A Real-Time, Flexible Logging and Monitoring Infrastructure for MonPoly

Jonas Degelo

Advisor:
François Hublet
Professor:
Prof. Dr. David Basin

Bachelor's Thesis

Information Security Group
Department of Computer Science
ETH Zürich
February 2023

1 Introduction

Our digital world consists of many large and small systems. Those systems are continuously logging vast amounts of data. In many use cases one might want to monitor those logs of data and check that those logs conform with a predefined specification. In case of a violation of such a specification or policy one might want to catch that violation and then trigger an appropriate action. Such monitoring is part of Runtime Verification (RV) [1].

One potential system is any service that handles user data. A user might be able to give and revoke permission to perform certain actions with their data. Like sharing it with advertisers or a different part of the same company. Or maybe there are data protection laws and regulations that restrict what can be done with user data.

MonPoly [3] is a tool for runtime Verification. It can do both online and offline monitoring of policies. For online monitoring it accepts new events via standard input. For offline monitoring it can read a timestamped log file that was generated during the runtime of a system.

This approach has some limitations. For one online monitoring can not easily be done on a different machine. Or imagine a system that has many different servers, all creating new events. Now we would want those events all to reach a single machine. The current version of MonPoly, working with standard input, is not optimized for such a use case.

Let's return to the example of user specified privacy preferences. It comes quite natural that the user might want to update their preferences. Currently MonPoly has no way of changing the policy it is monitoring.

1.1 Contributions

MonPoly works with timestamped, tabular data. The schema of the table is given to MonPoly as a signature file at launch. This makes the connection of MonPoly with a time series database quite natural. Many of the aforementioned shortcomings of MonPoly can be overcome by combining it with a separate database.

We extended MonPoly with a backend written in Python. The backend connects MonPoly to a time series database (QuestDB). It facilitates moving the monitoring from one machine to another and efficiently reload the previous state as long as the database is retained. We do this by making use of relative intervals and only loading events that are within the relative interval of a formula. Further the backend enables a first method for changing the monitored policy.

2 Background

2.1 Metric First-Order Temporal Logic

MonPoly uses Metric First-Order Temporal Logic (MFOTL) [**Chomicki1995**, 2] as a policy specification language. It can monitor a subset of possible MFOTL formulas. MFOTL is a

2.2 MonPoly

2.3 Time Series Databases

3 Architecture

4 Algorithms

5 Implementation and Evaluation

6 Conclusion

References

[1] Ezio Bartocci et al. *Lectures on Runtime Verification.* Ed. by Ezio Bartocci and Yliès Falcone. Vol. 10457. Springer International Publishing, 2018. ISBN: 978-3-319-75631-8. DOI: `10.1007/978-3-319-75632-5`. URL: `http://link.springer.com/10.1007/978-3-319-75632-5`.

[2] David Basin et al. "Monitoring Metric First-Order Temporal Properties". In: *Journal of the ACM (JACM)* 62 (2 2015), pp. 1–45. ISSN: 0004-5411.

[3] David A Basin, Felix Klaedtke, and Eugen Zalinescu. "The MonPoly Monitoring Tool." In: *RV-CuBES* 3 (2017), pp. 19–28.