Logic of Knowledge and Belief in the Design of an Integrity Kernel for an Office Information System

J. F. Peters, S. Ramanna, E. A. Unger
Department of Computing & Information Sciences
Kansas State University
Manhattan, KS 66506

Abstract

An integrity specification language (ISL) based upon a form of interval temporal logic is defined and the framework for a logic of knowledge and beliefs about data integrity is developed. The intended purpose of ISL is for the design of integrity kernels in an office information system in which the dynamic evaluation of data integrity based upon partial knowledge and informed judgement relative to update request by clients, is required. ISL itself is described in an earlier paper and summarized in this paper. Beliefs in ISL are specified with a form of interval temporal logic and provide an extension of the Moser technique for formulating beliefs. The Clark and Wilson model designed to prevent fraudulent and erroneous data modification is subsumed. An integrity Characteristic Tuple (ICT) incorporates the notions of correctness, completeness, quality, timeliness and confidence is associated with each Constrained Data Item (CDI). An integrity system which includes extensions of the concept and function of the server architecture as defined in the Multimedia Office Server (MULTOS) project, is given. An integrity kernel is defined which incorporates an algorithm for the detection of faults in presumptions about data integrity based upon knowledge and beliefs. A simplified integrity kernel is specified formally and a corresponding sample server-client session is provided.

AN ASSUMPTION ABOUT COMPUTER SECURITY
(It ain't necessarily so.)

Lee Ohringer*
Department of the Treasury
Bureau of Engraving and Printing
Assistant Manager
Automated Information Security Staff
Washington, DC 20228

It is often stated that "A chain is only as strong as its weakest link." However, this maxim doesn't hold true for Computer Security!

Do not despair if there are parts of your organiza-tion's computer security practices that defy your best efforts to improve them. You may be able to compensate for weaknesses in one part of your program by beefing-up other parts.

For example, software security is one of the links in the computer security chain. For some computers, this link is weak or almost nonexistent. But realize that the computer users are another link in the computer security chain. By strengthening the computer user link, which we can do by improving user awareness, we can more than compensate for weakness in the software security link. By doing so, we also buy time to develop appropriate software security.

So don't let the assumption that computer security is only as strong as its weakest link deter you from implementing good security, because it ain't necessarily so.

---

*The views expressed here are those of the author and do not necessarily represent any organization with which he is affiliated.