Securing Microservice Cluster Management Cockpit using Blockchain
Bachelor thesis/Master thesis/Guided research

Supervisor: Prof. Dr. Alexander Pretschner
Advisor: Mohsen Ahmadvand
Email: ahmadvan@in.tum.de
Phone: +49 89 289 – 17, 314
Starting date: Summer term 2018

Context
Microservices together with containers have significantly altered cloud-based service development and delivery. On the positive side, the new paradigm offers better resource utilization (vertical scalability), improves productivity (specialized microservices) and shortens release cycles (continuous delivery) [1]. These benefits, however, come at the cost of extended management processes for service deployment, orchestration, clustering, discovery, release and rollback. Kubernetes\(^1\) is one of the popular tools that addresses all the management issues effectively with a minimum user intervention. It only requires developers to specify their desired process configuration, the processes are then autonomously executed. To improve productivity, users can monitor and/or modify system configuration or data through Kubernetes cockpit, i.e. kubectl, at any time. This tool is a double-edged sword as it can potentially be used by a rogue insider to harm system assets. For instance, a malicious insider [2] can access clients’ sensitive data, or worse yet, downgrade system’s security configuration. To cope with such threats, kubectl is equipped with an audit trail mechanism to keep track of insiders’ actions. However, given the full control of insiders on the host, such logs can potentially be forged.

Goal
The goal of this thesis is to mitigate the risk of kubectl abuse by insiders without sacrificing productivity. That is, developers should still be able to freely monitor, modify and manage Kubernetes clusters, guaranteed that all their activities are recorded in a tamper-evident [3] storage. Such activity logs later enable analyzers to identify misbehaviors in case of a security compromise. In this thesis, the student will extend the kubectl’s native audit trail mechanism to guarantee its tamper-evident property. To do so, first two approaches, namely Blockchain-based [4] and secure-logging-based [5] are studied. Then, their applicability, advantages and disadvantages are analyzed. Based on the outcome of the analysis, the student implements, or adopts an open source implementation of, the technique that best suits kubectl. At last, the security as well as the performance of the implemented mechanism will be thoroughly evaluated.

\(^1\) www.kubernetes.io
Workplan

1. Compare cons and pros of Blockchain-based and secure-logging-based schemes
2. Implement/adopt the scheme that best suits the kubectl’s audit trail mechanism (written in go)
3. Evaluate the solution:
   a. Evaluate the security using attack defense trees
   b. Evaluate the performance of the mechanism including memory and runtime overhead analysis
4. Write the final thesis document
   a. Description of the problem and motivation
   b. Description of the theoretical background
   c. Implementation description
   d. Performance evaluation of the implementation
   e. Security evaluation
   f. Conclusions and future work

Deliverables

- Docker container able to run a demo of the implementation, including instructions on how to run the demo
- The container should also include the source code of the implementation
- Technical report with comprehensive documentation of the implementation, i.e. design decision, architecture description, API description and usage instructions
- Final thesis report written in conformance with TUM guidelines

References