BULGARIAN ACADEMY OF SCIENCES
БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ • BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, **62**
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, **62**

София • 2010 • Sofia

Information Security – Bell-La Padula Model[1]

Nina Dobrinkova

Institute of Information and Communication Technologies, 1113 Sofia
E-mail: ninabox2002@yahoo.com

1. Introduction

Before the Bell-La Padula model in the late 1960's, developments in commercial operating systems suggested the possibility of tremendous cost savings. Time-sharing was starting to provide commercial customers the ability to share the leasing costs of IBM and other big-iron computers through simultaneous or sequential use of the expensive mainframe computers. For those in classified government circles, this new capability promised even more savings. Before time-sharing, separate computers had to be used for each different security level which was processed on computers, or careful "color changes" had to be made so that the same equipment could be used sequentially to process information at different security levels (referred to as "periods processing"). There was therefore the possibility of sharing those computer systems across security levels, with an important proviso. It was crucial that processing artifacts of each security level (files, registers, data) be kept rigorously separate with a high degree of confidence. During this period the so called "tiger teams" groups of experts were assembled and were working on these computers [1]. During those days hackers found ways to enter into the high level security systems and they were opening back doors that would persist through system and compiler recompilations, which faced the computer security world to the need of better security model [2-5].

2. Bell-La Padula model

In the summer of 1972, MITRE Corporation initiated its task to produce a report entitled "Secure Computer Systems." The report was to describe a "mathematical model of security in computer systems." This task was one of several in an overall security project, mostly engineering tasks. The modeling task fell to Len La Padula and David Elliot Bell.

[1] This research paper has been supported by project No DVU01/0430 funded by the National Fund for Scientific Research.

Their initial work focused on a definition of “security” within a mathematical (conceptual) framework. At the beginning, they viewed the access monolithically, rather like possession of a book from a lending library. Having just completed abstract investigations into data sharing and appreciative of the complexities of “deadly embrace,” they began to think about the complexities that would result when an “object” (the generalization of all things that could hold information) had its security level changed. Would the system immediately change the security level? Would “subjects” currently accessing the object have their access terminated? Wouldn’t sudden termination of access imply that the classification had been increased? A mole in the organization could cache copies of a wide array of documents, determine when something has been upgraded, and pass copies along. On the other hand, if one took the tack of delaying the upgrade of the classification until all users had voluntarily given up access, there will be the possibility of indefinite delay: user A will have access; user B requests and gets access; user A gives up access; user C requests and gets access; user B gives up access; user A requests and gets access. In the midst of these considerations they started a project meeting the need of object’s security classification, but this time dynamically. They defined for this reason a basic security theorem, which purpose was to evaluate in the end of constructing their model [6].

Basic Security Theorem. Let $W \subset R \times D \times V \times V$ be any relation such that

$$(R_i, D_j, (b^*, M^*, F^*), (b, M, f) \in W$$

implies

(i) $f = f^*$,

(ii) every $(s, O) \in b^* - b$ satisfies SC rel f^*.

$$\sum (R, D, W, z) \text{ is a secure system for any secure state } z.$$

Their first theorem was just the start of the finer work that they had to perform. They faced the problem with * -property, the *W* relation was not conceptualized and the changes of the state were not well defined, that is why this first theorem was not sufficient for the analysis and formulation of core system calls that change the security state. Graphically the *-property can be described as increase of the security level [7].

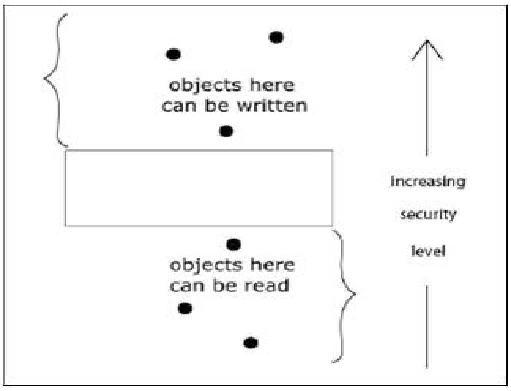

Fig. 1. Graphical description of the *-property

After refinement of the first theorem rules with changes in *W* relations, better structure of the access modes in general, La Padula and Bell came up to a new "rule" structure: *W* in the first volume of their research was an undefined relation that conceptualized allowable changes of state. In the first volume, a relation with a structure was not required in order to establish that volume-I security was inductive. If the model were to provide direct assistance in the formulation of core system calls, a more constructive form has to be developed. The approach adopted was to specify *W* as a function of a set of "rules," each addressing a particular change of state. The conditions limiting the assembly of rules ω to define a complete relation $W(\omega)$ were minimal: each had to address a well-formed class of request and no two rules could have responsibility for the same request for change of state. Within that structure, a set of rules governing every possible change of state was devised: getting access to objects (in the four access modes, read, append, execute, and write); releasing access; giving access to another subject; rescinding other subjects' accesses; changing security levels; creating objects; and deleting them. Each of ten rules was then proved to preserve simple-security (that is, the subject's classification is greater than the object's classification); discretionary-security (that is, the subject is listed as having access to the object); and *-property. Any combination of non-overlapping rules resulted in a conceptual system that stayed in secure states if it began in one.[8] After setting these rules a second better version of the **Basic Security Theorem** has been written:

Let $\omega = \{\rho_1, \rho_2, \ldots, \rho_{10}\}, \rho_i$

as defined in the new rules, and z_0 be a secure state which satisfies *-property. Then $\Sigma\,(R, D, W(\omega), z_0)$ is a secure system which satisfies *-property.

Refinement of the mathematical model. After the second better version of the basic security theorem has been written, Bell started refinements of the mathematical model including object hierarchy principle to make easier the engineers task to create secure prototype of the model.

Object hierarchy. The first refinement addressed control of access privileges, based on the object hierarchy within the operating system. For Multics (and by inheritance, for Unix), control of objects (segments in Multics, directories and files in Unix) is limited by access to the object's parent directory. The engineers building secure Unix prototypes and participating in securing Multics wanted and needed some conceptual guidance on the security implications of this variety of control. To deal with this diffuse, implicit control, the model had to be extended to include an object hierarchy *H* that indicates the structure of objects in any state. The collection of rules for building and analyzing systems was augmented by the addition of four alternative control rules: rules to give and rescind access in a system with implicit hierarchical control; and rules to create and delete objects, with and without "compatibility." The term "compatibility" was used for systems where the security levels are monotonically non-decreasing from the root directory down each pathname. That is, each object has a security level the same as or greater than that of its parent [9].

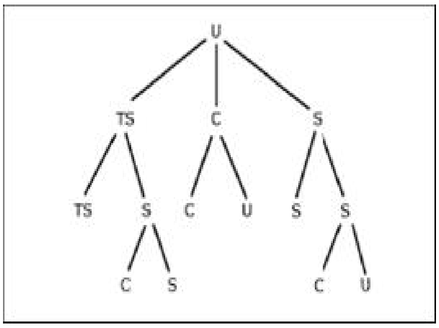

Fig. 2. Anti-Compatibility

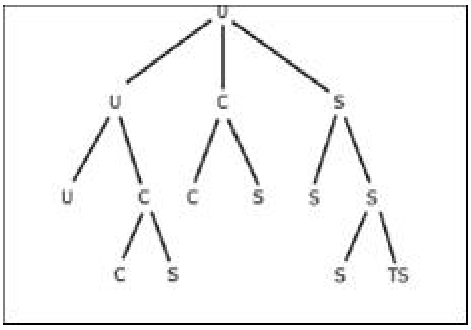

Fig. 3. Compatibility

After adding the object hierarchy to Bell-La Padula model has had a second refinement: **Simplification of *-property**. The second refinement simplified the *-property based on an engineering short cut. Lee Schiller noted that the original formulation of *-property required comparing the security level of a newly requested object to that of every object to which the subject currently has access. His simplification was to record each subject's "current security level" in a system variable and to compare the new object's security level to the current security-level, replacing many comparisons with a single one. For viewing accesses, the current security level had to be greater or equal to (or to "dominate") that of the new object. For altering accesses, the current security level had to be less than or equal to (or be dominated by) that of the new object. For access that included both view and alter, the two security levels had to be the same. This engineering short cut not only simplified the statement of the *-property but also narrowed the gap between practical implementations and modeling versions. This refinement required the addition of the current security level to the modeling definitions, then recasting rules ρ_1, ρ_2, and ρ_4 (get-read, getappend, and get-write) in terms of current security level.

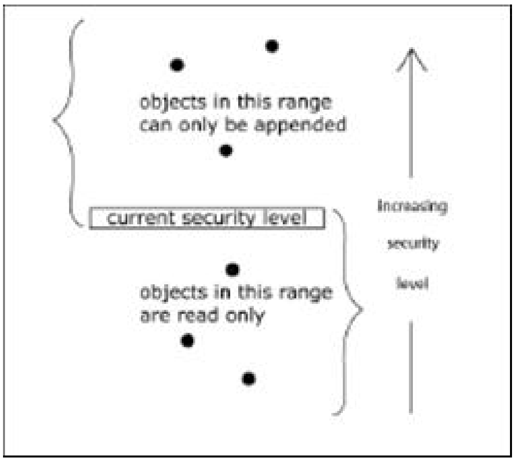
objects in this range can only be appended
current security level
objects in this range are read only
Increasing security level

Fig. 4. Revised *-property

Untrustworthy and trusted subjects. The third refinement was a revision of the *-property that introduced the concept of the "trusted subject." The stimulus for this revision came from the engineering task focused on building a secure operating system prototype. The intention was to apply the *-property to every process within the prototype. Consider the scheduler. Its function was to swap jobs when conditions require. If the current process were running at TOP SECRET, then the scheduler would have to read all the current state information and write it into swap space. If the next process were UNCLASSIFIED, the scheduler would similarly have to read all swapped out information, move it into place for execution, then kick off the new process. On reflection, the premise of the original *-property was that no subject could be trusted not to copy part of the intelligence summary into the bowling scores. The model was altered to identify a subset S'' of the full set S of all subjects. These subjects were the subjects that “are untrustworthy and may mix information as described” [10]. The concept of “*-property” was replaced by “*-property relative to S' ” and the subjects outside S (mathematically, the set $S - S'$) were not subjected to the restrictions under the original *-property. Subjects as a whole were divided into two parts, “untrusted subjects” and “trusted subjects.” Those “trusted subjects” were precisely those who “will never mix information of different security levels”.

For third time **Basic Security Theorem** has been rewritten:

If one desires a secure computer system which exhibits only compatible states satisfying the*-property, one can use the set of rules

$$w'_{iii} = \{\rho_1, \rho_2 , \rho_3, \rho_4, \rho_5, \rho_{12}, \rho_{13}, \rho_{15}, \rho_{16}, \rho_{17}\}$$

together with a compatible, secure initial state z_0 which satisfies the *-property [relative to S'].

Fourth refinement has been done after a year and new “rules” modified the theorem once again.

Final theorem. Let ρ be a rule and $\rho(R_k, v) = (D_m, v^*)$, where $v = (b, M, f, H)$ and $v^* = (b^*, M^*, f^*, H^*)$:

1) if $b^* \subseteq b$ and $f^* = f$, then ρ is ss-property-preserving;

2) if $b^* \subseteq b$ and $f^* = f$, then ρ is *-property-preserving;

3) if $b^* \subseteq b$ and $M^*_{ij} \supseteq M_{ij}$, then ρ is ds-property-preserving;

4) if $b^* \subseteq b$, $f^* = f$, and $M^*_{ij} \supseteq M_{ij}$, then ρ is secure-state-preserving.

3. Consolidation

After those final four refinements of the Bell La Padula model the engineers tried to put into practice all developed rules creating secure computer system following the model. Engineering and conceptual efforts continued from 1975 through 1985. At the beginning of the 1980’s, there was a concentrated effort to solidify and build on the successes of the 1970’s work in computer security and to learn from the failures. The various engineering tasks, conceptual tasks and research prototypes had identified security principles, methods and techniques and proved those techniques in exemplar systems such as the Kernelized Secure Operating System (KSOS) [11], the Provably Secure Operating System (PSOS) [12], the Kernelized Virtual Machine (KVM) [13], and Multics [14]. Steve Walker at the Office of the Secretary of Defense initiated the Computer Security Initiative with three goals: to formulate a metric for measuring the security of systems, to establish a center whose mission was to measure commercial systems against that metric, and to initiate a technical conference dedicated to computer security. The metric was the Trusted Computer System Evaluation Criteria [15], published in 1985. The center was the Department of Defense Computer Security Evaluation Center (later the National Computer Security Center). The conference was the Department of Defense/National Bureau of Standards Computer Security Conference (later the National Computer Security Conference).

In the 1990's the technology movement away from single logical platforms well understood in the computer security world posed a dilemma for the Computer Security Center with regard to the TCSEC and the product evaluation program. A series of initiatives focused on the special security issues, first in networks and later in database management systems. The results were the Trusted Network Interpretation (TNI) of the Trusted Computer System Evaluation Critieria and the Trusted Database Interpretation (TDI) of the Trusted Computer System Evaluation Critieria. Absent clear promulgated guidance on networking issues, product evaluation faced a hard choice: exclude networking or include networking without clear guidance. Neither choice was good. Caution led to the exclusion of networking features in the early days. Part of the consolidation was the inclusion of a requirement for security policy modeling for higher security products in the TCSEC. In addition, the TCSEC listed the Bell-La Padula model as a good representative of a model with the characteristics required. The Bell La Padula model has been modified when a conception for the hosts and connections were needed (this modification was called liaison in the model). This model was the basis for many inventions in the nowadays security systems and security networking, which makes it even more valuable, than at the beginning of the 1970's, when it was written [16, 17].

Our computer security legacy at the beginning of the 21-st century is not extensive, but neither is it inconsiderable. It is the result of dedicated work by master technologists and government champions. Lasting principles were hammered out conceptually, then refined and honed on engineering workbenches. The lessons were codified and an immense effort to prime the pump with secure, trusted systems was undertaken, compensating for the inefficiency of market forces. In the opening years of this century our legacy includes two very high security products and the community knowledge to deploy them wisely. We are in a much better situation now than in the early 1980's, but the knowledge has to be expanded in parallel with the needs of the nowadays society and hardware availability.

References

1. Anderson, J. P. Computer Security Technology Planning Study. – ESD-TR-73-51, Vol. **I**, AD-758 206, ESD/AFSC, Hanscom AFB, MA, October 1972, p. 4.
2. Karger, P., A. Roger, R. Schell. Multics Security Evaluation: Vulnerability Analysis. – ESD-TR-74–193, Vol. **II**, ESD/AFSC, Hanscom AFB, MA, June 1974.
3. Thompson, K. Reflections on Trusting Trust. – Comm. ACM, Vol. **27**, August 1984, No 8, 761-763.
4. Thompson, K. On Trusting Trust. – Unix Review, Vol. **7**, November 1989, No 11, 70-74.
5. Anderson, J. P. Computer Security Technology Planning Study. – ESD-TR-73-51, Vol. **II**, ESD/AFSC, Hanscom AFB, MA, October 1972, p. 16.

6. Elliot, Bell, D., L. J. La Padula. Secure Computer Systems: Mathematical Foundations. – MTR-2547, Vol. **I**, MITRE Corporation, Bedford, MA, 1 March 1973. ESD-TR-73-278-I.
7. Saltzer, Jerome H., M. D. Schroeder. The Protection of Information in Computer Systems. – In: Proc. of IEEE, 63(9), September 1975, 1278-1308.
8. La Padula, L. J., D. Elliot Bell. Secure Computer Systems: A Mathematical Model. MTR-2547, Vol. **II**, MITRE Corporation, Bedford, MA, 31 May 1973. ESD-TR-73-278-II.
9. Walter, K. G. et al. Primitive Models for Computer Security. ESD-TR-74-117, Electronic Systems Division, Hanscom AFB, MA, January, 1974.
10. Elliot, B. D. Secure Computer Systems: A Refinement of the Mathematical Model. MTR-2547, Vol. **III**, MITRE Corporation, Bedford, MA, December 1973. ESD-TR-73-278-III, p. 25.
11. McCauley, E. J., P. J. Drongowski. KSOS – The Design of a Secure Operating System. – In: Proc. AFIPS 1979 NCC, Vol. **48**, 345-353.
12. Neumann, P. G., R. S. Boyer, R. J. Feiertag, K. N. Levitt, L. Robinson. A Provably Secure Operating System: The System, Its Applications, and Proofs. Technical Report CSL-116, SRI International, 1980.
13. Schaefer, M., R. R. Linde et al. Program Confinement in KVM/370. – In: Proc. ACM National Conference, Seattle, October 1977.
14. Saltzer, J. Protection and the Control of Information in Multics. – Comm. ACM 17(7), July 1974, 388-402.
15. Department of Defense Trusted Computer System Evaluation Criteria, DoD 5200.28-STD, December 1985.
16. Irvine, C. E. Considering Lifecycle Subversion. Invited Presentation. – In: MLS Workshop, Alexandria, VA, 24 September, 2003.
17. Anderson, E. A., C. E. Irvine, R. R. Schell. Subversion as a Threat in Information Warfare. – Journal of Information Warefare, **3**, June 2004, No 2, 52-65.

Информационная безопасность – модель Bell-La Padula

Нина Добринкова

Институт информационных и коммуникационных технологий, 1113 София
E-mail: ninabox2002@yahoo.com

(Резюме)

Модель информационной безопасности Bell-La Padula является базой многих концептуальных инструментов для анализа и проектирования надежных компьютерных систем сегодня. Его принципы безопасности доказывают свое эффективное применение в разных компьютерных и сетевых средах.