# Evolution of security policies

Virginia N. L. Franqueira
University of Twente, Enschede, The Netherlands
Department of Computer Science, Information Systems Group
franqueirav@ewi.utwente.nl

## Abstract

Organizational security involves assuring data confidentiality, integrity and availability. These security principles have been captured by standards like ISO 17799 [1] which provides guidelines in the format of objectives to be achieved and controls to be implemented. Each organization interprets and selects applicable controls according to their culture, infrastructure and business to define its security policies. Cobit [2] provides orientations for enterprise security governance and considers control over information and technology (IT) as a core factor for the alignment between business objectives, IT goals and IT processes. In the format of control objectives Cobit aims to assure that i) "business objectives are achieved, undesired events are prevented or detected and corrected", ii) a measure of the security level and improvements required are in place. It is again up to individual organizations to define their overall security policies based on the Cobit best practices. However, security policies must be somehow enforced as these standards are at the paperwork level of security. This means that security policies have to be translated or refined to different domains either by means of human expertise or by means of tools. I assume that this refinement process is already in place and focus on the three key points of control defined by Cobit: i) business objectives enforced, ii) events monitored and iii) measures integrating objectives and events implemented. The IPID[1] project [3] aims to link these three factors by establishing a feed forward management loop and a feed backward compliance loop. The latter loop is the main focus of my research since it is absent in large organizations while, as prescribed by Cobit, it represents a key element of IT governance.

**This paper claims** that goal-driven requirements which are subject to continuous evolution when triggered by correlated security events (detected by several security devices) achieve the feed backward loop. In order to meet this claim I propose an approach consisting of the following ingredients.

1. A method to formalize security policies as goal-driven requirements. This formalism should also allow the formalization of events in a comparable way to facilitate the relationship between security policies and events.
2. A model of the policy evolution process.
3. A method to correlate events and extract information to be used as triggers to the evolution process.
4. A method to actually trigger the evolution process.

The first, third and fourth items of our approach are design problems. Thus, I will focus on the analysis of related work in the literature to either extract the requirements of the solution and identify opportunities to reuse existing approaches. After this initial stage, I will propose a framework which incorporates each of these modules as building blocks. The second item, however, is a knowledge problem and a case study will be used as an exploratory method [4] to provide insights about the state-of-affairs of the evolution process in the real world. I believe that currently, this process is not triggered by security events. Therefore, the model to be built is prescriptive rather than a descriptive model of the security policy evolution process.

[1] IPID stands for Integrated Policy-based Intrusion Detection.

## References

[1] BS 7799/ISO 17799 Information Security, 2000. http://www.bsi-global.com/Global/bs7799.xalter.

[2] CobiT: Control Objectives for Information and related Technology. http://www.isaca.org.

[3] IPID Proposal, 2005. http://wwwhome.cs.utwente.nl/~patveck/index.php?page=IPID.

[4] D. Pumareja, K. Sikkel, and R. Wieringa. Understanding the dynamics of requirements evolution: a comparative case study of groupware implementation. In *REFSQ'04: Proceedings of the 10th Workshop on Requirements Engineering: Foundation for Software Quality*, pages 177–194, London, June 2004. Springer.