Lateral Movement Identification in Cross-Cloud Deployment

1st Kokthay Poeng
Faculty of Computer Science
University of Namur
Namur, Belgium
kokthay.poeng@unamur.be

2nd Laurent Schumacher
Faculty of Computer Science
University of Namur
Namur, Belgium
laurent.schumacher@unamur.be

***Abstract*—In the cloud computing era, cross-cloud deployments enable organizations to operate across multiple autonomous cloud platforms, offering advantages such as resilience, cost and performance optimization. However, lateral movement attacks, which are critical in the progression of Advanced Persistent Threats (APTs), pose significant challenges in this environment. This paper proposes a Lateral Movement Identification (LMD) system to identify lateral movement attacks in cross-cloud containerized environments. The LMD system utilizes Dynamic Information Flow Tracking (DIFT) and extended Berkeley Packet Filter (eBPF) sandboxes to monitor and associate network traffic within container host kernel without kernel modification. Our experiments validate the efficiency of the LMD system in tracking ingress and egress traffic, differentiating between multiple simultaneous connections, and maintaining minimal performance overhead.**

***Index Terms*—Lateral Movement, Identification, Cross-cloud Deployment**

I. Introduction

In the cloud computing era, cross-cloud deployments are designed to operate across multiple autonomous cloud platforms rather than being confined to a single provider. Such deployments offer several advantages for organizations [1]. First, they enhance resilience and availability by enabling failover to alternative providers during outages or disruptions. Cost optimization is another benefit, as organizations can select the most cost-effective services from each provider on the spot. Furthermore, they optimize performance by leveraging the geographic reach of various providers.

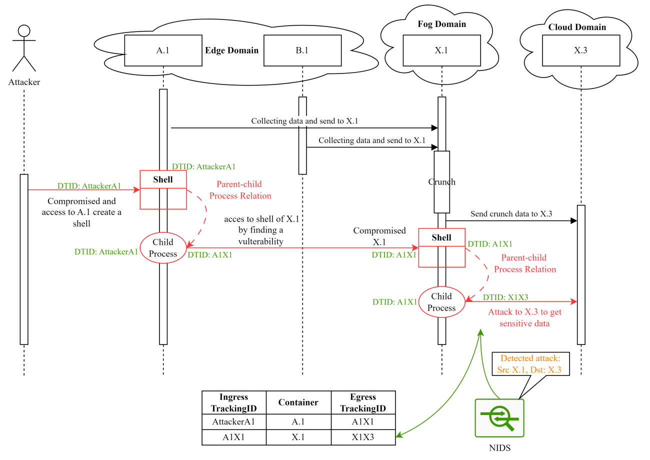
However, lateral movement attacks pose the most critical challenge in a cross-cloud environment due to multiple cloud platforms interconnection. Lateral movement plays an important role in the progression of APTs [2]–[4]. APTs are sophisticated and sustained cyberattacks designed to breach security defenses and remain undetected. Over an extended period, they aim to extract valuable data and resources. These threats are particularly dangerous because they often involve attackers gaining initial access to a system and then moving laterally within the network to exploit vulnerabilities to target both sensitive data and resources. In a cross-cloud environment, the complexity and scope of lateral movement attacks are amplified, increasing the attack surface and complicating the implementation of consistent security measures across multiple platforms. Current network defenses, such as Network Intrusion Detection System (NIDS) [5], have visibility into the source and destination of detected malicious traffic. However, they are unable to identify lateral movement attacks due to a lack of visibility into intermediate hosts.

We propose LMD approach to identify lateral movement attacks in cross-cloud containerized deployments supporting dynamic Transport layer protocols. LMD focused on container environment because cloud computing industries are rapidly moving towards containerization [6]. The core concept of LMD is DIFT, which is used to track the traffic flow of lateral movement in kernel space. LMD relies on a NIDS to detect attacks. The eBPF sandbox is used to capture, track, and associate lateral movement attacks within the kernel space of each container host, all without modifying the kernel source code. Additionally, eBPF enhances this setup by efficiently monitoring and capturing system calls, network activity, and filesystem interactions of containers within the kernel space of the container host.

II. Motivation and Example

For example, in an edge-fog-cloud scenario, there are four containers: A.1 and B.1 are edge containers, X.1 is a fog container, and X.3 is a cloud container storing sensitive data. A.1 and B.1 collect data and send it to X.1, which processes the data and sends it to X.3. We assume that the attacker aims to obtain sensitive information from X.3 by using lateral movement and exploiting trusted relationships within the edge-fog-cloud network. The attacker compromises A.1 and manages to find a vulnerability in X.1 that enables him/her to compromise X.1. After compromising X.1, the attacker launches an attack on X.3 to access sensitive information.

In Figure 1, we assume that an attacker is able to compromise A.1 and create a shell, taking advantage of resource constraints at the edge that prevent the deployment of adequate security measures. The attacker then uses this shell to find a vulnerability in X.1. Using the same shell, the attacker remotely access X.1, making the process that accesses X.1 from A.1 a child process of the shell. After remotely accessing X.1 a shell is created. The attacker uses this shell to attack X.3 to obtain sensitive information. This means that the process the attacker uses to attack X.3 from X.1 is a child process of the


Attacker
A.1
Edge Domain
B.1
Fog Domain
X.1
Cloud Domain
X.3
Collecting data and send to X.1
Collecting data and send to X.1
DTID: AttackerA1
Shell
DTID: AttackerA1
Parent-child Process Relation
Crunch
Compromised and access to A.1 create a shell
acces to shell of X.1 by finding a vulterability
Send crunch data to X.3
Compromised X.1
Shell
DTID: AttackerA1
Child Process
DTID: A1X1
DTID: A1X1
DTID: A1X1
Parent-child Process Relation
DTID: A1X1
Child Process
DTID: X1X3
Attack to X.3 to get sensitive data
Detected attack: Src X.1, Dst: X.3
NIDS

Ingress TrackingID	Container	Egress TrackingID
AttackerA1	A.1	A1X1
A1X1	X.1	X1X3

Fig. 1. Example of Lateral movement in a edge-fog-cloud scenario

shell process. In the X.3 cloud environment, the NIDS detects attack traffic from X.1 to X.3, but the NIDS only identifies X.1 as the source of the attack. However, with our LMD system, the NIDS is able to trace the attack back to all affected Containers.

III. Related Work

Tian et al. introduced CloudSEC [7], utilizing the Evidence Reasoning Network (ERN) for real-time lateral movement detection in cloud-edge environments. Their framework incorporates EventTracker for user activities and AlertCorrelator for intrusion alert analysis. A notable feature is vulnerability correlation, representing interconnected software defects exploitable in multi-stage attacks. They construct a directed graph ERN by capturing and correlating system calls made by users using EventTracker and correlating network vulnerabilities between hosts. When attacks are detected by NIDS in AlertCorrelator, CloudSEC constructs a sub-graph of ERN named Evidence Chain Reasoning to show the attack steps and hosts that are compromised. To determine which hosts are compromised, CloudSEC conducts a breadth-first search to find intermediate and initial hosts in the ERN graph based on timing. This determination can lead to false positives.

The P4Control framework proposed by Osama et al. [8] uses Decentralized Information Flow Control (DIFC) in the network layer with P4 switches and eBPF to prevent cross-host or lateral movement attacks at line-rate in enterprise environments. P4Control generates DIFC labels for network entities (such as hosts and packets) and system entities (like processes and files), propagating these labels across intra-host and inter-host flows. eBPF is used to monitor and propagate DIFC labels within hosts, while customized network packets carry these labels between Protocol and Payload fields. The Network Control Language (NETCL) is developed to enforce DIFC policies. NETCL policies in P4 switches can prevent data exfiltration, identify unauthorized access, downgrade information classification and restrict access to sensitive files, thus mitigating malware spread. However, there are several limitations of this framework when moving to cross-cloud environment. First, the framework modified the standard network packet by adding a new field between the protocol field and payload field to propagate the label, and the authors did not mention how to cooperate with the QUIC protocol since the protocol encrypts everything after the protocol field. Moreover, hyperscales like Meta have started investigating the QUIC protocol to replace TCP as the backbone network between their data centers and for communication between origin servers and cache servers of their CDN network since 2022 [9]. Second, the framework proposed using one bit to represent one category, but in cross-cloud scenario, this is insufficient. If the DIFC field offers only 256 bits, it can handle only 256 containers. Third, the P4Control solution aims to prevent traffic labeled from one or more specific domains from accessing other specific domains (e.g. the sales department cannot access the server network). However, in a cross-cloud environment, it is not feasible to strictly block traffic between domains because the purpose of a cross-cloud environment is to enable resources across multiple clouds to communicate with each other for cross-cloud deployment. Moreover, P4Control faces increased overhead in container environments, as the framework requires interrupting network packet processing to attach and detach DIFC labels in each host.

RTAG [10] introduces a novel system for efficient cross-host attack investigation by leveraging DIFT. It optimizes the tagging and tracking of data flows across multiple hosts, reducing the overhead typically associated with such systems. By decoupling tag dependency management and supporting both runtime and replay instrumentation, RTAG significantly improves the accuracy and performance of provenance tracking while maintaining minimal time and memory costs. This makes it particularly effective for large-scale distributed environments where traditional DIFT systems might struggle with performance bottlenecks and complexity. However, to associate traffic from one host to another using the UDP protocol, RTAG modifies the standard UDP packet in front of the payload field to attach information necessary for constructing tags on the receiving host. Consequently, RTAG also faces challenges when dealing with QUIC protocol traffic. Moreover, RTAG requires modifications to the kernel source code in order to deploy the system.

Inspired by these works, our LMD system utilizes DIFT in kernel space, relying on NIDS to identify lateral movement traffic flows within a cross-cloud environment without modifying standard network packets or kernel code. This approach supports dynamic Transport layer protocols, including QUIC, and reduces false positives in identification, thereby improving upon previous works.

IV. Overview

In our work, we propose the LMD approach designed to identify lateral movement within container environments across multiple cloud domains using DIFT. LMD relies on NIDS; when NIDS detects an attack, the LMD determines whether the attack involves lateral movement based on packet information from the attack. If lateral movement is identified, LMD identifies the affected containers and isolates them from the network. Within each affected container, LMD traces the attack footprint, identifying the processes involved. This is achieved using eBPF to capture, track, and associate process activity and network traffic within kernel space. A DIFT Tracking ID (DTID), constructed from network flow attributes (Source IP address, Source Port, Destination IP address, Destination Port), is used to uniquely identify and trace associated traffic flows and processes across the environment.

Threat Model and Assumptions. Our threat model is consistent with prior work on host-level auditing systems [10]–[13]. We assume the presence of attackers attempting to access, modify, or exfiltrate unauthorized resources by exploiting relationships within container networks and cross-cloud environments. We assume that the container host kernel, the eBPF sandbox, and the Flow Association Map are secure, even if containers are compromised. Attacks are assumed to occur after our system has been initialized, allowing us to capture relevant attack information from the beginning. We further assume that attackers cannot alter network packets during transfer between containers, preventing man-in-the-middle attacks. To mitigate privilege escalation risks, we recommend not granting containers privileges that could affect the container host kernel. If such privileges are required, enhanced security measures [14]–[18] can be employed to further secure eBPF operations. Our threat model does not account for attacks involving hardware trojans, side/covert channels, or malicious administrators who might disable eBPF or tamper with the Flow Association Map.

V. Lateral Movement Identification (LMD)

To achieve the goals of LMD, we deploy an eBPF sandbox on each container host across multiple cloud domains. This sandbox captures, tracks, and associates process activity, filesystem operations, and ingress/egress traffic at the host level using DTIDs. The DTID is constructed from a combination of Source IP, Source Port, Destination IP, and Destination Port (e.g. `trackingID = SrcIP + SrcPort + DstIP + DstPort`).

Ingress Traffic. When a container receives ingress traffic, the corresponding DTID is constructed and stored along with the process ID (PID) that accepted the traffic in a BPF map named `tracking_PID`. To capture PIDs and construct the DTID, we hooked our eBPF program into kernel functions responsible for handling ingress traffic (e.g. `skb_consume_udp`, `netif_receive_skb`, etc.).

Process Communication. The DTID from ingress traffic is propagated to related or communicating processes and files within the container. Process communication, including child-parent relationships and Inter-Process Communication (IPC) mechanisms (e.g. UNIX Domain Sockets, shared memory, semaphores, message queues), is tracked by monitoring relevant system calls (e.g. `clone()`, `exec()`, `msgsnd()`, `msgrcv()`, `bind()`, `connect()`, etc.). Files created or modified by processes are also tracked to propagate the DTID to the process that reads or opens the file.

For instance, when a new process is created, the sandbox updates the `tracking_PID` map with the new PID linked to the same DTID as its parent process, if the parent PID of the new process exists in the `tracking_PID` map. Similarly, if Process A communicates with Process B via IPC, the PID of Process B is updated in the `tracking_PID` map with the DTID of Process A. Files written or created by Process A are logged in another BPF map named `tracking_file` with their associated index node (inode) and DTID. When a new process B reads or opens the same file, the sandbox updates the PID of Process B with the DTID (retrieved from `tracking_file` by searching the inode) in the `tracking_PID` map.

Egress Traffic. To monitor egress traffic, the eBPF program is hooked to relevant kernel functions responsible for handling egress traffic (e.g. `tcp_v4_connect`, `udp_send_skb`, etc.). When egress traffic is generated, the sandbox checks the `tracking_PID` map for the PID of the egress process. If a match is found, it logs the associated ingress DTID and egress DTID in a map named `flow_associate`, which is shared globally across NIDSs and container hosts.

When the NIDS detects an attack, it uses the DTID constructed from the detected malicious packet information to determine if the attack involves lateral movement using the shared `flow_associate` map. If lateral movement is identified, LMD searches for all affected containers and enables each container host to isolate them from the network.

VI. Results

To evaluate the efficiency of the flow association map, the ability of the system to differentiate between multiple simultaneous connections, and performance overhead introduced by LMD, we conducted a series of tests in a controlled environment consisting of three Docker hosts, each hosting two containers. There were two different scenarios for this test: container-to-container communication across different hosts and within the same host.

Flow Association Map. We initiated an SSH connection from one container to another. From this session, we tested three protocols: SSH to a third container, SCP file transfer, and a QUIC connection from the second container (as a client) to the third (as a server). Additionally, we used Data Exfiltration Toolkit and Netcat to transmit a file through an intermediate container using both UDP and TCP protocols. The flow association map successfully captured and associated the ingress and egress traffic in all these scenarios, demonstrating the ability of LMD to track and correlate network flows accurately in complex, multi-hop scenarios, ensuring that lateral movement across containerized environments is correctly identified.

Distinction multiple simultaneous connections. To test the ability of LMD to distinguish between multiple simultaneous connections and reduce false positives, we initiated SSH connections from six hundred different containers to a single server container simultaneously. The eBPF program in the server container successfully distinguished the hundred ingress flows, accurately associating each with its respective PID without any false positives.

Performance Overhead. The LMD system was tested by measuring bandwidth during container-to-container communication. An SSH connection was established between containers, followed by an iperf3 test to a third container. Results showed minimal overhead on the same host with LMD enabled, increasing by just 0.014%. When testing across different hosts, no measurable overhead was detected. This low impact is due to the efficient design of the eBPF program, which monitors packet data without disrupting packet processing.

Uniqueness of DIFT tracking ID. DTID generated by the LMD system is crucial for accurate traffic flow association. The DTID is composed of the Source IP, Source Port, Destination IP, and Destination Port. In a container environment, there are various network configurations (e.g. NAT, direct physical networks, etc.). This means that the uniqueness of DTIDs depends on the network configuration of each individual container environment. However, it is essential for the DTIDs to be visible to both NIDSs and receiving container hosts. The main idea is to enable NIDSs to construct the DTID and identify lateral movement based on packet header information.

The results validate that the LMD system efficiently associates ingress and egress traffic across containers in a cross-cloud environment with minimal performance overhead. They also highlight the limitations of the uniqueness of DTID within a container host.

VII. Conclusion and Future work

This study presents LMD, an improved approach to identifying lateral movement attacks in cross-cloud containerized environments. By leveraging DIFT and eBPF sandboxes without modifying kernel code, the LMD system effectively tracks and associates network flows across containers, ensuring accurate identification of lateral movement with minimal performance impact. The results demonstrate that the LMD system can differentiate between multiple simultaneous connections without generating false positives and imposes minor performance overhead on container-to-container communication. The LMD system also supports dynamic Transport layer protocols by relying on IP and port-based tracking, avoiding the need for network packet modifications. This capability further enhances adaptability and effectiveness of LMD in various network environments. However, the study also identifies the limitations of DTID uniqueness within a container host, particularly concerning the constraints imposed by the network configuration. Despite these limitations, the LMD system offers a solution for enhancing security in cross-cloud deployments, addressing a critical gap in existing cloud security frameworks.

Future work includes enhancing the LMD system to support container orchestration platforms, such as Kubernetes. We also plan to conduct simulated attacks to test efficiency of LMD, fully integrate it with NIDS, and establish fine-grained investigation capabilities for filesystems, commands, and processes used in attacks. Additionally, we will improve the integrity of packet headers, investigate the possibility of false positives and false negatives, demonstrate LMD in real-world scenarios, and provide a detailed evaluation of its effectiveness and its impact on both network and container host performance.

Acknowledgment

This research is supported by Belgium Walloon Region CyberExcellence Program (Grant #2110186).

References

[1] J. Hong, T. Dreibholz, J. A. Schenkel, and J. A. Hu, "An overview of multi-cloud computing," in *Web, Artificial Intelligence and Network Applications: Proceedings of the Workshops of the 33rd International Conference on Advanced Information Networking and Applications (WAINA-2019) 33*. Springer, 2019, pp. 1055–1068.

[2] Fortinet. (2024) Advanced persistent threat (apt). [Online]. Available: https://www.fortinet.com/resources/cyberglossary/advanced-persistent-threat

[3] B. Lenaerts-Bergmans. (2023) Lateral movement. [Online]. Available: https://www.crowdstrike.com/cybersecurity-101/lateral-movement/

[4] P. Chen, L. Desmet, and C. Huygens, "A study on advanced persistent threats," in *Communications and Multimedia Security: 15th IFIP TC 6/TC 11 International Conference, CMS 2014, Aveiro, Portugal, September 25-26, 2014. Proceedings 15*. Springer, 2014, pp. 63–72.

[5] Snort. Snort - network intrusion detection & prevention system. [Online]. Available: https://www.snort.org/

[6] AWS. What are cloud containers? [Online]. Available: https://aws.amazon.com/what-is/cloud-containers/

[7] Z. Tian, W. Shi, Y. Wang, C. Zhu, X. Du, S. Su, Y. Sun, and N. Guizani, "Real-time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4285–4294, 2019.

[8] O. Bajaber, B. Ji, and P. Gao, "P4control: Line-rate cross-host attack prevention via in-network information flow control enabled by programmable switches and ebpf," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2024, pp. 147–147.

[9] T. Ingale. (2022) Watch meta's engineers discuss quic and tcp innovations for our network. [Online]. Available: https://engineering.fb.com/2022/07/06/networking-traffic/watch-metas-engineers-discuss-quic-and-tcp-innovations-for-our-network/

[10] Y. Ji, S. Lee, M. Fazzini, J. Allen, E. Downing, T. Kim, A. Orso, and W. Lee, "Enabling refinable {Cross-Host} attack investigation with efficient data flow tagging and tracking," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1705–1722.

[11] M. N. Hossain, S. M. Milajerdi, J. Wang, B. Eshete, R. Gjomemo, R. Sekar, S. Stoller, and V. Venkatakrishnan, "{SLEUTH}: Real-time attack scenario reconstruction from {COTS} audit data," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 487–504.

[12] P. Fang, P. Gao, C. Liu, E. Ayday, K. Jee, T. Wang, Y. F. Ye, Z. Liu, and X. Xiao, "{Back-Propagating} system dependency impact for attack investigation," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 2461–2478.

[13] P. Gao, X. Xiao, Z. Li, F. Xu, S. R. Kulkarni, and P. Mittal, "{AIQL}: Enabling efficient attack investigation from system monitoring data," in *2018 USENIX Annual Technical Conference (USENIX ATC 18)*, 2018, pp. 113–126.

[14] Y. He, R. Guo, Y. Xing, X. Che, K. Sun, Z. Liu, K. Xu, and Q. Li, "Cross container attacks: The bewildered {eBPF} on clouds," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 5971–5988.

[15] J. Corbet. (2023) Finer-grained bpf tokens. [Online]. Available: https://lwn.net/Articles/947173/

[16] S. Y. Lim, X. Han, and T. Pasquier, "Unleashing unprivileged ebpf potential with dynamic sandboxing," in *Proceedings of the 1st Workshop on eBPF and Kernel Extensions*, 2023, pp. 42–48.

[17] A. Ahmad, S. Lee, and M. Peinado, "Hardlog: Practical tamper-proof system auditing using a novel audit device," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1791–1807.

[18] R. Paccagnella, P. Datta, W. U. Hassan, A. Bates, C. Fletcher, A. Miller, and D. Tian, "Custos: Practical tamper-evident auditing of operating systems using trusted execution," in *Network and distributed system security symposium*, 2020.