A policy compliance detection architecture for Digital Data Marketplaces (DDM)

Lu Zhang, Reginald Cushing, Cees de Laat, and Paola Grosso

MultiScale Networked Systems (MNS) lab, University of Amsterdam, Science Park 904, Amsterdam, The Netherlands

Email: l.zhang2@uva.nl, r.s.cushing@uva.nl, delaat@uva.nl, p.grosso@uva.nl

The project Data Logistics for Logistics Data (DL4LD)[1] aims to facilitate secure and trustworthy data sharing among Dutch logistic partners with the concept of Digital Data Marketplaces (DDM). With the definition of our project, a DDM is a digital infrastructure that facilities policy-driven data exchange applications. For instance, different DDM parties may want to gather their local data together and run a machine learning (ML) algorithm on their joint data, so that they can gain benefits from a more accurate prediction model. Those parties could be competing, so they concern about the confidentiality of their data and whether the computing result is trustworthy. In a DDM, there is a unique identifier for each data and compute object. The parties agree on permissible actions on specific data and compute objects and express them into a policy. The compute objects are containerised for better portability. A container image is a lightweight, executable package including source code, program runtime and libraries. It is crucial to build components in the digital infrastructure that enforces the policy in the data exchange application.

At ICT.Open, we will present an architecture that can effectively detect policy compliance with Linux system call monitoring during the execution stage. The Linux system calls are an interface between an application and the Linux kernel. As illustrated in Figure 1, the architecture comprises multiple modules, a *Profile Generation and Validation* module (in purple), a *OC-SVM based distributed intrustion detection systems (IDS)* module (in orange) and a *Sanitization* module (in red). They collaboratively enforce the policy in the execution stage by analysing system calls generated by a running container. For each compute object, a profile and an anomaly detection model are built or trained if it was used for the first time. These initial processes are conducted in a secure environment, our trusted 3rd party (in yellow), and they are distributed to endpoint execution platforms (in blue) in a secure manner.

First, we implement a *Profile Generation and Validation* module that can discriminate the algorithm running inside the container only by externally monitoring. We profile runtime behaviours of a specific containerised algorithm with frequency distributions of n-grams of system call symbols. In a endpoint execution platform, it computes the similarity, cross entropy, between the observed system calls and the profile of the authorised algorithm. The computed results are allowed to leave the container only if they match. This module helps to ensure that only the authorised algorithm can access on a particular data object. [1].

Secondly, we implement a distributed real-time intrusion detection system. We adopt One Class Support Vector Machine (OC-SVM) as the anomaly detection algorithm due to its capability of dealing with complex non-linear problems. To detect malicious behaviours in a real-time manner, the streaming system calls are separated into segments before being mapped into feature vectors. We also apply the signature-based methodology to reduce false alarms. [2]

To adapt to the dynamic characteristics of the algorithm behaviour, the anomaly detection model is retrained whenever new data is available, shown as the red arrows in Figure 1. This may provide

[1] https://www.dl4ld.nl/

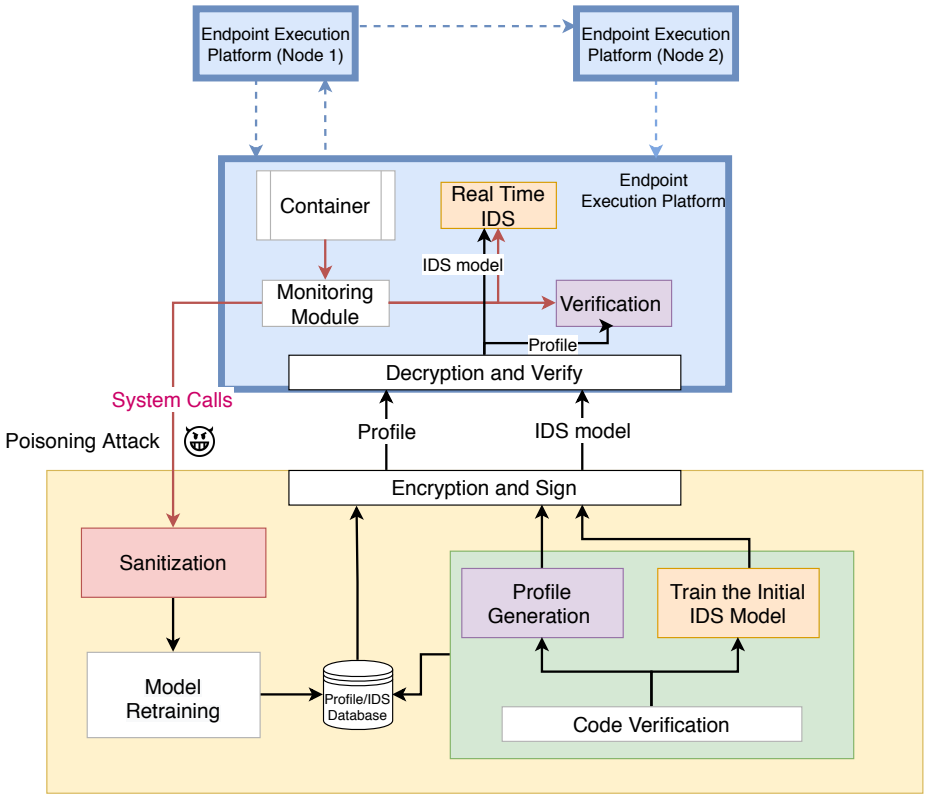

Figure 1: The architecture of a DDM policy enforcement component.

opportunities for adversaries to carry out attacks by poisoning the collected data and degrade the IDS' performance. It is vital to ensure that the retraining data are attack-free. Last but not least, we implement a *Sanitization* module in a trusted 3rd party to filter out malicious samples. The sanitization process is based on the DBSCAN clustering algorithm because it does not require any pre-knowledge of the normal data and can separate clusters of any shape.

In addition, We will present our experiments results that demonstrate the effectiveness of our proposed architecture with a DL4LD use case.

References

[1] L. Zhang, R. Cushing, R. Koning, C. de Laat, and P. Grosso, "Profiling and discriminating of containerized ml applications in digital data marketplaces (ddm)." in *ICISSP*, 2021, pp. 508–515.

[2] L. Zhang, R. Cushing, C. Koning, and P. Grosso, "A real-time intrusion detection system based on oc-svm for containerized applications," in *The 24th IEEE International Conference on Computational Science and Engineering*, 2021, pp. 508–515.