SOME CONUNDRUMS CONCERNING SEPARATION OF DUTY[1]

Dr. Michael J. Nash and Dr. Keith R. Poland

GAMMA SECURE SYSTEMS LIMITED

Diamond House, 149 Frimley Road, Camberley, Surrey, GU15 2PS, United Kingdom

ABSTRACT

This paper examines some questions concerning commercial computer security integrity policies. We give an example of a dynamic separation of duty policy which cannot be implemented by TCSEC based mechanisms alone, yet occurs in the real commercial world, and can be implemented efficiently in practice. We examine and describe a commercial computer security product in wide use for ensuring the integrity of financial transactions, show that it implements a well defined and sensible integrity policy that includes separation of duty, yet fails to meet either the TCSEC criteria or the Clark and Wilson rules.

INTRODUCTION

Clark and Wilson assert in their seminal paper distinguishing commercial and military security [4] that **"... a lattice model is not sufficient to characterize** (commercial) **integrity policies, and distinct mechanisms are needed."** This is a strong statement, with major implications to the developers of security products attempting to satisfy the needs of both military and commercial users. It is not universally accepted: within a few months of first publication, a number of papers were presented at the first Workshop on Integrity Policy in Computer Information Systems (WIPCIS) [9] that appeared to show that lattice based mechanisms, properly applied, could implement the kinds of commercial integrity policies identified by Clark and Wilson [10, 13].

In this paper we give an example of a dynamic separation of duty policy that cannot be implemented by lattice based mechanisms alone, yet corresponds to sensible real world needs, and can be implemented efficiently.

Furthermore, there are real world commercial integrity requirements that do not match the Clark and Wilson model. We give the example of the RGL250, a real commercial security product which is accepted in the financial world as implementing security with a high level of assurance, yet fails not only to meet the TCSEC criteria for Class C1, but also to obey some Clark and Wilson rules.

SEPARATION OF DUTY

In the portion of [4] headed "**Commercial Security Policy For Integrity**" Clark and Wilson identify two mechanisms at the heart of commercial fraud and error control: the well-formed transaction, and separation of duty amongst employees. Separation of duty attempts to ensure the correspondence between data objects within a system and the real world objects they represent. This correspondence cannot normally be verified directly. Rather, the correspondence is ensured indirectly by separating all operations into several subparts and requiring that each subpart be executed by a different person. We accept this definition, and it matches our own observations of both automated and manual commercial systems with integrity requirements. However, Clark and Wilson then state in the same section of their paper when discussing how these rules could be implemented in computer systems that "**to ensure separation of duties, each user must be permitted to use only certain sets of programs**

[1] PUBLISHED AT THE IEEE SYMPOSIUM ON RESEARCH IN SECURITY AND PRIVACY, 7-9 MAY 1990, OAKLAND, CALIFORNIA. (pp. 201-209) © 1990 IEEE.

(transactions)". We shall refer to this approach as static separation of duties (as do Clark and Wilson in [5]).

In common with Clark and Wilson, as expressed in their later draft paper [5], we do not believe this is sufficiently flexible to satisfy the practical requirements for enforcement of separation of duty found in many commercial organizations. We propose a less restrictive rule that we believe is appropriate in many circumstances. Under this rule, a user can be permitted to execute a transaction on a particular data item if:

a) The user is authorized to perform the type of transaction in question on that data item.

b) The user has executed no other transaction upon that data item.

We shall refer to this as object based separation of duty. This is an example of dynamic separation of duty as defined in [5].

AN EXAMPLE

To illustrate the differences between these two ways of implementing separation of duty, consider the example of a purchasing department and the processing of invoices for goods received.

Let us assume that there are three types of transaction involved in processing an invoice:

a) Recording the arrival of an invoice and the information recorded on it;

b) Verifying that the goods in question have been received and that the details of price etc. match those agreed with the supplier;

c) Authorizing the payment of the supplier.

We can also postulate three distinct types of user:

a) Data entry clerks, who enter the information on invoices into the system;

b) Purchasing officers, who verify the validity of invoices;

c) Supervisors, who authorize payments.

If we allocate fixed users to fixed types of transaction, then data entry clerks, and only data entry clerks, can record the arrival of invoices; purchasing officers, and only purchasing officers, can perform verification transactions; and supervisors, and only supervisors, can authorize payments.

We can express this static separation of duties using the Transaction Control Expression notation developed by Sandhu [12] as:

enter ■ clerk;
verify ■ officer;
authorize ■ supervisor;

This is not what usually happens in the real world.

Although some roles are restricted to certain transactions (for example, a data entry clerk is never permitted to authorize payments), it would be normal for an otherwise unoccupied purchasing officer to enter details of invoices if there is a backlog of unprocessed incoming paper, or for a supervisor to handle the verification of the details of a particularly complex invoice. What *never* happens is that the *same* purchasing officer verifies an invoice that he entered into the system, or the *same* supervisor who verifies an invoice is permitted to authorize payment against it. Indeed, when dealing with an unfamiliar commercial organization as a customer, we would regard as exceedingly unhelpful a supervisor who refused to answer an inquiry on the grounds that he was a supervisor and company security rules prohibited supervisors from performing clerical tasks under any circumstances.

We can express this example of object based separation of duties quite elegantly using the voting syntax of Sandhu's Transaction Control Expression notation:

1: enter ■ clerk = 1, officer = 1, supervisor = 1;
1: verify ■ officer = 1, supervisor = 1;
authorize ■ supervisor;

Rather than reproduce a large portion of Sandhu's paper in order to show that this expression indeed reflects the intent of our example as expressed in English, we refer the reader to [12] for a description of the notation. It is, however, interesting to note that Sandhu appears not to have considered the possibility of this type of overlapping assignment of roles to transactions whilst developing the notation.

RELATIONSHIP TO PUBLISHED WORK

A number of papers have considered the problem of implementing commercial integrity using TCSEC based mechanisms. The thrust of these papers is perhaps best expressed by Theodore Lee in [10]. Lee claims that, in addition to the principles and mechanisms of the TCSEC [6], all that is required to enforce the kind of commercial security policy described by Clark and Wilson are two concepts, not explicitly stated in the TCSEC but logical extensions of it, namely:

a) Mandatory integrity categories (as defined in [2])

b) The use of partially trusted subjects (as described in [1]).

We will start by considering Lee's own proposal [10]. In this paper Lee puts forward eight steps to be used to configure and operate a computer system containing a TCSEC based TCB so as to enforce the Clark and Wilson rules. Using this procedure results in transactions having an integrity label defined as the set of user roles which can execute that transaction. These labels are by their nature static and cannot cope with dynamic separation of duty. In the example we have given above, the integrity categories generated by step P1 of Lee's procedure would be clerk, officer and supervisor. Step P4 would cause all three of these roles to be classed as conflicting. In step P6 we would have to certify all three roles as permitted to perform the enter transaction. However, having done this, an officer would be authorized to execute both the enter and verify transactions on the same invoice, and there would be no means of preventing him from doing so.

Lee's approach fails to implement Clark and Wilson's intent by requiring authorization checks on a user role rather than individual user basis. Using Lee's approach it is necessary to prohibit officers from entering data. In reality, adequate separation of duty would be maintained as long as verification is performed by a different officer or supervisor. This same defect is also found in Terry and Wiseman's interpretation of Clark and Wilson [15].

In contrast to Lee, and Terry and Wiseman, Shockley in [13] and [14] deliberately implements dynamic separation of duties (see [14] Section 1.1). His implementation requires that checks for conflict using Clark and Wilson access control triples are made as part of subject creation. In the final version of the paper, as opposed to in [13], the potential for conflicts to be created after the subject exists is recognized and handled by only permitting subjects to have the lifetime of a single transaction. The TCB must perform additional controls at the start of each transaction before creating the subject, based on the stated user identity and data objects in question. It would be possible for such a mechanism to implement object based separation of duties as we have described it. However, the checking procedure would be non-trivial and to the commercial auditor it would be the primary security control. It would clearly be additional to the ordinary lattice based mechanisms and thus a "**distinct mechanism needed to provide integrity**" in the Clark and Wilson sense.

Jueneman in his paper on integrity controls [7] gives a different example of why a particular user could need to be prohibited from executing a particular transaction on a particular data object, namely that the data in question referred to that user. Jueneman observes that such conflicts would normally be rather complex, since they depend on an indirect and non-explicit link between user and data item. Jueneman proposes that the TCB should not be responsible for detecting such links. Certainly it would be very difficult for a standardized product to predict such data content dependent conflicts. He suggests instead that the TCB should provide a mechanism for transactions to establish the invoking user identity and implement whatever controls are necessary. Unfortunately, using the strict definition of TCB as given in the TCSEC [6], such transactions would then become part of the TCB and so liable to evaluation. A defense of Jueneman's approach could be that data content dependent conflicts should be very rare, with the mechanism in question only brought into use in occasional circumstances, and thus it could be ignored in an assessment of the main protection mechanisms.

Jueneman's mechanism could implement object based separation of duties as we have defined it. Indeed, we have seen such an approach successfully implemented in a real commercial system. However, when used for this purpose it would become a primary control, clearly an example of an additional mechanism required to implement the integrity policy.

THE CONSEQUENCES OF DYNAMIC SEPARATION OF DUTY

Within [4], Clark and Wilson present a framework for ensuring integrity based on nine rules. Within their model it is Rule E2 that implements separation of duty. E2 states that "**the system must maintain a list of relations of the form: (UserID, TPi, (CDIa, CDIb, CDIc, …)), which relates a user, a TP** (transaction), **and the data objects** (CDIs) **that TP may reference on behalf of that user. It must ensure that only executions described in one of the relations are performed**". A static interpretation of separation of duty implies that there will be a fixed relationship between UserIDs and TPs, and thus E2 can be simplified by the substitution of a concept of job function, rather than maintaining lists of relations specifying individual users (see [9], Chapter 7 and Chapter 4 Paragraph 3.1.2.3).

However, if an object based or other dynamic view of separation of duty is taken, then the full Rule E2 as stated above is required, since there will be some specific combinations of individual users, TPs and referenced data objects that must be excluded. It is our belief that the failure to identify dynamic separation of duty as defined in [5] within [4] led to confusion over why there was a need for the precise wording of Clark and Wilson Rule E2 as it is expressed in [4], and thus the consequent and erroneous belief that a number of papers have shown conclusively that TCSEC mechanisms can implement Clark and Wilson completely.

IMPLEMENTATION CONSIDERATIONS

In this section of the paper we propose a practical way to implement object based separation of duty.

There are other ways this could be done, such as maintaining a complete and dynamically updated set of relations in accordance with Clark and Wilson Rule E2. Alternatively the token capabilities plus access control list approach described by Karger [8] could be used.

We propose what we believe to be in many circumstances a simpler and more efficient solution. We require that every user has an authorized transaction list, and every data object is given a security label when created, and that this label contains sufficient space to record the UserIDs of those users that execute transactions upon it. The simplest way to do this would be to reserve space in the label for a UserID for each possible transaction type in the system. This will normally be practical because in real commercial systems the total number of transaction types is reasonably small. We also need two special UserID markers, namely *transaction type not permitted,* and *transaction type not yet executed.* At the time of object creation we can fill in each transaction type field with one of these two markers, as appropriate. As each transaction type is successfully executed, we replace the *not yet executed* marker with the appropriate UserID.

Our access mediation control then becomes merely checking that the user is permitted to perform the type of transaction in question (done thru examination of the authorized transaction list associated with the user), that the transaction type in question is permitted to act upon the data object, and that the user's UserID does not appear against any other transaction type.

This approach has a number of useful characteristics. It is possible to ensure that transactions are executed only in a legitimate sequence by examination of the object label. When initiating a transaction it is possible to exclude other transaction types which would be alternatives by modifying the label to replace *not yet executed* entries by *not permitted* entries. If data is modified in a way that invalidates certain previous transactions, we can force their re-execution by replacing the relevant UserIDs by *not yet executed* markers.

There is one other important implementation detail. The description above implies successful transactions cannot be repeated unless data is altered. However, in the real world it is not uncommon for there to be types of transaction where at certain times of the day (such as lunch breaks) only one person will be present who is authorized to perform those transactions. It is also possible that objects may exist where that user has already executed other transactions upon those objects. In the commercial world it will often be necessary to continue processing these objects without waiting for another authorized member of staff to return: if this is not permitted by the security rules, then those rules will be broken (for example, by "borrowing" passwords). A simple solution to this problem is to permit the original transaction to be rerun under the control of a different user who is authorized to perform the original transaction type. No data is changed, but the second user's UserID can legitimately replace that of the original user in the label entry. This then permits the original user to perform the transaction for which only he is authorized. This can be considered analogous to crossing out a signature on a document and re-signing by a different person, where the second signatory takes over the responsibilities implied by signature. The original signatory is then free to act in a different capacity.

In practice, we have found this implementation approach to be very effective. It is simple to implement, and has low overheads, both in the amount of secure information that must be maintained, and in the efficiency of performing access control mediation. It does require a clear understanding of the relationships between transactions, in order to ensure that no transaction is permitted by object label information from a valid but unusual state to execute, but then produces what is an invalid state.

A REAL COMMERCIAL SECURITY PRODUCT

In the previous section we proposed an efficient way to implement a large class of the dynamic separation of duty policies identified by Clark and Wilson. In this part of our paper we describe a real computer security product - the Racal-Guardata RGL250 - which is accepted in the commercial world as implementing security with a high level of assurance, yet not only fails to provide mechanisms required by the TCSEC C1 class, but also fails to meet certain of the rules proposed by Clark and Wilson in their model of commercial integrity. We hope that a description of this product will remind our fellow researchers of the dangers of assuming that real world needs always match our academic beliefs about those needs.

The RGL250 is a compact, battery-powered hand-held sealed unit that is used to generate Message Authentication Codes (MACs) to be attached to financial instructions passed between branches of the same financial institution, or between financial institutions. The MAC guarantees the authenticity and integrity of the instruction to the recipient and replaces the code books and test keys that are conventionally used for this purpose.

Figure 1 is an illustration of the RGL250. In use, a clerk enters details of the instruction into the device. These details can be viewed on the miniature screen built into the device and confirmed as correct by two authorizing officers from disjoint predefined groups. The device will then calculate and display an eight digit MAC which can be added to the instruction as if it were an authorizing signature. The instruction can then be transmitted by any agreed method - including telex, fax, electronic data network or voice telephone - to the intended recipient. A secure key distribution system is needed to ensure that the recipient can independently generate the MAC and thus confirm the authenticity of the claimed origin of the message, and that none of the significant data of the message differs from that which was entered into the sender's RGL250 for authorization.

For the purposes of this analysis, the device performs three types of transaction:

a) Data entry

b) Authorization

c) Managerial.

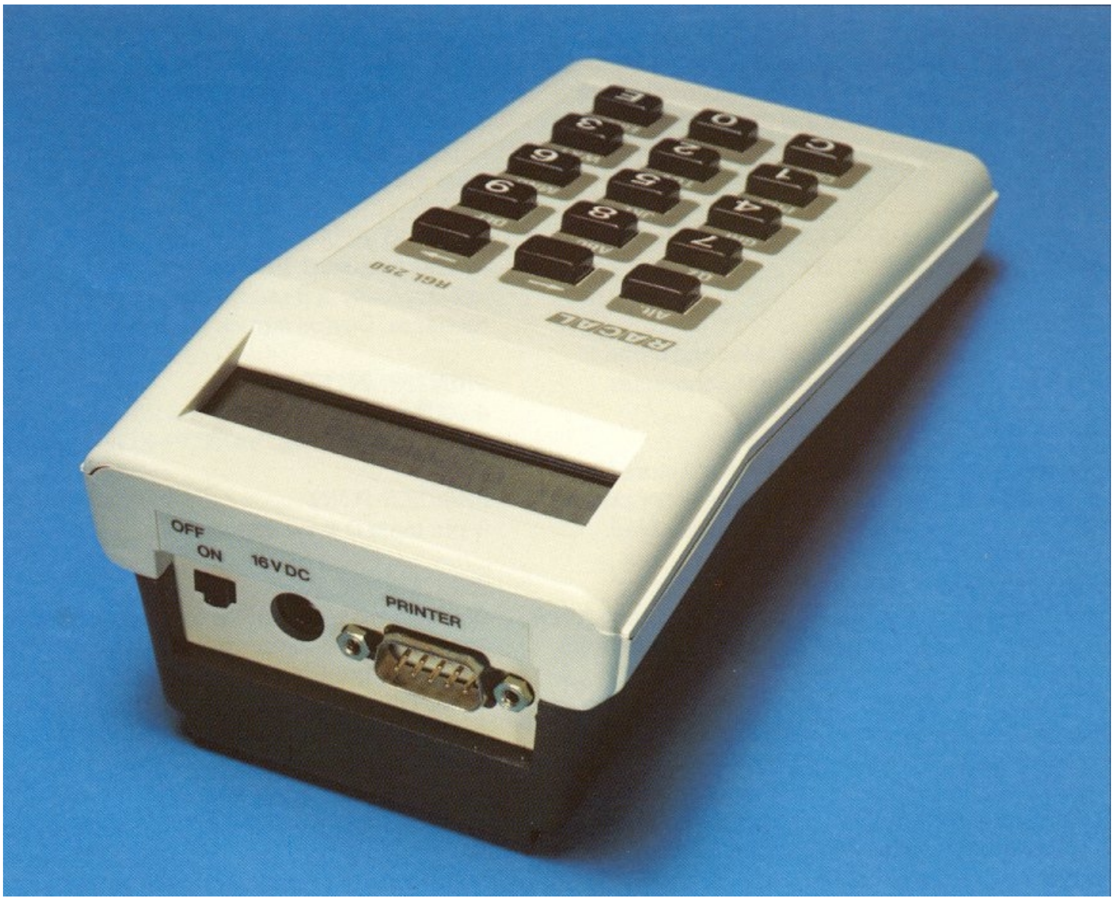

Figure 1 -The RGL250 Personal Authenticator

There are other types of transaction; for example, to verify a MAC on an incoming message from another branch or institution, but these will be ignored for the purposes of this description. The device can store the details of approximately 50 typical financial instructions in its internal memory.

Data entry, the routine entry of the details of instructions, is a straightforward procedure, normally performed by clerical staff.

At the end of data entry, or at the end of a batch of entries, two authorizing officers must then approve the payment instructions before the device will generate the MAC.

The choice of authorizing officers is a daily management decision. In total, the device will recognize up to twelve officers permitted to perform authorizing functions, storing a name and an encrypted password for each one. The officers are divided into two teams, called Green and Yellow. These two team names are built into the product and cannot be changed, a curious restriction.

Each day as a managerial function at least one but not more than two authorizing officers from each team must be activated. This is intended to impede collusion amongst the authorizing officers. There are other managerial functions such as changing passwords and entering new cryptographic keys when necessary. Only two officers may be nominated as managers capable of performing managerial functions; these officers may but need not also be authorizing officers.

The RGL250 stores an audit trail in its internal memory. The information recorded includes full details of each instruction and the names of the authorizing officers. It also records start and end of day operations. The audit trail can be displayed on the device screen, and can also be transmitted to an external printer.

Authorizing officers and managers must identify and authenticate themselves thru a logon procedure by specifying a password and (optionally) the use of a PIN-based personal identification challenge-response device. This procedure is well thought out; for example, passwords have a minimum and maximum length and are required to contain at least one alpha and at least one numeric character. Identification and authentication is not required before performing clerical transactions.

Each instruction or batch of instructions must be authorized by an activated officer from the Green team, and an activated officer from the Yellow team. These authorizations can be performed in any order.

Thus any instruction entered into the device will be in one of four states: unauthorized, authorized by Green only, authorized by Yellow only, or fully authorized.

IMPLICATIONS OF THE RGL250

This product poses several problems for the computer security community. Several thousand of these units are in use and it has been widely accepted by the internal and external auditors of commercial organizations as providing acceptable high assurance controls for the integrity of financial messages transmitted between locations. Thus either we must accept that the product is fit for purpose and our criteria for confidence are inadequate, or we must claim that use of the product is misguided.

For example, within this product there is no identification and authentication process for clerical transactions such as data entry, and thus clerks never need to enter their names or a password. This reflects normal rules for accountability for this type of work found within such organizations. Responsibility for the accuracy of clerical transactions rests with the officers of that organization or department. Officers can and will be held individually accountable for their authorization transactions, including checking the accuracy of the preceding clerical transactions, and thus the product needs and offers a strong password and challenge-response token based mechanism to authenticate authorizing officers.

Nevertheless, the product clearly fails the TCSEC criteria, even at Division C Class C1, since the criteria for this class state that "**the TCB shall use a protected mechanism (eg. passwords) to authenticate the user's identity**" and user is defined in the Glossary as "**any person who interacts directly with the computer system**".

Similarly, the product fails to meet the equivalent Clark and Wilson rule, Rule E3 - "**the system must authenticate the identity of each user attempting to execute a TP**" - because there is no identification and authentication required to perform the data entry transaction.

Since there are no constraints on who performs the data entry transaction, this also means that the product cannot be described in Sandhu's notation without extending that notation to cover steps where no separation of duty controls apply.

Although the product enforces separation of duty between authorizing officers and managers, the manufacturer suggests that roles can be shared in small branches, and, for example, one of the managers could also act as an authorizing officer [11]. This would be done by giving the manager two distinct identities from the point of view of the device. It must be assumed that some commercial security authorities have accepted that, with adequate external controls, improper switching between roles would be impossible to conceal in the small branch environment. Similarly, there are no separation of duty controls, of any form, between data entry and other types of transaction.

By contrast, even two distinct Green team or two distinct Yellow team officers cannot fully authorize an instruction: it is necessary to use one officer from each team. The Green and Yellow authorization transactions provide a static, role based, separation of duty control, and could be implemented by a TCSEC based approach such as that proposed by Lee [10].

There is no constraint that the Green and Yellow authorization transactions are executed in a fixed order, but both must be executed before an instruction becomes fully authorized. There is thus more than one legitimate sequence in which transactions can be executed, totally at the discretion of the officers concerned.

There are other aspects of this product that *do* match the requirements of Clark and Wilson. For example, the product provides a secure audit trail which includes full details of all transactions and names of the associated authorizing officers. It thus fully implements Clark and Wilson Rule C4 "**all TPs must be certified to write to an append-only CDI (the log) all information necessary to permit the nature of the operation to be reconstructed**".

Similarly, the division of authorizing officers into two disjoint teams and the selection of each day's active authorizing officers by the two managers working in concert at the start of day ensures that "**any proposed collusion is only safe by chance**" (see [4]).

The product also fully implements Rules C2/El concerning control of the transaction types that can act on objects in different internal states: in particular, no instruction can ever become fully authorized without both the Green and Yellow authorization transactions being executed upon it. It is also impossible for *anyone* in the branch to alter transactions or the types of data item they can act upon (Rule E4): as soon as the device is opened, its memory, including the cryptographic keys necessary to generate the MACs, is automatically and completely erased.

CONCLUSIONS

A number of authors have challenged the assertion by Clark and Wilson in [4] that lattice-based mechanisms are not sufficient to implement commercial integrity policies. In this paper we have shown that their constructions either fail to handle dynamic separation of duty, or do so by the use of unspecified mechanisms either "outside the TCB" or to be added to a lattice-based TCB. We have also shown that there is a need for such distinct mechanisms to implement facets of commercial integrity policies (such as object based separation of duty) that would be found in the real commercial world.

We have shown that it is possible to implement efficiently the form of dynamic separation of duty which we have called object based separation of duties.

We have presented the example of the RGL250, a real security product for the commercial market, that by its level of adoption in financial institutions clearly satisfies some real security needs. Unlike RACF or ACF/2, it cannot successfully be evaluated against the TCSEC, even against the criteria for Division C.

We have shown that certain aspects of this product fail to correspond to the Clark and Wilson model. Other aspects of the product not only match Clark and Wilson, they can be implemented by TCSEC based mechanisms.

It is interesting to speculate why a static interpretation of separation of duty is implied in [4]. It is worth pointing out that Clark and Wilson appear to have been motivated to write their paper from observed inconsistencies between TCSEC ratings of commercial products and their actual value as seen by the commercial world [3]. Thus it is perfectly possible that Rule E2 as worded reflected the controls they observed and believed were necessary in practice, rather than their explanation of what they saw.

ACKNOWLEDGEMENT

The authors would like to thank Racal Milgo Limited for permission to reproduce the photograph of the RGL250 included within this paper .

REFERENCES

[1] BELL, D.E.
"Secure Computer Systems: A Network Interpretation", Second Aerospace Computer Security Applications Conference, McLean, December 1986, pp. 32-39.

[2] BIBA, K.J .
"Integrity Considerations for Secure Computer Systems", USAF ESD Report ESD-TR-76-372, April 1977 .

[3] CHALMERS, L.S.
"An Analysis of the Differences Between the Computer Security Practices in the Military and Private Sectors", IEEE Symposium on Security and Privacy, Oakland, April 1986, pp. 71- 74.

[4] CLARK, D.D. and WILSON, D.R.
"A Comparison of Commercial and Military Computer Security Policies", IEEE Symposium on Security and Privacy, Oakland, April 1987, pp. 184-194.

[5] CLARK, D.D. and WILSON, D.R.
"Evolution of A Model for Computer Integrity", Working Draft Paper distributed at the Eleventh National Computer Security Conference, Baltimore, October 1988.

[6] DEPARTMENT OF DEFENSE
Department of Defense Trusted Computer System Evaluation Criteria, DOD 5200.28-STD, December 1985.

[7] JUENEMAN, R. R.
"Integrity Controls For Military and Commercial Applications", Fourth Aerospace Computer Security Applications Conference, Orlando, December 1988, pp. 298-322.

[8] KARGER, P .A.
"Implementing Commercial Data Integrity with Secure Capabilities", IEEE Symposium on Security and Privacy, Oakland, April 1988, pp. 130-139.

[9] KATZKE, S.W. and RUTHBERG, Z.G. (Eds)
Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS), NIST Special Publication 500-160, National Institute of Standards and Technology, January 1989.

[10] LEE, T.M.P.
"Using Mandatory Integrity to Enforce "Commercial" Security", IEEE Symposium on Security and Privacy, Oakland, April 1988, pp. 140-146 (This paper also appears as Appendix A7 to [9]).

[11] RACAL GUARDATA FINANCIAL SYSTEMS
RGL250 Portable Authenticator Brochure, Racal-Guardata Publication RG007/388/CL, 1988.

[12] SANDHU, R.
"Transaction Control Expressions For Separation of Duties", Fourth Aerospace Computer Security Applications Conference, Orlando, December 1988, pp. 282-286.

[13] SHOCKLEY, W.R.
"Implementing The Clark/Wilson Integrity Policy Using Current Technology", Appendix A9 to the Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS), Reference [9] above.

[14] SHOCKLEY, W.R.
"Implementing The Clark/Wilson Integrity Policy Using Current Technology", Eleventh National Computer Security Conference, Baltimore, October 1988, pp. 29-37 (This is a significantly revised version of [13]).

[15] TERRY, P. and WISEMAN, S.
"A 'New' Security Policy Model", IEEE Symposium on Security and Privacy, Oakland, May 1989, pp. 215-228.