Journal of Computer Security 4 (1996) 229–231
IOS Press

Editor's preface to the Bell–LaPadula model

This issue includes volume II of the Bell–LaPadula model. The Bell–LaPadula model played, and continues to play, an important role in the development of multilevel computer security. It was called for as part of the DoD paradigm for implementing secure systems, as described in an Air Force planning study [1].

The Anderson Report recommended using a model of an "ideal system" as the starting point for the design, and David Bell and Leonard LaPadula of The MITRE Corporation were commissioned to write one.

Between them, Bell and LaPadula generated several versions of the model. Volumes I–III contained different, progressively more complex models. The later models reflected different design decisions; they were not refinements of earlier ones. There was also a "Unified Exposition and Multics Interpretation".

Volume I [2] took its inspiration from general systems theory rather than the theory of automata, though it adopted the access matrix idea from contemporary operating system design papers. It modelled a system as having subjects, objects, and a current state with three components: an access relation, indicating the existence of current access to an object by a subject; an access matrix, indicating the types of access (read, write, copy, append, owner, or control); and an assignment of classifications and need-to-know categories to subjects and objects. State transitions involved requests and decisions, which were not listed. In a secure state, any kind of access by a subject to an object required that the subject dominate the object in classification and need-to-know categories.

Volume II [3] limited the access types to read, write, append, execute, and control. It introduced a form of the *-property, with execute access viewed as a kind of read, and "write" access implying both read and write; append access was write-only. It had ten specific transition rules. The access relation now indicated the access types, and the access matrix became a permission matrix for discretionary access control. Rules for giving and rescinding discretionary access permissions tested control access.

Volume III [4] introduced an object hierarchy as a new component of the state, doing away with control access. A subject had the equivalent of control access, the ability to change an object's security attributes, if it had write access to the parent object. Subjects were given a "current" security level, distinct from, but dominated by their maximum security level, leading to a change in the way the *-property was stated.

The Multics Interpretation [5] revised the rules to be a better match for the Multics kernel primitives, and it added the discretionary security property as an axiom, which stated explicitly that current access must be consistent with the permission matrix.

Despite the fact that there were several versions of the model, it is usual to speak of "the Bell–LaPadula model" as though there were only one. The features that are called to mind by the name are (1) the basic subject–object-access model, with at least read and write access modes, (2) the structure and comparison of sensitivity levels, (3) the simple security property and some form of the *-property, and (4) a set of transition rules corresponding roughly to the operating system kernel calls that have an effect on the access state.

The Bell–LaPadula modelling style had considerable influence beyond its use for Secure Multics. The Trusted Computer System Evaluation Criteria [9] called for a formal policy model for high-assurance evaluation classes. The models produced to satisfy that requirement typically followed the Bell–LaPadula style. Successful as a prototype for these activities, these models nevertheless had limitations that gave rise to debates over modelling style.

One of the early debates concerned the tranquility principle, mentioned first in Volume II, saying that the classification of active objects would not be changed during normal operation. This was not stated formally as an axiom, or even as a theorem to be proved about the transition rules; it was informal design guidance for the rules. McLean published an example called "System Z" with one rule that downgraded all subjects and objects and granted all access [6], which violated the design guidance but not the formal axioms. This called attention to the fact that the security of a system based on the Bell–LaPadula model, and in particular its ability to protect information from compromise, depended to a large extent on the choice of transition rules.

Regardless of the choice of transition rules, there are general theoretical questions concerning how much one can prove about the security of a system given an abstract description of it. Access control models have undecidable questions such as the safety problem concerning propagation of discretionary access controls, and they do not address the deeper aspects of information flow. Some of this context is discussed in [7].

The Bell–LaPadula model was expressed using set-theoretic notation, in which a state transition is a relation, that is, a set of tuples, as is done in automata theory. Viewed as a mathematical model, it is complex and highly particularized. It is perhaps better understood as a high-level design specification for a family of related systems, than as a mathematical structure.

When it was written, there were no formal languages for system specification in common use; Parnas had only just introduced the idea of having such things [8]. Later on, as Bell–LaPadula-style models were written for TCSEC evaluation, they were expressed in various languages that were being developed for formal design environments.

One challenge we had to face in presenting the Bell–LaPadula model in this journal was the conversion from the original typed hardcopy to a machine-readable form. Leonard LaPadula retyped Volumes I and II, entering it into a word processor, and the result had then to be converted to TeX. There were questions about

whether to preserve the original appearance of the typed reports, which had underlining in place of italics, and full-sized superscripts and subscripts, or whether instead to apply standard typesetting conventions. A complete typesetting update would have forced some difficult and probably unsatisfactory decisions. A simple conversion of underlining to italics, for example, would have resulted in inconsistencies with the usual conventions. In the end, it was felt desirable to retain the overall appearance of the original, but not attempt to imitate the monospaced typewriter font. Underlining is still used, but subscripts and superscripts are reduced in size and italicized where appropriate.

This process has been completed so far only for Volume II, in this issue. In the future, we may also publish Volume I and others.

In current efforts to design secure systems, we are now faced with the engineering problem of providing suitable formal design models for networks and network components, using security policies that have more to do with privacy and authentication than with comparison of security levels. Perhaps one day a modelling style will emerge for these applications that is as useful and stimulating as the Bell–LaPadula model has been for secure operating systems.

References

[1] J.P. Anderson, *Computer Security Technology Planning Study*, Vol. 1, ESD-TR-73-51, AD 758 206, James P. Anderson and Co., Fort Washington, PA, 1972.

[2] D.E. Bell and L.J. LaPadula, *Secure Computer Systems: Mathematical Foundations*, ESD-TR-73-278, Vol. 1, The MITRE Corporation, Bedford, MA, 1973.

[3] L.J. LaPadula and D.E. Bell, *Secure Computer Systems: A Mathematical Model*, ESD-TR-73-278, Vol. 2, The MITRE Corporation, Bedford, MA, 1973.

[4] D.E. Bell, *Secure Computer Systems: A Refinement of the Mathematical Model*, ESD-TR-73-278, Vol. 3, The MITRE Corporation, Bedford, MA, 1973.

[5] D.E. Bell and L.J. LaPadula, *Secure Computer System: Unified Exposition and Multics Interpretation*, ESD-TR-73-306, The MITRE Corporation, Bedford, MA, 1975.

[6] J. McLean, Reasoning about security models, in: *Proc. 1987 IEEE Symp. on Security and Privacy*, pp. 121–131.

[7] J.K. Millen, Models of multilevel security, in: *Advances in Computers*, Vol. 29, M.C. Yovits, ed., Academic Press, 1989, pp. 1–45.

[8] D.L. Parnas, A technique for software module specification with examples, *Comm. ACM* **15**(5) (1972), 330–336.

[9] Department of Defense Trusted Computer System Evaluation Criteria, DOD 5200.28-STD, December 1985.

Jonathan Millen