# Review: Faceted Dynamic Information Flow via Control and Data Monads

Schmitz et al (2016) describe the processes involved in securing information through a faceted dynamic information flow. The authors, correctly, highlight the importance of potential data breaches. They describe the tension between working functionality and security. Specifically, they state that 'Developers primarily focus on correct functionality; security properties are prioritized only after an attempted exploit' (Schmitz et al, 2016). However, this assertion is not backed up with any evidence. Furthermore, the opinion is dated and there has been more emphasis on security since the article was written. However, it is also the case that the Agile Manifesto specifically states that working software is prioritised over comprehensive documentation and that this is an inherent risk to secure software development unless there are mitigating strategies, such as secure scrum, implemented effectively.

According to Schmitz et al (2016), reference secure multi-execution where a programme is repeatedly run with different privileges, allowing sensitive information to be access and written to private channels in one execution, while completing the operations on a second execution which can only access non-sensitive information but may write to public channels. This creates a separation between sensitive information and public channels.

The article then promotes the faceted evaluation technique that simulated the secure multi-execution method, but with a single execution of the programme. They offer a vast example of technical code that is, perhaps, beyond the scope of this module of work. The approach is to use lambda calculations which can be used for partitioned equivalence testing. As a result, this can determine the relevant permissions and associated view of the data.

The advantages of the faceted evaluation technique are that the values can be introduced as a library, rather than an extension of the coding language. Faceted evaluation also has an advantage regarding efficiency as the programme or routine is executed once. Finally, this approach can provide a more flexible approach to permission and access and poetically provide improvements when updating code.

## References

Agile Manifesto (2001), Manifesto for Agile Software Development. Available from: https://agilemanifesto.org [Accessed 21 October 2024]

Schmitz et al (2016), Faceted Dynamic Information Flow via Control and Data Monads. Available from: https://kennknowles.com/research/schmitz-rhodes-austin-knowles-flanagan.post.16.faceted.pdf [Accessed 21 October 2024]