of the attacker to reach the target is minimized. For instance, in [11] the honeypot placement is formulated as mixed-integer program (MIP), which is further solved via a clustering-based heuristic, suitable for large scale, dynamic graphs. In [4], the dynamic placement of honeypots is modeled as a partially observable stochastic game. Attacker sequentially chooses which hosts to attack and progresses towards the ultimate target, while the defender selects the hosts that act as honeypots and detect specific attacker's moves. The authors of [6] model the lateral movement attack as a time-expanded network with randomly appearing service links. They propose an optimal strategy to place honeypots dynamically on idle production servers, so as to proactively mitigate long-term risks. In [13], the authors model the attacker-defender interactions as a continuous-time Markov decision process where the defender detects and ejects the attacker, at any time, and finds an equilibrium between risks and information obtained from the attacker. In [5], the authors use a semi-Markov decision process to model the attacker's movements and the defender's actions. The defender changes the honeypots' interactivity configuration to optimize the attacker's sojourn time in a honeypot. Using reinforcement learning, they find policies to optimize information gains while reducing risks. In contrast to the above efforts, our model is intended to both (i) evaluate the number of honeypots necessary to slow down the attacker, and (ii) to place them, assuming that the attacker cannot distinguish honeypots from genuine services. Based on our model, a cloud administrator may carefully estimate and place the right number of honeypots to slow down the attacker so as to gain sufficient time to develop countermeasures, or even completely discour-

age the attacker. Depending on the number of honeypots, a cloud administrator estimates the associated overhead[3] (e.g., financial cost), which depends on e.g., the cloud system, its virtualisation/hardware components.

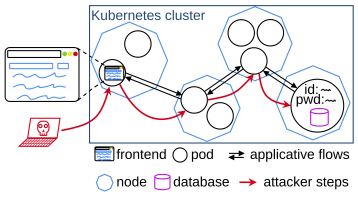
Our model assumes that the attacker cannot distinguish a genuine microservice from a honeypot and therefore chooses the next move at random: the next microservice to attack is selected, be it a genuine microservice or a honeypot. We assume that the attacker, which believes he/she is trapped in a honeypot, has no option but to start from the entry point (i.e., the frontend microservice). Recall that (i) the attacker does not distinguish honeypots from genuine microservices, and (ii) there is also no backtracking to a genuine microservice from a honeypot. Our model takes into account two types of attackers: a naive attacker (also called *myopic* attacker) that has no information about the cloud/application, and an *experienced* attacker that is capable of estimating the number of microservices between the frontend microservice and the target. Thus, when too many lateral movements have been performed without finding the target, the experienced attacker returns to the frontend microservice to perform another attempt. In contrast, a myopic attacker has no information about the system and randomly chooses when to return to the frontend microservice.

Overall, we provide two main contributions. First, we introduce a Markov model of a lateral movement attack performed on a microservices chain equipped with honeypots. The model accounts for generic attackers who do not distinguish honeypots from genuine microservices (Sec. 2). Specifically, we introduce two models that deal with myopic and experienced attackers. Second, we find a mathematical closed form for the mean number of lateral movements necessary to reach the target (Sec. 3). We validate our model numerically w.r.t. the number of attack steps to reach the target and describe the (optimal) behaviour of the myopic and experienced intruders (Sec. 4). Finally, the attack is studied under a real-world setting with a lateral movement attack performed on a microservice chain in a Kubernetes cluster. In addition, our real-world evaluation (Sec. 4.3) confirms that, even with a lightweight deployment, with i.e., one honeypot per microservice, the time for an attacker to reach the target becomes significant, namely at least 20 times longer than without the proposed deception strategy.

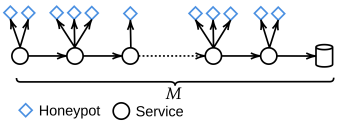
2 System Model

We consider a cloud environment (Fig. 1a) with a microservices chain in which the microservice at the frontend serves as an entry point for any user (including an attacker). The attacker aims at accessing the database at the backend of the chain. The system is modeled (Fig. 1b) as a path composed of M nodes. Each node represents a genuine microservice, the first node is the frontend microservice, and the M-th node is the target, $M \geq 2$. Overall, the attacker performs $M-1$ moves along the microservice chain to reach the target. For each genuine node i (with $1 \leq i \leq M-1$), the defender creates a set containing $L_i - 1$ honeypots, which appear to the attacker as genuine neighbors of node i. Honeypots

[3] Determining an overhead is out of scope of this paper



(a) Lateral movement attack on Kubernetes cluster: starting by attacking/poisonning the entry point, the attacker moves laterally exploiting the vulnerabilities on the encountered microservices.



(b) System is represented as a graph composed of genuine microservices and honeypots.

Fig. 1: An application, composed of a microservices chain, is hosted in a Kubernetes cluster along with honeypots.

are connected to other honeypots. Thus, at step k (with $1 \leq k \leq M-1$) the attacker is offered L_k possible microservices to attack, which may refer to genuine microservice iff node k is a genuine node.

2.1 Lateral movement policy and objective function

At each step k of a lateral movement attack, the next microservice to attack is chosen uniformly at random among the L_k available microservices. We assume that an attacker is aware of the potential presence of honeypots and is willing to avoid getting trapped indefinitely. An attacker may henceforth stop the current attempt and may start a new one from the entry point. We call each of these attempts an *attack*.

The global cost perceived by the attacker depends on the Lateral Movement Policy (LMP), denoted by π. At each step of the lateral movement, the attacker continues the attack with probability π or starts a new attack with probability $1-\pi$. The probability depends on the number of explored nodes. Note that the number of honeypots may be greater than the number of genuine microservices M, in which case the current attack is trapped in a honeypot. We denote q_i the probability of reaching a honeypot once performing a lateral movement from a genuine node of index i; this probability depends on the placement of honeypots, i.e., L_i, in the microservice architecture. The total number of steps $\tau_M(\pi)$ necessary for the attacker to hit the target starting from the initial entry point and using the LMP π verifies:

$$\tau_M(\pi) = (M-1) + \sum_{k=1}^{N(\pi)-1} \tau_\pi(k), \tag{1}$$

with $N(\pi)$ the number of attacks and $\tau_\pi(k)$ the length of the k-th attack, i.e., the number of lateral movements they performed. The total cost $J(\pi)$ for the

attacker following the LMP π satisfies:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{k=1}^{\tau_M(\pi)} \gamma^t \right] = \mathbb{E}_\pi \left[\frac{1 - \gamma^{\tau_M(\pi)}}{1 - \gamma} \right]. \quad (2)$$

The discount γ (with $\gamma \in [0, 1]$) mimics the persistence of the attacker: the smaller γ, the more likely an attacker prolongs the current attack before starting another attack. With $\gamma = 1$, the total cost corresponds to the expected number of steps to reach the target. As shown in the following, for any LMP π, attacks are independent, and thus the lengths of attacks are i.i.d and independent of the number $N(\pi)$ of attacks.

2.2 Markov model

We introduce a Markovian model to study the cost associated with a LMP performed by an attacker. The proofs of the theorems and propositions are provided in the Appendix, except for Theorem 2 for which a sketch of proof is provided because a formal proof requires using hitting time theory. Each attack is performed in discrete steps at $t = 0, 1, \ldots$, where the initial step ($t = 0$) is associated with the exploitation of the vulnerability at the entry point. At any step t, the system state is described by $S(t) = (X(t), Y(t))$ where $X(t) \in \{1, \ldots, M\}$ is the number of explored genuine microservices and $Y(t) \in \{0, \ldots, \infty\}$ is the number of explored honeypots. We denote q_k the probability that an attack gets trapped in a honeypot while performing a lateral movement from the k-th genuine microservice. The transition probabilities $\mathbb{P}(S(t+1)|S(t) = P((k', h'), (k, h))$ write

$$\begin{aligned}
&\mathbb{P}((k+1, 0)|(k, 0)) = (1 - q_k)\pi(k), k = 1, \ldots, M-1,\\
&\mathbb{P}((k, 1)|(k, 0)) = q_k\pi(k), k = 1, \ldots, M-1,\\
&\mathbb{P}((k, h+1)|(k, h)) = \pi(k+h), \forall h \geq 0,\\
&\mathbb{P}((1, 0)|(k, h)) = 1 - \pi(k+h), \forall k \neq M,\\
&\mathbb{P}((M, 0)|(M, 0)) = 1.
\end{aligned}$$

The attack starts at the entry node, which means that $\mathbb{P}(S(0) = (1, 0)) = 1$. An attacker that reaches the target, stays there indefinitely with a zero transition cost. Overall, the total number of steps τ_M involved to hit the target satisfies:

$$\tau_M = \min\{t \in \mathbb{N} | X(t) = M\}.$$

The total number of steps depends on the number of attacks and their respective lengths . Let $\tau_b(k)$ be the length of the k-th time an attack starts, i.e., the attacker backs off to state $(1, 0)$. It verifies:

$$\tau_b(0) = 0 \text{ and } \tau_b(k) = \min\{t \in \mathbb{N} | t > \tau_b(k-1), S(t) = (1, 0)\},\ k > 0.$$

Note that τ_b are the hitting times for state $S_0 = (1, 0)$. In order to avoid trivialities, we assume in the rest of the paper that the attacker is rational, i.e., the

attacker is not trapped indefinitely in honeypot and therefore backs off with positive probability: formally, the LMPs are such that $\sum_{k=1}^{\infty}(1-\pi(k)) = +\infty$, i.e., the state S_0 is positive recurrent once we remove the absorbing state $(M, 0)$ from the kernel of the original MDP. The length of an attack is $\tau(k) = \tau_b(k) - \tau_b(k-1)$. The number of attacks launched from the entry point is defined by:

$$N = \max\{k \in \mathbb{N}_+, \tau_M > \tau_b(k)\} + 1.$$

The next theorem provides the probability distribution of the number of attacks N.

Theorem 1. *Let* $\rho = \prod_{k=1}^{M-1}(1-q_k)\pi(k)$. *Then:*
i. If $\rho = 0$, *then* $N = \infty$ *w.p.1., i.e., the attacker never reaches the target.*
ii. If $\rho > 0$, *then the number of attacks* N *that are necessary to reach the target follows a geometric distribution* $\mathbb{P}(N = h) = \rho(1-\rho)^{h-1}, \quad h = 1, 2, \ldots$

Theorem 2. *i. The attack lengths* $\tau(1), \ldots \tau(N-1)$ *are independent.*
ii. The distribution of the length of failed attacks $\tau(h), h = 1, \ldots, N-1$ *writes*

$$\mathbb{P}\left(\tau(1) = k\right) = (1-\pi(k)) \prod_{j=1}^{k-1} \pi(j),$$

The above theorem gives the distribution probability of the length of failed attacks. Incidentally, such length depends only on the impatience of the attacker. Finally, the following proposition gives a closed-form expression of the total cost for any LMP π.

Proposition 1. *Let* $\gamma < 1$, *and* ρ *be as in Theorem 1, then*

$$J(\pi) = \frac{\gamma}{1-\gamma} \frac{1 - G_{\tau(1)}(\gamma)(1-\rho(1-\gamma^{M-1}))}{G_{\tau(1)}(\gamma)\big(1-(1-\rho)G_{\tau(1)}(\gamma)\big)}$$

where $G_{\tau(1)}(\gamma) = \sum_{k=1}^{+\infty} \gamma^k \mathbb{P}(\tau_1(h) = k)$.

While the attacker is interested in finding the optimal attack policy π^*, which is the one that minimizes its total cost, i.e., $\pi^* := \arg\min_\pi J(\pi)$, the attacker has a partial knowledge, which implies that the attacker cannot solve for the optimal policy. In the next section, we investigate the best attack strategies for a myopic attacker and a deterministic attacker.

3 Specific LMP

We model the Lateral Movement Policy associated with a myopic attacker and a deterministic one. The myopic attacker randomly decides at each step to continue the attack or to go back to the initial node (and launch another attack); the

deterministic attacker takes into account the number of nodes already visited to decide whether to continue the attack.
The deployment of honeypots is assumed uniform, i.e., the defender deploys a constant number $L-1$ of honeypot nodes attached to each genuine microservice on the microservice chain. Hence the probability of falling into a honeypot from a genuine microservice at the k-the step of an attack is $q_k = 1 - L^{-1}$.

3.1 Myopic LMP

During each attack, we assume that the myopic attacker takes at least a first step to the next microservice or honeypot (i.e., $\pi(1) = 1$). Afterwards, the attacker performs a lateral movement with positive, constant probability $\pi(i) = 1 - p$, and start a new attack with constant probability $p = 1 - \pi(i)$. Thus, according to Theorem 1, the probability ρ to reach the target during an attack satisfies $\rho = \frac{(1-p)^{M-2}}{L^{M-1}}$. For $\gamma = 1$, the total cost function is the expected number of steps to reach the target:

$$J(p) = M - 1 + \mathbb{E}\left[\sum_{k=1}^{N-1} \tau(k)\right] = M - 1 + \mathbb{E}[\tau(1)]\mathbb{E}[N-1],$$

where we applied Wald's equation [15]. From theorem 2 it follows that $\mathbb{E}[N] = \frac{1-\rho}{\rho}$. We can write explicitly $\mathbb{P}(\tau(1) = h) = p(1-p)^{h-2}$, $h = 2, 3, ...$, from which it holds that:

$$J(p) = (M-1) + \frac{1+p}{p}\left(\frac{1}{\frac{(1-p)^{M-2}}{L^{M-1}}} - 1\right) \tag{3}$$

Due to the lack of space, we do not provide derivations for $\gamma < 1$ that are similar to determine. Instead, we investigate in the following the worst case for a myopic attack.

Proposition 2. *i. There exists a unique optimal myopic attack policy with parameter p^*.*
ii. $p^* = \Theta(\frac{1}{M-1})$ for large L.

The previous proposition implies that, by increasing the number of honeypots for a given value of M, the optimal policy depends only on the number M of genuine microservices. In particular, in order to avoid being restrained in honeypots indefinitely, the optimal myopic attacker makes paths of average length M. Typically, an attacker ignores the value of M but may have knowledge of its distribution. This kind of Bayesian approach leads to the study of the deterministic threshold policies derived in the next section.

3.2 Threshold policies

We consider an attack behavior in which the decision to continue the exploration of the microservices chain is deterministic. In practice, being unable to

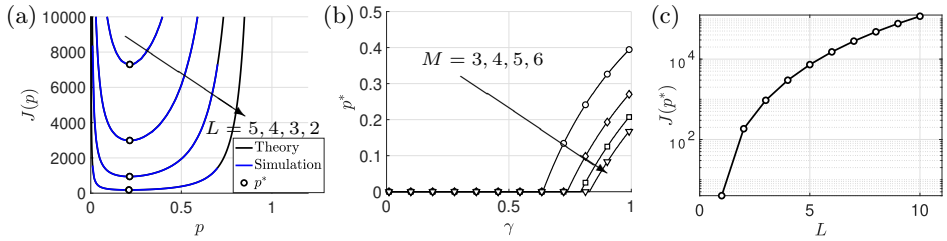


Fig. 2: (a) Equation (1) (black) and simulation curve (blue) for myopic attacker, for $M = 5$ and $L = 5, 4, 3, 2$; the round marker indicates the optimal myopic policy p^*. (b) Impact of the discount factor γ on the optimal myopic policy, for $M = 3, 4, 5, 6$ and $L = 3$. (c) Minimum number of steps to reach target for a myopic attacker as a function of L.

distinguish honeypots from genuine microservice nodes, the attacker explores a microservices chain of fixed length σ before going back to the initial entry point, i.e., the attacker goes back to the entry point whenever $X + Y = \sigma$. Formally, a threshold policy π_σ, for a given $\sigma > 0$, is a LMP defined as

$$\pi_\sigma(k) = \begin{cases} 1 & \text{if} \quad k = 1, \dots, \sigma - 1\,, \\ 0 & \text{if} \quad k \geq \sigma. \end{cases} \tag{4}$$

Note that if the attacker knows the exact number M of genuine nodes composing the microservice chain, then the optimal policy is the deterministic threshold policy π_M that consists of pursuing an attack of length $\sigma = M$. In general, we assume that the attacker may have just a conjectural distribution on the number of nodes M which compose the microservice chain, namely β. Hence, we determine the expected cost corresponding to a certain threshold policy π_σ. The attacker's cost function is $J(\pi_\sigma) = \mathbb{E}_{\pi_\sigma}\left[\frac{1-\gamma^{\tau_M(\pi_\sigma)}}{1-\gamma}\right]$, where $\tau_M(\pi_\sigma)$ is the total number of steps to hit the target under LMP π_σ.

Let us fix $M = m$ and define $J(\pi_\sigma, m)$ the cost under π_σ. We distinguish two cases:
$\sigma < m$: The attacker never hits the target, i.e., $\tau_M(\pi_\sigma) = +\infty$ w.p.1., so that $J(\pi_\sigma, m) = \frac{1}{1-\gamma}$.
$\sigma \geq m$: The attacker hits the target after a finite number of attacks $N(\pi_\sigma)$ w.p.1., where from theorem 1, $N(\pi_\sigma)$ follows a geometric distribution with parameter $\rho = 1/L^{m-1}$.

$$J(\pi_\sigma, m) = \rho \sum_{j=0}^{+\infty} (1-\rho)^j \left(\frac{1-\gamma^{j\sigma+m-1}}{1-\gamma}\right) = \frac{1}{1-\gamma}\left(1 - \frac{\rho\gamma^{m-1}}{1-(1-\rho)\gamma^\sigma}\right) \tag{5}$$

Finally, given a conjectural distribution β for the attacker about the number of genuine microservices in the architecture, the attacker will choose the LMP π_σ

that minimizes the following expected cost:

$$J(\pi_\sigma) = \sum_{m=1}^{\infty} J(\pi_\sigma, m)\beta(m) = \frac{1}{1-\gamma}\left(1 - \sum_{m=1}^{\sigma} \frac{L^{-(m-1)}\gamma^{m-1}}{1-(1-L^{-(m-1)})\gamma^{\sigma}}\beta(m)\right).$$

Note that the latter function depends on σ, so that the optimal threshold for an apriori distribution β can be computed, as shown later in the numerical section.

4 Numerical illustration and empirical experimentation

We perform numerical evaluations for the case of an attacker adopting either the myopic policy (Sec. 4.1) or the deterministic threshold policy (Sec. 4.2). We complement evaluation (Sec. 4.2) with a set of real experiments performed on a cloud emulation platform. We consider a microservice architecture where users access the web page of a target service. The server hosting the web page is forwarding the users' requests to a gateway, which is in charge of routing such requests to the actual service. The service is composed of several microservices, chained together and executed in cascade according to the given chain order. The last microservice extracts data available in a database and forwarding it to the previous microservices. We detail the setup of our microservice architecture and of the lateral movement attack.

4.1 Numerical illustrations of a myopic policy attack

Expected time to reach the target: In Fig. 2a, we consider the average number of movements of an attacker to hit the target for different probabilities of restarting an attack ($p = 1 - \pi$). We consider a number of microservices $M = 5$ and a different number of honeypots per node. The theoretical curve, given by equation (1), in black, matches the simulation results, in blue. As expected from Prop. 2, the optimal probability p^* for a naive attacker to reach the target depends on M and it is upper bounded by $\frac{1}{M-1}$.

Attacker persistence: We can consider a discount factor on the myopic intruders to characterize their persistence when performing an attack. In our model, it indicates whether attackers take into account the future when planning their steps. An impatient attacker takes into account the whole future and is wary of traps: this corresponds to a large value of γ. Conversely, a small value of γ corresponds to a persistent attacker.

In Figure 2b, we plot the optimal probability p^* to start back at front-end for $L = 3$ and different values of M. We observe the presence of a critical threshold value γ^c for which $p^*(\gamma) = 0$ for all $\gamma < \gamma^c$. In particular, $\gamma^c = 0.64, 0.74, 0.8$ and 0.83 for $M = 3, 4, 5, 6$, respectively. For attackers with higher discount factors (e.g., for $\gamma \geq 0.83$ when $M = 6$), the optimal probability to restart the current attack grows mildly super-linearly.

Comparing Fig. 2a and Fig. 2b, the worst attackers from the point of view of an administrator are impatient. Their probability is set to reduce the number of

steps as γ tends to one. Actually, for low values of γ, an attacker may not even return to the front-end service, as visible for $\gamma \leq 0.5$.

Impact of the number of honeypots: to illustrate the impact of the number of honeypots in slowing down attackers, we reported in Fig. 2c the value of $J(p^*)$ for $M = 5$ and increasing L. From (3), it can be seen that the mean number of steps increases polynomially with L. We see that an optimal myopic attack would require around 200 steps on average when deploying one honeypot per genuine microservice ($L = 2$), and around 10^5 steps when we deploy 9. With this graph, we obtain an indication on how many honeypots are needed to shield the microservice chain. Indeed, if the attack needs to be slowed down to at least 10^3 steps on average, then 2 honeypots per microservices are enough. However, if the attacker needs to be slowed for no less than 5000 steps on average, we would need 4 honeypots per microservices. Overall, as the number of honeypots is $(M-1)\times(L-1)$, 4 decoy honeypots per microservice would require to deploy 16 honeypots along a microservice chain with 5 nodes.

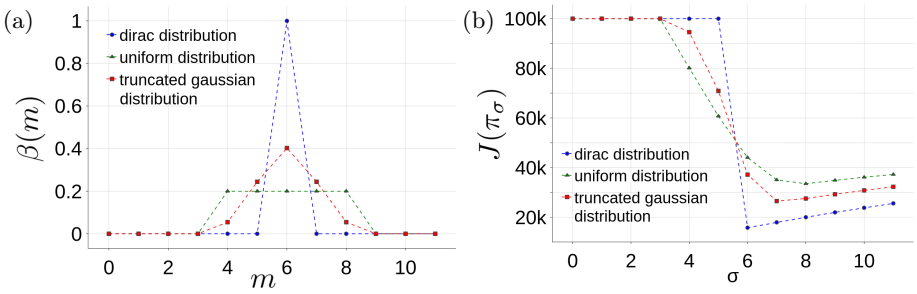


Fig. 3: Threshold attack policies for various β distributions. (a) Attacker's knowledge of the length of the microservices chain $\beta(m)$. In blue (with circles), perfect knowledge, in green (with triangles) uniform possibilities, and in red (with squares) gaussian possibilities, all with mean $M = 6$. (b) Attackers chooses the optimal σ as the one that returns the least average amount of steps, for a given distribution. In blue (with circles), perfect knowledge, in green (with triangles) uniform possibilities, and in red (with squares) gaussian possibilities, all with mean $M = 6$.

4.2 Numerical illustrations of a threshold policy attack

We consider three distributions representing the possible knowledge on M of an experienced attacker (Fig. 3). In Fig. 3b, we plot the cost $J(\pi_\sigma)$ that a persistent attacker ($\gamma \to 1$) undergoes by choosing threshold policy σ. As γ is close to 1, the cost is as large as 10^5 (actually, $\gamma = 1$ would mean infinite average cost). When the attacker knows the exact microservice chain length M, their optimal strategy is $\sigma^* = M$ (Fig. 3b, blue curve with circle markers). It is interesting to notice that when instead the attacker hesitates uniformly between

Table 1: Efficiency of myopic and threshold policy with honeypots compared to baseline policy without honeypots for $M = 6$.

Policy	L	**# Experiments**	**Avg. time to success**	$\hat{\tau_M}$	**Avg. attack duration**	$\hat{\tau}$	$\hat{N}$
Myopic: $p^* = 0.175$	3	9	16851 s	935	116.5 s	6.9 (var = 24)	142.2 (var = 5182)
Threshold: $\sigma^* = 6$	3	10	13023 s	690	92.4 s	5 (var = 0)	138 (var = 8225)
Myopic: $p^* = 0.173$	2	10	3309 s	221	92 s	6.3 (var = 21)	35.2 (var = 1181)
Threshold: $\sigma^* = 6$	2	10	1644 s	75	107.7 s	5 (var = 0)	15 (var = 192)
Baseline	1	44	70.9 s	5	70.9 s	5 (var = 0)	1 (var = 0)

several possible values of M (Fig. 3b, green curve with triangle markers), their optimal strategy σ^* is to explore up to the highest possible value, i.e., $\sigma^* = 8$. However, in the truncated gaussian case (Fig. 3b, red curve with square markers), where higher values correspond to a smaller tail of the distribution, the optimal threshold σ^* is smaller than the highest possible value. In this case, in fact, the event that $\{M = 8\}$ has a very low probability, and the optimal choice for the attacker is $\sigma^* = 7$. We observe that a more precise apriori knowledge on M – a skewed distribution mimics a more precise apriori knowledge than a uniform one – leads also to a reduction of the minimum expected cost for the attacker.

4.3 Empirical experimentation in a virtualized environment

Using kubeadm, we setup and manage a Kubernetes cluster made of 3 master nodes that control the cluster and 2 workers that run the service. We use the Calico framework to configure secure networking and enforce advanced policies for cloud-native microservices/applications. Using this framework, we deploy vulnerable microservices, and then we attack the cluster through a lateral movement strategy. We setup 5 genuine microservices, each with two pods, and a single backend microservice, resulting in a total of $M = 6$ microservices. Each microservice corresponds to a Python web server with a vulnerability: the attacker may execute any user-submitted data as a system command. As illustration, the frontend microservice allows the execution of arbitrary commands submitted through a webpage while the backend microservice reachable from the previous vulnerable microservice, returns a success flag. Additionally, we deploy honeypots mimicking a genuine microservices but with no possibility to reach another genuine microservice: this is prevented by the calico network rules. Similarly, Calico network rules enforce lateral movement between consecutive microservices.

The attacker relies on the Netcat tool, which creates TCP connections between machines and enables remote system commands execution. In practice, the

attacker first gets a reverse shell on the front-end microservice with a terminal capable of executing remote commands. Then, the attacker listens for connections to the front-end machine, and uses the vulnerability of the next microservice to obtain another reverse shell. Repeating these steps, the attacker reaches the backend service. To automate the attack, a Python script sends HTTP calls to the vulnerable microservices, associated with a specific Netcat command, to initialize reverse shells. The script listens for connections, and once a reverse shell is established, the script deploys other scripts for lateral movement and scanning. This process repeats until either the attacker reaches the backend or, depending on the LMP, it starts another attack on the front-end.

Results - To evaluate the efficiency of the honeypots strategy, we consider several metrics related to lateral movement attacks:

1. The average time to success — the time it takes for an attacker to reach the target microservice or database;
2. The average number of steps to reach the target $\hat{\tau}_M$;
3. The average duration of an attack;
4. The average number of steps per attack $\hat{\tau}$;
5. The average number of attacks to reach the target $\hat{N}$.

We compare the results for the myopic and threshold policies with a honeypot reachable from each microservice, against a baseline policy with no honeypots, where the attacker always succeeds in one attack (Table 1). Experiments were conducted on a chain of 6 microservices. The optimal myopic policy was found using the theoretical curve (eq. (1)), and the best threshold policy is when $\sigma^* = 6$ is exactly the number of microservices to reach the target. For myopic and threshold policies, with honeypots, the attack (until success) was executed 10 times. The attack was executed 44 times without honeypots, as a baseline. In the baseline setup, the first attack always succeeds as no honeypot can deceive and trap the attacker. The attack then takes an average of 71 seconds to reach the target, or exactly 5 lateral movement steps. However, with one honeypot reachable per microservice, or a total of five honeypots, the attacker can get stuck in honeypots and now needs to start new attacks with no knowledge over which microservice is genuine or not.
Table 1 shows the effectiveness of adding honeypots. The optimal myopic policy $p^* = 0.173$ takes an average time to success of 3309 seconds and an average of 221 total steps, about 45 times the average time to success of the baseline. The optimal threshold policy with $\sigma^* = 6$ also performs worse than the baseline, taking an average time to success of 1644 seconds and an average of 75 steps. This represents 23 times the average time to success of the baseline, and about 15 times the number of steps. In practice, the threshold policy is used by experienced attackers that gather intelligence on the system. Interestingly, the only way for the attacker to follow the optimal myopic policy is to know the chain length, meaning that the myopic policy is used only when no information on the system is known, as the threshold policy may have no chance of reaching the target.

5 Conclusion

Lateral movement attacks are a critical threat in cloud environments since they allow attackers to explore and compromise various internal cloud sub-systems. Honeypots are traditionally used to deceive attackers and collect attack information. As shown in this paper, they can also serve as a dynamic shield against lateral movement attacks. However, while honeypots deter attackers from reaching their target, honeypots are expensive to deploy and maintain. In this regards, our analysis supports reference performance to dimension the number of honeypots that are deployed for effective defense. We measured the practical impact of the scheme by executing lateral movement attacks in a Kubernetes environment. By adding just 5 honeypots on a 6 microservices chain, the attack average time increases significantly: about 45 times for naive attackers, and 25 times for experienced attackers.

Our model can be extended on the attacker side and defender side. On the attacker side, the model may take into account the attacker's expertise, i.e., the ability to distinguish genuine microservices from honeypots, which translates into a probability α that equals to 1 for a skilled attacker and 0 for a naive attacker. During each lateral movement, the attacker determines if the microservice corresponds to a genuine microservice (in which case the attacker continues the attack) or a honeypot, in which case the attacker starts a new attack. The transition probabilities $\mathbb{P}(S(t+1)|S(t))$ corresponds to:

$$\mathbb{P}((k,h+1)|(k,h)) = (1-\alpha)\pi(k+h), \forall h > 0, \forall k \neq M,$$
$$\mathbb{P}((1,0)|(k,h)) = 1 - \pi(k+h) + \alpha\pi(k+h), \forall h > 0, \forall k \neq M.$$

Note that the average number of attacks does not change and the average length of attacks depends on the probability of detection α .

On the defender's side, rather than assuming a uniform repartition of honeypots (i.e. $q_k = 1 - \frac{1}{L}$), we suppose that the closer we get to the target, the more honeypots are deployed. Thus, a defender makes it increasingly difficult for each attacker's step to succeed.An extreme case refers to an exponentially distributed number of honeypots.

Finally, the microservice architecture can be extended to deal with a directed acyclic graph rather than a chain. In such a case, the system is modeled as a Markov chain with one absorbing state (corresponding to the ultimate target), and, the time to reach the absorbing state follows a phase-type distribution (Section 2.5 in [9]).

References

1. Brémaud, P.: Markov chains: Gibbs fields, Monte Carlo simulation, and queues. Springer Science & Business Media (1999)
2. Chowdhary, A., Sengupta, S., Huang, D.e.a.: Markov game modeling of moving target defense for strategic detection of threats in cloud networks. In: Workshop on Artificial Intelligence for Cyber Security (2019)

3. Fraunholz, D., Schneider, D., Zemitis, J.e.a.: Hack my company: An empirical assessment of post-exploitation behavior and lateral movement in cloud environments. In: Central European cybersecurity conference (2018)
4. Horák, K., Bošanský, B., et al., P.T.: Optimizing honeypot strategies against dynamic lateral movement using partially observable stochastic games. Computers & Security **87** (Nov 2019)
5. Huang, L., Zhu, Q.: Adaptive honeypot engagement through reinforcement learning of semi-markov decision processes. In: International conference on decision and game theory for security (2019)
6. Huang, L., Zhu, Q.: Farsighted risk mitigation of lateral movement using dynamic cognitive honeypots. In: International conference on decision and game theory for security (2020)
7. Iqbal, S., Kiah, M.L.M., Dhaghighi, B.e.a.: On cloud security attacks: A taxonomy and intrusion detection and prevention as a service. Journal of Network and Computer Applications **74** (Oct 2016)
8. Kuwatly, I., Sraj, M., Al Masri, Z.e.a.: A dynamic honeypot design for intrusion detection. In: IEEE/ACS International Conference on Pervasive Services (2004)
9. Latouche, G., Ramaswami, V.: Introduction to matrix analytic methods in stochastic modeling. Society for Industrial and Applied Mathematics (1999)
10. MITRE: MITRE ATTACK Framework. `https://attack.mitre.org` (2025)
11. Ngo, H.Q., Guo, M., Nguyen, H.: Catch me if you can: Effective honeypot placement in dynamic ad attack graphs. In: IEEE Conference on Computer Communications (2024)
12. OWASP Foundation: OWASP: Common Attack Types. `https://owasp.org/www-community/attacks` (2025)
13. Pawlick, J., Nguyen, T.T.H., Colbert, E.e.a.: Optimal timing in dynamic and robust attacker engagement during advanced persistent threats. In: IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (2019)
14. Poeng, K., Schumacher, L.: Lateral movement identification in cross-cloud deployment. In: IEEE International Conference on Network and Service Management (2024)
15. Ross, S.M.: Stochastic processes. John Wiley & Sons, 2nd edn. (1995)

Proof of theorem 1

Proof. Let $f = \mathbb{P}(\tau_b(1) < \infty | S(0) = S_0)$. In particular, as the attacker is rational, $f = 1 - \mathbb{P}(X(M-1) = M)$ so that

$$1 - f = \mathbb{P}(S(t) = (t,0), t = 2, \ldots, M | S(0) = (1,0)) = \prod_{k=1}^{M-1} (1 - q_k)\pi(k)$$

Case i. is trivial. In order to prove case ii., we observe that $N = N_{S_0} + 1$, where N_{S_0} is the number of returns to state $S_0 = (1,0)$, i.e., $N_{S_0} = \sum_{t>0} \mathbb{1}\{S(t) = S_0)\}$. In our case, the number of returns to a given state has probability distribution $\mathbb{P}(N_{S_0} = h | S(0) = S_0) = (1-f)f^h$ for $h \geq 0$ [1][Thm. 7.2, pp. 86]. By identifying $\rho = 1 - f$, this concludes the proof.

Sketch of proof of theorem 2

Proof. i. In order to prove that the attack lengths are independent, observe that, by definition, $\{\tau_b\}$ are the successive return times to $S(0)$. As such, they are stopping times for the process $S(t)$. Hence, from the Strong Markov Property, [1][Thm. 7.1, pp. 85], $\tau(k)$ is independent of any $\tau(h)$, $h > k$. Also, because the Markov chain is homogeneous, they are identically distributed, thus we just study the probability distribution of $\tau(1)$.

ii. We provide a sketch of proof. The formal proof requires to introduce hitting time theory and related derivations, which are out of the scope of the present work. The distribution of a finite return time $\mathbb{P}(\tau(1) = h)$ is obtained by conditioning on the fact that the target is never reached. Thus, the length of a failed attack only depends on the event of continuing the attack for $h-1$ steps and then terminating the attack at step h. In turn, this depends just on the policy π. Taking previous remark into account, we have:

$$\begin{aligned}
\mathbb{P}(\tau(1) = h) &= \mathbb{P}(X(h) = 1, \bigcap_{t=1}^{h-1} \{X(t) \notin \{1, M\}\} | S(0) = S_0, X(t) \neq M\}) \\
&= \mathbb{P}(X(h) = 1, X(h-1) \notin \{1, M\} | X(t) \neq M) \\
&\times \prod_{t=1}^{h-2} \mathbb{P}(X(t+1) \notin \{1, M\} | X(t) \notin \{1, M\}) \\
&\times P(X(1) \notin \{1, M\} | S(0) = S_0, X(t) \neq M)
\end{aligned}$$

from which the result follows.

Proof of proposition 1

Proof. We first recall basic facts about probability generating functions. Let $W = \sum_{j=1}^{N} X_j$, where $\{X_j\}$ are i.i.d. integer nonnegative random variables and

N is a random natural number, then $G_W(z) = G_N(G_X(z))$, so that $G_{W-1}(z) = \frac{G_N(G_X(z))}{G_X(z)}$. The proof is based on a different rewriting of (2) using an alternative expression of (1) as

$$J(\pi) = \mathbb{E}_\pi\left[\sum_{m=1}^{M-1} \gamma^{\sum_{j=1}^{N-1}\tau(j)}\gamma^m + \sum_{k=1}^{N-1}\sum_{i=1}^{\tau(k)} \gamma^{\sum_{j=1}^{k-1}\tau(j)}\gamma^i\right] = J_1(\pi) + J_2(\pi)$$

The first term $J_1(\pi)$ writes as $J_1(\pi) = \frac{\gamma-\gamma^M}{1-\gamma}\mathbb{E}_\pi\left[\gamma^{\sum_{j=1}^{N-1}\tau(j)}\right] = \frac{1-\gamma^M}{1-\gamma}\frac{G_N(G_{\tau(1)}(\gamma))}{G_{\tau(1)}(\gamma)}$.

Similarly, we obtain

$$\begin{aligned}
J_2(\pi) &= \mathbb{E}_\pi\left[\sum_{k=1}^{N-1}\mathbb{E}\left[\gamma^{\sum_{j=1}^{k-1}\tau(j)} \mid N\right]\mathbb{E}\left[\sum_{i=1}^{\tau(k)}\gamma^i \mid N\right]\right] \\
&= \gamma\mathbb{E}_N\left[\sum_{k=1}^{N-1}\mathbb{E}\left[\frac{1-\gamma^{\tau(k)}}{1-\gamma} \mid N\right]\prod_{j=1}^{k-1}\mathbb{E}\left[\gamma^{\tau(j)} \mid N\right]\right] \\
&= \gamma\frac{1-G_{\tau(1)}(\gamma)}{1-\gamma}\mathbb{E}\left[\sum_{k=1}^{N-1}G_{\tau(1)}^{k-1}(\gamma)\right] = \gamma\frac{1-G_{\tau(1)}(\gamma)}{1-\gamma}\mathbb{E}\left[\frac{1-G_{\tau(1)}^{N-1}(\gamma)}{1-G_{\tau(1)}(\gamma)}\right] \\
&= \frac{\gamma}{1-\gamma}\frac{1-G_N(G_{\tau(1)}(\gamma))}{G_{\tau(1)}(\gamma)}
\end{aligned}$$

Now, replacing the expression $G_N(z) = \frac{z\rho}{1-(1-\rho)z}$, the statement follows.

Proof of proposition 2

Proof. i. For $M = 2$ the statement is obvious. For $M > 2$, the optimal policy solves the equation $\frac{d}{dp}J(p) = 0$, which writes $f(p) = g(p)$, where

$$f(p) = L^{M-1} - (1-p)^{M-2} \text{ and } g(p) = \frac{p(1+p)}{1-p}L^{M-1}(M-2)$$

In particular, f is a increasing concave, whereas g is an increasing convex. Since $f(0) = L^{M-1} - 1$ and $f(1) = L^{M-1}$ and $g(0) = 0$ and $g(1) = +\infty$, it is easy to see that the solution in $(0, 1)$ exists and is unique.

ii. We now observe that, for $M > 2$, the solution $p^* \in [p_{min}, p_{max}]$, where p_{min} solves for $g(p_{min}) = L^{M-1} - 1 = f(0)$ and p_{max} solves for $g(p_{max}) = L^{M-1} = f(1)$. For the sake of notation, we can express both p_{min} and p_{max} as the solutions of

$$L^{M-1} - \chi = \frac{p(1-p)}{1-p}L^{M-1}(M-2) \tag{6}$$

where $\chi = 1$ to determine p_{max} and $\chi = 0$ for p_{min}. If we define $A := \frac{L^{M-1}-\chi}{(M-2)L^{M-1}}$ then (6) becomes $\frac{p(1+p)}{1-p} = A$. For p_{max}, we can write

$$\frac{1}{M-2} = \frac{p_{max}(1+p_{max})}{1-p_{max}} \geq \frac{p_{max}}{1-p_{max}}$$

from which $p_{max} \leq \frac{1}{M-1}$. For p_{min}, from (6),

$$p = \frac{A+1}{2}\Big(-1+\sqrt{1+\frac{4A}{(A+1)^2}}\Big) \tag{7}$$

By leveraging the series expansion $(1+x)^\alpha = \sum_{n=0}^{+\infty} x^k$, we obtain $p_{min} = \frac{A}{A+1} + o(4A/(A+1)^2)$, where $4A/(A+1)^2 < 1$. Observe that $\frac{A}{A+1} \xrightarrow{L} \frac{1}{M-1}$, so that $p_{min} = \Omega(\frac{1}{M-1})$, which concludes the proof.