Modelling Multidomain Security

José de J. Vázquez-Gómez*

Supélec-Rennes. Service Informatique·Avenue de la Boulais
BP 28. 35511, Cesson-Sévigné Cedex, France

Abstract

This paper outlines our ongoing work to deal with the Multidomain Security Problem. This problem arises when two systems in two different security domains try to interact with each other in a secure way. We propose an approach where all interactions between policies may be controlled by a multipolicy. The policies in the approach must be defined in considering the multidomain environment (i.e., where several different policies coexist). A policy specification should be built by including security properties in several specification steps. To decide what properties to include in each specification step, we propose to use the order of an "abstraction level hierarchy". This hierarchy is determined by intuition criteria applied to each security property in the multidomain. We argue that policies built in this way are better adapted to be controlled by an interaction Multipolicy and are easier to understand.

Keywords: *Security Policy, Security Model, Security Domain, Multidomain, Multipolicies.*

1. Introduction

In recent years various security policies have been proposed. Some of them have been modeled or implemented on distributed systems [1, 2, 3, 4]. Such systems tend to surpass their initial domains by extending over new ones which leads to new interaction needs. Because of secrecy, integrity, and/or availability reasons, these new interactions must be secure. Nevertheless, today there are no formal tools to specify and analyze these new inter-domain interactions. We address this interaction requirement as "Multidomain Security Problem". There are two main reasons that account for this problem: 1) Domains having different security authorities, and 2) Domains having different environments.

This section outlines the current security vision and shows why it is not applicable when different security domains interact.

Security domain definition

Some definitions found in literature are cited below.

"The security domain is a bounded group of security objects and security subjects to which applies a single security policy executed by a single security administrator" [5]. "The security domain concept may also be used to denote the scope of a specific set of security rules that apply, not to a network but, for example, to a single distributed application" [5]. "A security domain is the purview of a security administration" [6]. A domain is "The set of objects that a subject or resources in an automated information system has the ability to access" [7].

In this paper, we adopt the first definition.

Current security vision

Nowadays, International Organizations consider a single security policy. This policy may be separated into several security sub-policies, e.g. in TCSEC, secrecy, integrity and denial of service [8]. A TNI evaluation is based on an overall network security policy. It considers the interconnection of accredited AISs (Automated Information Systems) having the same security policy and different security attribute values [9]. The ITSEC enables users to specify their own security objectives (the user's security policy), and to choose a system or product called TOE (Target Of Evaluation) that satisfies their needs. An evaluation process verifies if the selected TOE complies with the security objectives. However, the user must integrate the security objectives into a single coherent objective called "security target for the TOE" [10]. The ISO proposes directives to place security services and security mechanisms in OSI architecture but is not concerned with a particular policy security definition [11].

The current vision is not applicable when interconnecting or combining systems where multiple policies are involved as we explain below.

* Supported by a grant from the CONACYT-SFERE fellowship program jointly held by México and France.
e-mail : jvg@univ-rennes1.fr

Multiple system authorities

Security controls vary from one system to another depending on the security criteria chosen by the local security authorities. When two systems are interconnected, the resulting extended system is managed by multiple jurisdictive authorities (see [12]). Even in one single organization it is possible to find several authorities [7]:

A Security Officer, whose responsibility is the security of the Automated Data Processing (ADP) system.

An Accreditation Authority, whose responsibility is to accredit systems handling sensitive information.

ADP Security Staff. Personnel working as "Action Officers" in their respective organizations.

Environment diversity

"Security policies differ widely because of differences in the environment in which organizations operate" [5]. Environment diversity is present at different levels:

Different nations. Each nation proposes its own norms and standards for national security enforcement.

Different social-economic sectors. Each sector has particular requirements which are not necessarily shared by other sectors (e.g. military, commercial, health, financial).

Different products. Because of subtle policy differences in policy implementation, products may not interact as expected; even if they enforce the same security policy, e.g. operating systems, data bases systems (see [13]).

Implications

Some of the implications of authority and environment diversity are:

- Different security policy interpretations [12, 13].
- Different security mechanisms are used to implement a single security policy [12].
- Different security policies are employed [5, 14]. The existence of different security policies in one organization is possible [15].

The first two implications are respectively specification and standardization problems. The third one is a consequence of particular social-economic sector needs.

Current solution

In current Multidomain Security Problem solutions, agreement between different domain authorities permits translating (mapping) of the sensitive data attributes being exchanged (see [5, 6]). Nevertheless attribute translation upgrades or downgrades information, making it difficult to know if security is still preserved. We consider this as a "discretionary interaction control" approach.

We propose a new solution where all interactions between different domains are controlled in a mandatory fashion (i.e. where no attribute translation exists).

This paper is organized as follows. Section 2 presents our approach. Section 3 introduces the abstraction level concept. Section 4 defines interaction conditions in the multidomain. Section 5 outlines the policy specification process. Finally, in section 6 we present our conclusion and the future work.

2. Approach

A multidomain is a set of security domains (see Figure 1). Each domain possesses its own policy to mediate local-domain accesses to objects. A domain may consist of one or several systems. All subjects in the multidomain have been authentified by a uniform mechanism (e.g. Kerberos [16]). In a multidomain, subjects in one domain can access objects in another domain according to multidomain rules. These rules constitute the multidomain policy or multipolicy. So, a multipolicy is used as a medium to control interaction between different policy domains (see [14]). In this approach, the main requirement to achieve this multipolicy is that the source and target domains possess some common security properties.

An "abstraction level" is associated to each security property. The abstraction levels will provide a hierarchy of properties in the multidomain which is called "abstraction level hierarchy" (ALH). This hierarchy is determined by intuition criteria applied to each security property in the multidomain. Abstraction levels will help to define the interaction multipolicy (see section 4).

In the multidomain context each domain policy specification must take into account that it will be applied in a heterogeneous environment, i.e., where different security policies may interact. So, a domain's policy is a set of rules taken from properties of different security policies (e.g. *-property, simple-security, conflict of interest). A domain authority specifies its domain policy from properties in the ALH. We argue that abstraction levels could help to decide what property to include in each step of a policy specification process. The expected benefit is a better understanding of combination and comparison of properties.

The multidomain security information is stocked in a database called MDSIB (MultiDomain Security Information Base). This database is maintained by the multidomain authority, and contains general security information. For example:

- The ALH of the multidomain (including all the properties that could exist in the domains' policies).
- The sets of attribute values for all property in the ALH. A property being used by different domains may have different sets of attribute values which are put together by the multidomain authority while respecting their structure (e.g. lattice, subset).

- The structure of the attribute values for each property (i.e. how values are related to each other, in partial orders, in subset relations, etc).
- The current security domains in the multidomain.
- The set of properties and attribute values for each domain (this constitutes the domain policy).
- The type of permited interactions between the different domains.
- The list of domains that are allowed to interact.

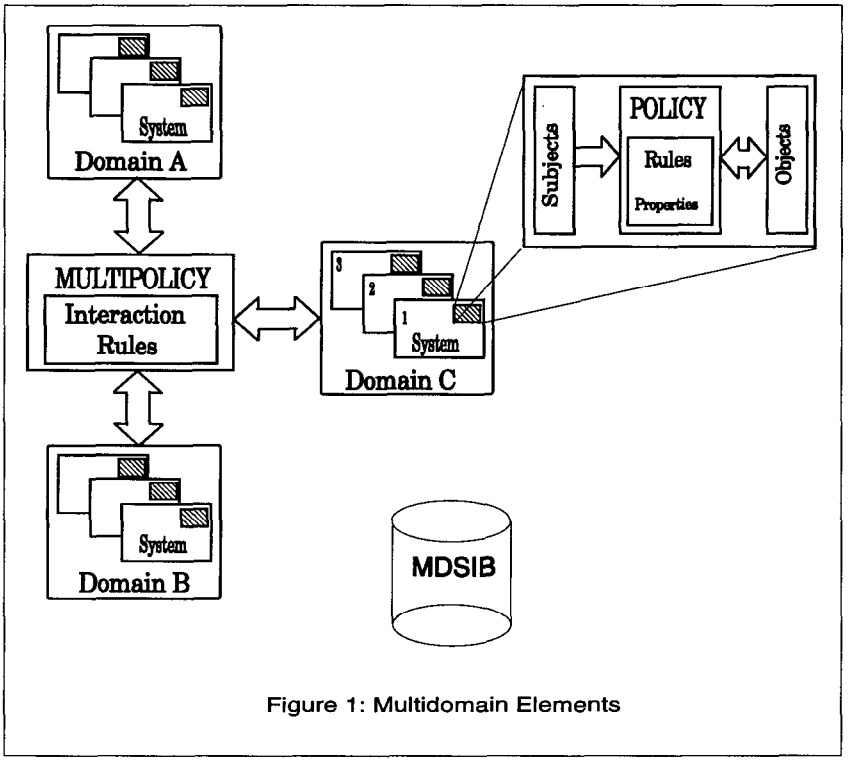

Figure 1: Multidomain Elements

3. Abstraction level

In comparing security policies, a single level is used. Properties of policies are considered as if they concern subjects and objects seen from the same perspective. It is said that "a Chinese wall cannot be correctly represented by a Bell-La Padula model" [4] or "the lattice model is not sufficient to characterize integrity policies" [3]. However, each policy perspective is quite different. For example, from a financial policy viewpoint, objects belong to different groups of "conflict of interest classes" [4] (e.g. groups of organizations competing for common markets). For a commercial policy these objects (possibly named otherwise) belong to different groups of "well-formed transactions" [3] (e.g. groups of procedures in a particular organization ensuring integrity and correctness in all the operations on objects). Finally, from a multilevel policy viewpoint, these objects belong to groups of "classifications" [1] (e.g. groups of confidentiality levels stating hierarchical conditions on objects use).

If we try to combine or compare properties of two different viewpoints in a single level, we will become entangled in "impossible problems" [13]. In our approach, security properties are considered to be non comparable but complementary. So, different policy properties are needed to model accesses in the real world.

Hierarchy of properties

In Figure 2, by assigning levels to different policy properties, we try to illustrate that it is possible to analyze the overall security in a natural way.

An organization requires its financial operations which are accessing objects to satisfy financial properties for protection against competition (e.g. Unsanitized Information Flow Control, Free Choice, Conflict of Interest). However, these properties do not differentiate between users executing financial operations. So, an employee may perform the same operations as the organization Director. One possible way to implement such operations, in taking into account differences between users' roles, is by defining well-formed procedures from a commercial perspective. Thus the subject executing a procedure on an object is indeed permitted to do so (e.g. separation of duty), and the procedure handles the object adequately (e.g. well-formed transactions). Procedures modifying Direction financial

information can be executed exclusively by the organization Director. In the same vein, to describe well-formed procedures that manipulate information in a constrained way from a multilevel security viewpoint, it is necessary that all procedures accessing objects enforce the multilevel properties. So, multilevel properties mediate access to information considering the hierarchical nature of subjects and objects in the organization. For example, the organization Director is cleared to access sensitive information through the execution of a financial operation.

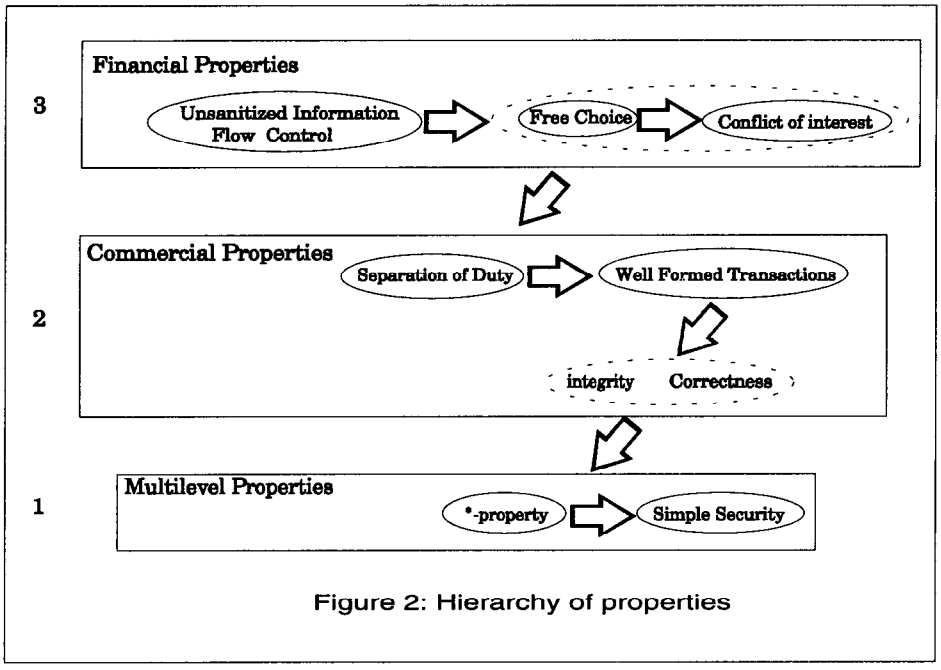

Figure 2: Hierarchy of properties

3.1. Abstraction level determination

This paragraph presents some intuition criteria that may permit determination of abstraction levels for security properties.

Dependency

There exists properties that are expressed in terms of other properties. In such cases, the dependable property has an abstraction level greater than that of the independent one. These properties occupy consecutive levels in the hierarchy. For example, the "*-property" includes the "simple-security" property as was initially stated in [1]. Thus, the "*-property" has a greater level than the "simple security" property.

Information flow

Properties describing information flow between two objects have a greater level than properties describing the access from a subject to an object. It is supposed that any information flow between two objects may be seen as two accesses; one access to the source object and the other to the destination object. More complex information flows may be described as for example from one object to many or from many to one, and they will even be at a higher abstraction level.

Dynamic access

Dynamic access properties have a higher abstraction level than the static access properties. The intuition here is that if the static access properties refuse the access to an object, this will be refused always independently of the previous access that the subject has made. Conversely, dynamic access properties refuse (or grant) the access to an object depending on the previous accesses (including static accesses).

Safety and liveness

In general there are two main property types: *Safety properties* and *liveness properties*. A safety property asserts that something bad does not happen, and a liveness property asserts that something good eventually does happen. In our approach, we suppose that bad things are actions that if happen compromise system security, and good things are actions that if happen do not compromise security or that are good for it. So, we propose to use safety properties to describe restrictions on 'bad things' and liveness properties to describe concessions for 'good things'. For example:

"A non-authorized user cannot access any sensitive information"; it is a safety property, i.e., a non authorized user will never have access to sensitive information.

"A cash desk user must read the payroll information"; it is a liveness property, i.e., user does not read this information all the time but sometimes (e.g. when he must pay).

In system security we may induce two boundaries:

- The upper boundary "Total Security" where no action can happen (e.g. every action is a 'bad thing');
- The lower boundary "Minimum Security" where any action may happen (e.g. any action is a 'good thing').

Actions represent operations in the multidomain. We suppose knowledge of all security concerned actions in the multidomain. So, if all restricted actions are known, we can conclude that all non-restricted actions must happen in the future (otherwise such non-restricted actions have no reason to exist) and vice versa, if all non-restricted actions are known, we can conclude that all restricted actions will never happen. Under these suppositions we can express the two boundaries of security as follows. For Total Security:

Must_happen{auxiliary action A} (a liveness property)
or equivalently
Cannot_happen{action_1, action_2, ..., action_n, auxiliary action B} (a safety property)

The liveness property expression give us a more abbreviated version of Total Security than the safety property expression. For Minimum Security:

Cannot_happen{auxiliary action B} (a safety property)
or equivalently
Must_happen{action_1, action_2, ..., action_n, auxiliary action A} (a liveness property)

For Minimum Security the safety property expression seems to be more condensed than the liveness property version. In Total Security the only action which *must happen* is the auxiliary action A. In Minimum Security the only action that *cannot happen* is the auxiliary action B. These two boundaries allow us to induce an order in security properties where safety properties are used to express lower abstraction levels (Minimum security) while liveness properties are used to express higher abstraction levels (Total security). For a certain "intermediary security", using liveness or safety properties do not simplify the security expressions.

Cannot_happen{action_1, action_2, ..., action_r, auxiliary action B} (a safety property)
or equivalently
Must_happen{action_s, action_t, ..., action_x, auxiliary action A} (a liveness property)

Note that action A must happen, and action B cannot happen in any "kind" of security defined between the boundaries Total and Minimum Security. Auxiliary action A may be for example "all subject is eventually audited", and auxiliary action B "non-authenticated subjects access the system". This particular way of determining abstraction levels assume that all security properties can be expressed in such a way (e.g. something that cannot happen or something that must happen in the future).

3.2. Abstraction level hierarchy

In the multidomain, properties will be ordered by their associated abstraction levels to form an ALH (Abstraction Level Hierarchy). An example of a possible hierarchy is (in decreasing order):

- *Unsanitized information flow control*
- *Free choice property*
- *Conflict of interest property*
- *Separation of duty property*
- *Well-formed transaction property*
- **-property*
- *Simple security property*

To induce this order, we apply the criteria. By dependency, the "Unsanitized information flow control" depends on the "Free choice property" and the "Conflict of interest property"; "Free choice property" depends on "Conflict of interest property" as stated in [4]. "Separation of duty" depends on the "Well-formed transaction" property [3]. The "*-property" depends on the "Simple security" property [1]. According to the information flow criterion, the "Well-formed transaction" property is greater than "*-property" because a "Well-formed transaction" may also be considered as a complex information flow (e.g. where a Transformation Procedure takes a Constrained Data Item, CDI, and outputs one or more of them). Finally, by dynamic access, "Free-choice" is greater than the "Well-formed transaction" and the "*-property". The safety-liveness criterion requires expression of these properties as something that must happen or something that cannot happen maybe in using a formal language such as "Security Logic" [17].

4. Interaction multipolicy

During interaction, information leaving a domain must possess all the security attributes of the source domain policy. This information does not necessarily possess the same attributes as the destination domain. The multipolicy must verify that all leaving attributes possess the adequate values (e.g. that all the non-common attributes were set to their minimal values). After verification, the source domain non-common attributes are removed. Then the multipolicy adds to the information the destination domain non-common attributes carrying minimal values.

Minimal attribute values

Since the set of attribute values associated to a particular property depends on the domain authority choice, these values may be different for the same property in different domains. However, there are values that must be known in all domains as, for example the "minimal values". There exists a minimal value for any property attribute (e.g. the attribute "clearance" for simple-security may have the minimal value "unclassified"; the attribute "conflict of interest class" for the Chinese Wall policy properties may have the minimal value "sanitized"). Information possessing a minimal value on one of its attributes indicates that it is not a sensitive information from the viewpoint of the property associated to such attribute.

Multipolicy

Some conditions must be fulfilled to allow interaction between domains. These conditions are a part of the informal description of the multipolicy controlling interactions in the multidomain.

Condition 1. The interaction between two domains is possible if they have at least one common security property.

Condition 2. The interaction between two domains possesses secure communications if their respective set of properties contains the "basic properties" (condition 1 is consequently satisfied). Basic properties should be satisfied by any security policy in the multidomain since they enable secure communication and reflect the implicit hierarchy of individuals in one organization. We propose the MLS properties as "basic properties" (see section 4.1).

Condition 3. This condition is applied when condition 2 is satisfied. In such a case, the interaction control will permit utilization of the whole set of attribute values associated to each one of the common properties (e.g. all of the classifications of security). For the non-common properties, the interaction control will permit utilization of the respective attributes set to their minimal values. For example, we have two domains having a common commercial property, one of them has in addition a financial property. During interaction, commercial interactions can use as attribute values "CDIs" (Constrained Data Item) or "UDIs" (Unconstrained Data Item) while financial interactions can only use "sanitized" as attribute value (i.e. their minimal value).

Condition 4. This condition is applied when condition 1 is satisfied but not condition 2. In such case, the interaction control will permit utilization of a *particular* set of maximum attribute values associated to each one of the common properties. This set of attribute values is determined by the multidomain authority (information possessing higher attribute values should not be permited because of the absence of communications security). For the non-common properties, the interaction control will permit utilization of the respective attributes at their minimal values.

Condition 5. Two domains may interact in a particular abstraction level if their policies possess a common property at that level and from this level they possess a peer-domain interaction type (see below). Properties at higher levels must be considered as non-common properties. For example, in a hierarchy such as in Figure 2, a "commercial" interaction level indicates that the interacting domains possess "commercial" and "multilevel" common properties; "financial" information must carry attributes having minimal values even if both domains possess "financial" properties.

Interaction level

The interaction level that may be attained in a relation between two domains is determined by the hierarchy of properties of each domain.

For a *"peer-domains"* relation, where two domains have a common subset of policy properties including the basic properties, a level of interaction may be selected.

The lowest interaction level in a peer-domain relation occurs when domains interact at the basic properties abstraction level. Domains which possess policies described by the same hierarchy of abstraction levels may select the level of interaction from this hierarchy (see condition 5). For a peer-domain relation between DOMAIN 1 and DOMAIN 2 (below) any property abstraction level may be used to interact.

DOMAIN 1	DOMAIN 2
Property n	*Property n*
...	...
Property y	*Property y*
Property x	*Property x*
...	...
Basic Properties	*Basic Properties*

If one of the interacting domains possesses, besides its own properties, all the properties of the other one (relation *domain to subdomain*), and condition 2 is satisfied then their highest interaction level will be equal to the highest abstraction level in common (e.g. at the Property y abstraction level below).

DOMAIN 1	DOMAIN 2
Property n	--------
...	...
Property z	--------
Property y	*Property y*
Property x	*Property x*
...	...
Basic Properties	*Basic Properties*

If two interacting domains possess at least one common property (abstraction level) and the peer-domain and domain to subdomain relations are not satified then their relation is *domain to domain*. The interaction must satisfy condition 4.

DOMAIN 1	DOMAIN 2
Property n	*Property n*
Property n-1	*property w*
...	...
Property z	--------
Property y	*Property y*
Property x	*Property x*
...	...
Basic Properties	*property a*

4.1. Basic Properties

In all organizations exist an implicit connotation of hierarchy. For example, an organization has a president, directors, managers, administrative personnel, sales personnel, and so on. This hierarchy may be described by a structure of levels such as classifications in a multilevel policy. Operations handling information on behalf of the organization users should be mediated by properties taking in account their multilevel structure. In addition, distributed systems need controls such as access control and information flow control between network devices. These controls should be applied when subjects and objects which are classified differently or possess different attributes require:

- To use common system utilities (e.g. mail functions, directory services, data base servers);
- To access remote objects by passing over several network devices.

These controls may be enforced by using properties of a policy such as MLS adapted to distributed systems (see [18, 19]). We consider that MLS properties (e.g. simple security and *-property) may be retained as the basic properties. Basic properties are the lowest abstraction level in the ALH.

5. Policy Specification

The specification of a particular domain policy could be thought as a process of security properties inclusion. Initially, all the policy properties are selected from the ALH of the multidomain. In order to include these properties, successive specifications are needed. This implies a previous knowledge of the security requirements that are concerned in each specification step. Usually, this is based on experience. This is not a trivial problem. We propose the abstraction levels to decide how security properties are to be included in each specification stage. For the ALH proposed above (section 3.2), "Unsanitized information flow control" is the first property to be included, and "simple security" the last one.

Policy specification may also follow some rules (e.g. a multipolicy to help in the policy specification process). For example, possible rules are: "All selected properties to specify a policy exist in the ALH", and "If a dependable property is included in a policy then the property from which it is expressed must be also included".

5.1. Intermediary structure

An intermediary structure is necessary to describe the characteristics of the security properties in the multidomain. An example of the elements of this structure is:

- *Property name;*
- *Source policy (e.g. from which traditional policy the property has been obtained);*
- *Property abstraction level;*
- *Associated attributes and the set of their possible values;*
- *Dependency with other properties;*
- *Informal description;*
- *Formal description (e.g. in a common language).*

6. Conclusion

This paper outlines our ongoing work on the Multidomain Security Problem.

In a multidomain, multiple security policies interact. The current security vision does not consider these interactions, hence a new vision is needed. To handle this problem we propose to construct policies by selecting properties from former traditional policies. We associate an abstraction level to each security property by applying intuition criteria. Such levels allow us to build a hierarchy of properties called "abstraction level hierarchy". Thus, all policies in the multidomain will be built from properties in this hierarchy. We define a Multipolicy as a set of conditions to permit interaction among different policy domains. Finally, we describe briefly the process of domain policy specification through an ordered inclusion of properties. Future work will provide: a more formal abstraction level determination, a common formal language for property specification, a formal description of multipolicy conditions.

Acknowledgments

I would like to thank the anonymous referees for their insightful comments to this paper, Roland Baduel for supervising my work, and all the participants of the New

Security Paradigms Workshop II for their comments to my work.

Glossary

ECMA	European Computer Manufacturers Association
ITSEC	Information Technology Security Evaluation Criteria
ISO	International Standards Organization
MLS	Multi Level Security
TCSEC	Trusted Computer System Evaluation Criteria
TNI	Trusted Network Interpretation

References

[1] D.E. Bell and L.J. La Padula, "Secure Computer Systems: Unified Exposition and Multics Interpretation", MTR-2997, MITRE Corp., Bedford, Mass., July 1975.

[2] C.E. Landwehr, C.L. Heitmeyer and J. Mclean, "A Security Model for Military Message System", ACM TOCS, Vol. 2, pp. 198-222, 1984.

[3] D.D. Clark and D.R. Wilson, "A Comparison of Commercial and Military Computer Security Policies", IEEE Symposium on Security and Privacy, 1987.

[4] D. Brewer and M. Nash, "The Chinese Wall Security Policy", IEEE Symposium on Security and Privacy, 1989.

[5] ECMA, "Security in Open Systems : A security Framework", ECMA TR/46, 1988.

[6] STANDARD ECMA-138, "Security in Open Systems : Data Elements and Service Definitions", December 1989.

[7] ISO, "Glossary of Information Technology Security Definitions", International Standards Organisation Publication ISO/IEC JTC1/SC27 N270., 1991.

[8] U.S. Dep. of Defense, "Dep. of Defense Trusted Computer System Evaluation Criteria", Rep. DOD 5200.28-STD, 1985.

[9] National Computer Security Center, "Trusted Network Interpretation of the TCSEC", NCSC-TG-005, vers.1, July 1987.

[10] Service Central de la Securite des Systemes d'information, "Criteres d'evaluation de la securite des systemes informatiques (ITSEC)", 1990.

[11] ISO, "Information Processing Systems -- OSI Reference Model -- Part 2: Security Architecture", International Standards Organisation Publication No. 7498 Part 2. 1989.

[12] D.M. Nessett, "Factors Affecting Distributed System Security", IEEE TOSE, February 1987.

[13] H. H. Hosmer, "Integrating Security Policies", Proceedings of the Third RADC Workshop of Multilevel Database Security, Castile, NY, June 1990, MITRE MTP 385, May 1991.

[14] H. H. Hosmer, "Metapolicies I", ACM SIGSAC Data Management Workshop, San Antonio, TX, December 1991.

[15] J. P. Kruys, "Security of Open Systems", Computers & Security, Vol. 8, April 1989.

[16] S. P. Miller, B. C. Neuman, J. I. Schiller, and J. H. Saltzer, "Kerberos Authentication and Authorization System", MIT Project Athena Technical Plan, Section E.2.1, december 1987.

[17] J. Glasgow, G. MacEwen and P. Panangaden, "A Logic for Reasoning About Security", ACM TOCS, Vol. 10, N 3, August 1992.

[18] W.P. Lu, and M.K. Sundareshan, "A Model for Multilevel Security in Computer Networks", IEEE trans. on software engineering, vol.16, No.6, June 1990.

[19] V. Varadharajan, "Network Security Policy Models", Lecture Notes in Computer Science, N453, pp 74-95, 1990.