SECURITY MODEL FOR HEALTH CARE COMPUTING AND COMMUNICATION SYSTEMS

Anas Abou El Kalam — Yves Deswarte
LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse cedex 4, France.
anas@laas.fr, deswarte@laas.fr

Abstract Health Care Computing and Communication Systems (HCCS) are characterized by the complexity of the organizations to take into account and the richness of properties that are required. To address this complexity and richness, we propose a security policy based on roles, groups of objects and context. Indeed, similarly to roles that structure the subjects, we introduce the new concept "group of objects" which structures objects. Our major aim is to facilitate the security policy management, to cope with access right complexity, and to reduce administration errors. Then we develop a security model that covers the diversity of HCCS while achieving a good compromise between the respect of the least privilege principle and the flexibility of the access control. Following a logical approach, we design a formal system that extends the deontic logic, and we express the security policy in our language.

Keywords Security policy and models, RBAC, Context aware computing, deontic logic.

1. PROBLEM DEFINITION

Recent progress in telecommunications has influenced in various levels the medical processes while this imposes stronger requirements on systems and networks to achieve various dependability attributes: reliability, safety, maintainability, confidentiality, integrity and availability. The aim on this section is to describe the HCCS, to identify the information to protect as well as the threats these systems are facing, and to describe security requirements that can counter the identified risks.

1.1 Identification of sensitive information

A HCCS can be defined as a large network dedicated to healthcare, connecting clinicians, patients, healthcare and social security organizations and interfaces to other systems (e.g., disease data banks). The system should possess reliable means assuring the communication, the processing, as well as the backup of medical, paramedical, administrative and financial information. Ethical obligations aim to protect the privacy of personal data. Unfortunately, even when identities are hidden, it is often possible to identify a person by combining some information, like telephone numbers, medical history or health insurance data. In addition to sensitive data, laws aim at protecting other HCCS resources and services. Indeed, the resolution 45/95 of the General assembly of the United Nations adopted guidelines for the regulation of computerized personal data files [1]. The European Commission has qualified the protection of the medical records as a priority: the Council of Europe has established recommendations concerning the automated medical data banks [2]; the European Parliament and the Council of Europe have adopted diverse directives: on these topics [3, 4, 5].

1.2 Some identified risks

Risk analysis can help to evaluate the consequences of existing vulnerabilities on the system security. This section identifies which entities to protect and enumerates some threats. But first, let us recall some definitions [6]: a v*ulnerability* is a weakness in the system (an accidental fault, or a malicious or non-malicious intentional fault in the requirements, the specification, the design or the configuration that can be exploited to provoke a security failure); an a*ttack* is a malicious interaction fault exploiting a vulnerability and aiming to intentionally violate one or more security properties; an *intrusion* is a malicious, externally-induced fault resulting from an attack that has been successful in exploiting a vulnerability. Some of the laws mentioned above identify several threats. For example, [5, art 4] specifies that the supplier of an electronic service (e.g., a data warehouse) must take technical and organizational measures to guarantee the security of its services. It should also inform the subscribers (e.g., patients) about risks. On the same way, [1, guideline 7] lists some threats: accidental loss; destruction; human risks; non-authorized access; data diversion; viruses. HCCSs have to face specific threats such as patient's privacy violation by illegitimate means (e.g., exploiting vulnerability on the anonymisation process or on the socio-technical system), linking information (e.g., the knowledge of two of a woman's childbirth dates is sufficient to identify her), denial of HCCS critical services (emergencies, payment), illegitimate modification of data (e.g., allocation rules for social rights), etc.

The US government's Office of Technology Assessment [8] and the UK audit Commission [9] confirmed that healthcare appears to be one of the most attractive targets of attacks: both internal and external abusers attempt to harm the HCCS security (invasion of privacy, hacking, virus, fraud, theft of data or software). The threat against privacy is increased by data aggregation that networked computer systems encourage. In HCCS, an intrusion has a very critical aspect, because: if medical information is disclosed illegitimately, besides the violation of the privacy, it may cause a catastrophic damage to the patient (e.g., insurances might refuse to insure persons who have fatal diseases); if information is corrupted, clinicians may take incorrect decisions, which may harm or even kill patients; if information may occasionally be unavailable due to system failure or sabotage, both clinicians and patients will loose confidence in the system.

To sum up, if a HCCS is not secure, its value as a basis for clinical decisions is diminished. Moreover, clinicians called to justify their actions may not be able to rely on computer records for evidence [10].

1.3 HCCS Security requirements

Having identified the risks that HCCS have to face, we can define their security requirements according to confidentiality, integrity, and availability.

Confidentiality can be defined as the capacity to protect sensitive information from being disclosed to unauthorized recipients. In HCCS, confidentiality has two meanings: *privacy* founded on the rights of the patient to control the access to his data; *secrecy* of sensitive information maintained by the organisation (e.g., hospital).

Integrity is defined as the non-occurrence of inadequate information alterations. The system must ensure that patient records, as well as applications, are changed only in an authorized manner. The problem is more complex here because a HCCS is a largely distributed system. But at the same time, it must ensure the consistency and the accuracy of data stored even in case of system failures. Data integrity might also include verification of the data-entry validity. For instance, for a personal pseudonym-code to be always the same for a given patient, it is crucial to use methods to minimise the consequence of spelling mistakes (e.g., when nominative information is transformed by a compression or anonymisation algorithm).

Availability corresponds to readiness for usage. We distinguish: *(1) Short-term availability*: the system resources must be available to authorized users with reasonable response-time (process or data may have timeliness constraints). In emergency cases, clinicians must be able to access the medical data in a reasonable delay. *(2) Long time availability*: regulations impose that some records must be kept for a very long time: cancer records must be kept for the patient's lifetime, and records of genetic diseases have to be kept even longer. Only some well-identified users should have the

ability to delete those data, and only after the appropriate time period has expired. It may also be desirable for clinicians to have at their disposal a longitudinal patient record, covering different periods and providers.

2. TRADITIONAL SECURITY MODELS

2.1 Security policy based on the RBAC model

The basic concept of Role Based-Access Control (RBAC) [11] [12] is that roles are assigned to users, permissions are assigned to roles and users acquire permissions by playing roles. A user's role may reflect his tasks in the organization. Hierarchical RBAC supports the role hierarchies, while constrained RBAC enforces the separation of duties. RBAC is used to alleviate user and resource administration: since permissions to access resources are associated with roles rather than users, an update in the users list has no effect on the managed resources, and vice versa.

2.2 Security policy based on the TMAC model

A TeaM-based Access Control (TMAC) was originally proposed in [13]. TMAC uses Teams to resolve some access control problems in collaborative environments. A concrete use of the team concept has been presented in C-TMAC (Context TMAC) [14]. C-TMAC gives a framework to deduct user permissions by combining roles and teams permissions. It also extends original TMAC to use other contextual information.

2.3 Discussion

RBAC is a good model that could be applied to most of actual organizations. To remain as general as possible, RBAC does not detail actions realized on the objects. One of the principal RBAC limits is that all the users having the same role possess the same permissions. However, even if two clinicians have the same role, they do not have necessarily access to the same patient records: only the attending clinician of a patient can consult his medical record. Indeed, in HCCS context, RBAC should be extended to cover other aspects, like the existing relation between the clinician and the patient; the involvement of the clinician, at the moment of the request, and in the process of care; and other contextual information such as location, time, emergency, etc. Our model extends RBAC with such notions.

The notion of team will also take part in our model because in HCCS users are indeed assigned to various teams. In C-TMAC, team permissions are deducted from the combination of the permissions of different users participating in the team. The combination can be an *aggregation*, a *maximum* or a *minimum*. We believe that this approach has certain weaknesses: (1) if the combination is an *aggregation*, the set of team

permissions is the sum-up of the permissions of all team members. The derivation of permissions will assign the same set of permissions to every member of the team. But even if users belong to the same team, they do not play the same roles, so they should not have the same permissions. (2) If the combination is a *maximum* or a *minimum*: the permissions will have no sense in this application and does not resolve any particular problem of the HCCS. Moreover, contrary to C-TMAC where a user can activate any one of his roles in any one of his teams, in our model, the user can play different roles in different teams. In other words, C-TMAC consider separately two binary relations (user, role) and (user, team) but, as we will explain later, our model consider a ternary relation (user, role, team). Our proposal is more flexible since, for example, a physician can participate to two teams (*Ti* et *Tj*) but he can play the role "*team manager*" only in *Tj*.

3. CONCEPTS FOUNDING OUR MODEL

In the first step of this study, we opted for a cross use of roles and teams. We perceive the team as an entity grouping together a set of users having various roles and collaborating to realize a specific task. As mentioned in the previous section, in accordance with the relation *(user, role, team),* a user can activate a subset of his roles in each of the teams which he participates in. Sometimes, it is useful to impose *constraints* on team administration, like the max/min number of users in a given team (e.g., the surgery team should be constituted by at least: a surgeon, a nurse and a medical secretary); the max/min number of users activating a specific role in the team; a max/min number of users, playing a specific role, who may/must realize a certain task (e.g., in team *X*, every morning, at least one of the nurses should take the patient's temperature).

In the rest of this chapter, we present the novel concept of "group of objects" and we detail the various types of HCCS contexts. Active entities (which manipulate the information; e.g. users) are denoted as *subjects*, and passive entities (which essentially contain the information, and on which subjects realize actions; e.g. medical record) as *objects*.

3.1 Group of objects

Group of objects is the most novel concept of this paper. It is natural because, on the one hand, it is extracted from the real functioning of several organizations and, on the other hand, it contributes to facilitate comprehension, expression and management of the security policy.

In fact, a HCCS is a set of organizations that interact (e.g., hospitals, insurance companies, etc.). The hospital is a structure associating several organizations (e.g., units, services, departments). Every organization is a

structure that establishes the link between a team and the objects that it manipulates (e.g., a healthcare unit implies the presence of a medical *team* that provides care to the patients of the unit). It is thus essential to build *groups of objects* (such as patient records) associated to units and to define the access rights for the teams of the unit and for the other users which interact with it. Furthermore, the specification can identify different classes of objects (e.g., record, equipment, etc.). But, from the access control point of view, this classification is insufficient and it is necessary to distinguish the different objects (of the same class) on which subjects can have different permissions. We propose that permissions be broken down into: a deontic operator (authorization, interdiction, obligation or recommendation); an action (e.g., read, write); and a group of object (e.g., all the records of the patients treated by the team T_i). So, instead of concentrate our model on role permissions, we focus it on actions to realize on groups of objects. It is important to note that groups of objects are constructed according to *logical criteria based on access rights*. Intuitively, the groups G_{di} (resp. G_{dj}) can designate the records Rec_{i1} ... Rec_{in} (resp. Rec_{j1} ... Rec_{jm}) handled by the team T_i (resp. T_j). Records Rec_{in} and Rec_{jm} are both instances of the class *record*, but they belong to two different groups. In oriented-object language, this can be interpreted by the creation of the method *ChangeTeam()* of the class *patient*. This method allows changing the attending team of a patient, by changing the value of the variable *AttendingTeam*, so as to give to the new team the permission to access to the medical record of the patient. Patients having the value of the attribute *AttendingTeam* equal to T_i, constitute the group of objects "*Patients treated in* T_i*"*. From a relational database point of view (SQL), the elements of this group are the answer to the request: SELECT *AttendingTeam*, *RecNum* FROM *Record* WHERE *Record.AttendingTeam*=T_i. To clarify this point, let us detail how to construct groups of objects:

Step 1: according to a *logical view*, group together objects so as to distinguish between objects on which various groups of subjects realize different actions. In HCCS, the semantics could be some criteria such as: being part of a hospital, of a project, of a process, etc.

Step 2: establish the links between each group and the actions made on the group objects. *Steps 1 and 2* can be done by grouping together objects on which a subject (e.g., team) realizes the same actions. By playing a given role, a user obtains permissions allowing him to realize actions, not on all the objects of a class, but on the instances belonging to the group.

Step3: establish the *inheritance* of classes of objects and the *composition* of groups of objects. For examples, a resource of the emergency unit *is a* resource of a patient ward (inheritance); resources of the surgical ward C_5 can be made up from three groups: *rooms of the surgical unit* C_5, *workstations of the surgical unit* C_5 and *records of the surgical unit* C_5.

This leads us to discuss the usefulness of other groups of objects. Indeed, inheritance and composition relations reduce the complexity of administration: they favour the distribution of the values of subclasses attributes towards the super-classes, and the actions on the aggregate towards constituents. In addition, the cost of associations {action, object} (without using groups) is of the order N_A*N_O, (N_A: number of actions; N_O: number of objects) while by using the groups, the cost is N_A+N_O. Moreover, the entities (roles, classes and groups), as well as associations {Role, team}, {Permission, action}, {action, groups} remain relatively stable in the information system; they can so be managed by the administrator. The association {objects, group}, for example (Marie, *patient in the ward* C_5), changes more often and it can be managed locally (assignment of patients to wards can be done by the reception staff). Thus, the new concept *group of objects* contributes to reduce the risk of errors and the cost of administration.

3.2 Context

Our model takes into account the context in which the access request is made. A context can be defined as any information that can be used to characterize the situation of an entity (person, place, physical or computational object) [20]. Several existing access models have used the notion of context: Covington et al. [21] expands the RBAC model by incorporating environment roles to capture environmental attributes; Bertino et al. [22] have examined temporal authorization in database systems, etc. For a potential oriented-object implementation, our work distinguishes different types of context and associates each type to the corresponding entity. Our interpretation considers the context as an attribute of classes such as: role, object, team, etc. For example:

The *context of a role* identifies the values that certain variables should have when a user requires to play a given role. For example: the role *ward physician* is valid during the normal hours of work whereas the role *duty physician* is valid at night. Constraints can be associated to roles. For examples, cardinality (the maximal number of users authorized to play this role); static mutual exclusion (the same user cannot be authorized for different roles, e.g., clinician and accountant) and dynamic mutual exclusion (the same user cannot be simultaneously in two given roles, e.g., physician at the hospital and expert for an insurance company) [15].

The *context of objects* defines specific contextual attributes of objects or groups of objects. For example, a duration attribute for the storage of certain data; a location attribute: healthcare units have their own files (stored and managed locally) and their own workstations whose use is authorized only to the unit members.

Users attributes describe characteristics that can influence access control: age, specific authorization, temporary rights, etc. For example, the

membership of a national/international professional union; the experience in the practice of certain types of care; specific knowledge, etc.

The *context of use* is a novel concept that helps to realize a good trade-off between the respect of the least privilege principle and the needed flexibility of access control, in order to facilitate the healthcare professional work while preserving patient interests. We are partially inspired by the notion of *purpose* introduced in [16]. In HCCS, we think that if a user requires access to a resource, the context must belong to one of the two following cases:

(1) the user's team participates in a *process treating the patient*. In this case, the activity of care (process) has been already defined by authorized persons (e.g., the consulting physician) and registered into the server. For example, let us take the case of a patient that has heart problems; his consulting physician makes a first evaluation and initializes *the process of care* (he registers the activity of care into the server). Then he sends the patient to the hospital where specialists complete the diagnosis. The members of the team C_5 of the cardiology care unit take over the care of the patient and access, each one according to his role, the parts of the shared records (that contain the current medical history of the patient as well as temporary results and opinions) and archival records that interest them. The sharing of the data, belonging to this patient, among these various clinicians is made through the framework of the process of care declared by the consulting physician.

(2) Sometimes, even if there is no current process, certain users can declare a *purpose* (e.g., only the consulting physician is permitted to create a process, but in case of emergency, another doctor can declare the purpose "emergency" and so provide care to the patient). Security rules should precisely specify which user/role has the right to declare which purpose, in which conditions (*upstream control*). For instance, for the purpose "emergency", the access must be validated by values of certain environment variables: request made by an emergency ward; patient wounded in a vital organ; lack of personnel, etc. Permissions for a normal use in the framework of a process of care differ from permissions in emergency situations, and both differ from permissions for research/statistical purposes. Certain uses require the patient's consent; others, sometimes more urgent, take into account the purpose mentioned and grant the access with a higher responsibility and auditability (*downstream control*). For example, to favour flexibility, a security rule can authorize the physician who has formerly treated the patient to access the patient's current medical record, provided that he specifies as purpose *diagnosis revision*. This purpose will be the essential point of the authorization and will activate automatically a high-level audit or the automatic sending of a notification to the patient. To summarize, the aim of *purpose of use* is to assure flexibility while determining the responsibility of the user and to provide evidence in case of

problem or abuse of privilege. Furthermore, we notice that in addition to access control, our model takes into account the obligation to provide some measure such as accountability and replication (for increasing availability). Obligation can concretely be implemented by automating certain actions.

There are several means that can implement our approach, especially, the capabilities proposed by MAFTIA [19] or a language interpreted by access control mechanisms. For example, the use of XML documents to support authorization-based access control for web applications [17].

4. FORMAL SYSTEM

In this section, we propose a model able to formally express the security policy, the operational rules and the security requirements. The use of a formal approach allows to: manipulate the specification by using logical proofs; verify the security policy consistency; and verify that access control mechanisms satisfy the security requirements. Deontic logic [18] is a formal framework that is adaptable to specify security policies by means of modal operators such as obligatory "*O*", permitted "*P*" and forbidden "*F*". The specification language we present in this section relies on deontic logic.

4.1 Alphabet of the language

The alphabet of our language is made up of the following sets: constants, functions, variables, predicates and actions.

(1) Constants: designate the entities: users (e.g., Bob), roles (e.g., physician), teams (T_i), patient wards (e.g., radiology), files (f_i), data (d_j), group of objects (g_j), patient (p_i), purpose (e.g., scientific research).

(2) Variables: u∈*User*, r∈*Role*, t∈*Team*, d∈*Data*, o∈*Object*, go∈*group of objects*, w∈*ward*, p∈*Patient*, rec∈*record, f*∈File, *proc*∈Process, etc.

(3) Function: is an element of the language that permits to build the terms. For example, a shared-record can be expressed by the function $shared-record(patient \times File \times File \times File \times File \times File \times File) \rightarrow Record$, arguments are respectively: patient identity; motive of the consultation; medical examination report; anaesthesia data; operating reports; prescriptions; and nursing report. Furthermore, any constant or variable is also a term.

(4) Predicate: relations are denoted by predicates, e.g., AURT(u,r,t) associates the roles that a user can play in each of his teams; AWard(t,go,w) associates the teams and the groups of objects to their patient ward; AObj(o,go) indicates the membership of objects to groups; ARec(p,rec) allows to establish the link between the patients and their records; etc.

(5) Actions: e.g., TRANSMIT*(u,f,u')*, $CREATE_t$ *(u,t)*: the user *u* creates a new team *t*, $ADD_{u,t}$ *(u,t,u',r)*: *u* adds *u'* to *t* and assign him the role *r*.

4.2 The language

By means of predicates and actions, we can define the atomic formulas: if "A" is a *predicate* or an *action* of type $\lambda_1 \ldots \lambda_n$ and if $t_1 \ldots t_n$ are terms of type $\lambda_1 \ldots \lambda_n$, then $A(t_1 \ldots t_n)$ is an atomic formula. For example: AR(*Bob*, *clinician*) or READ$(Bob, record_t)$. The language is generated by the following grammatical rule, given in notation EBNF (Extended Backus Normal Form), where f is a formula: $f ::= A(t_1,\ldots t_n) \mid \neg f \mid f \vee f \mid f \wedge f \mid \mathbf{O}f \mid \mathbf{P}f \mid \mathbf{F}f$

4.3 The semantics

The language semantics is defined by the model $M = \langle W, \Re, V, D \rangle$ where W is a set of possible worlds; $\Re$ is a binary relation over W called *accessibility relation*; D is a domain (non-empty set of values); V is a function that gives truth values to the elements of the language: $V(w,A) \subseteq D^n$, $V(w,\pi) \subseteq D^n$, $V(x) \in D$ and $V(a) \in D$ (A: predicate, π: action, x: variable, a: constant). Intuitively, $(Bob, doctor) \in V(w,AR)$ means that in the world w, *Bob* plays the role doctor; $(Sam, record_X) \in V(w, READ)$ means that in the world w, *Sam* executes the *READ* action on $record_X$.

4.4 Truth conditions

In a modal logic language, the notation $M,w \models p$ signifies that p is true in a world w of model M; our language truth-values, which extend usual propositional calculus, are as follow: $M,w \models A(t_1,\ldots t_n) \Leftrightarrow (V(t_1),\ldots,V(t_n)) \in V(w,A)$; $M,W \models \neg f \Leftrightarrow$ *we do not have* $(M,w \models f)$; $M,w \models f \vee g \Leftrightarrow M,w \models f$ *or* $M,w \models g$; $M,w \models f \wedge g \Leftrightarrow M,w \models f$ *and* $M,w \models g$; $M,w \models \mathbf{O}f \Leftrightarrow \forall w' \in W / w \Re w' \rightarrow M,w \models f$; $M,w \models \mathbf{P}f \Leftrightarrow \exists w' \in W / w \Re w' \rightarrow M,w \models f$; $M,w \models \mathbf{F}f \Leftrightarrow \forall w' \in W / w \Re w' \rightarrow (M,w \models \neg f)$

The formulas $\mathbf{O}f$ (it is obligatory that f), $\mathbf{P}f$ (it is permitted that f) and $\mathbf{F}f$ (it is forbidden that f) mean respectively: f is true in every world w' which w is in relation with; it should be possible to reach a world in which f is true; none of the accessible worlds should allow to conclude that f is true.

5. SPECIFICATION OF THE SECURITY POLICY

5.1 System description

The system described by means of the propositional logic operators (non modal). We represent essentially the functional aspects of the system that are relevant for security. At the level of semantics, the operational rules define the internal structure of the worlds ($q \rightarrow r$ means in any world w where q is true, r is also true). For example, take the role hierarchy: $\mathrm{AR}(u, On-callPhysician) \vee \mathrm{AR}(u, WardPhysician) \vee \mathrm{AR}(u, ServicePhysician) \vee \mathrm{AR}(u, nurse) \rightarrow \mathrm{AR}(u, ClinicalStaff)$ or take the role activation: $\texttt{PROVIDE}(u, HPC(\_,\_,r,\_,t,\_)) \rightarrow \mathrm{AURT}(u,r,t)$: if the user provide his HealthCare Professional Card (*HPC)*, then he plays the role

in the team mentioned on the card (the function *HPC* has as parameters: identification number, name, role, mode of duty, team, rate).

5.2 Security properties

Security properties are expressed by using modal operators. These operators allow modifying the properties of the accessibility relations between the various worlds of the model. They indicate if two worlds are accessible from one to the other. $\mathbf{F}[\mathrm{AR}(u, pharmacist) \wedge \mathtt{CREATE}(u, prescription)]$ is a rule that forbids a pharmacist to create prescriptions.

5.3 Security rules

A security rule is a modal formula with at least a non-modal clause (e.g., $r \rightarrow \mathbf{P}q$). It describes the link between the permissions, interdictions or obligations, and the state of the system. $\mathrm{ARec}(p, record_i) \rightarrow \mathbf{P}[\mathtt{READ}(p, record_i)]$ means: every patient is permitted to read his medical record; $\mathrm{APurpose}(u, emergency, p) \wedge \mathrm{AR}(u, Physician) \wedge \mathrm{AR}(s, system) \wedge \mathrm{ARec}(p, d) \rightarrow \mathbf{P}[\mathtt{READ}(u, d)] \wedge \mathbf{O}[\mathtt{APPEND}_{\mathtt{f}}(s, ConnectionData(u, emergency, p, r), AuditFile)]$ signifies that the user who plays the role *physician* and declares the purpose *emergency* can read the patient record. Simultaneously, the system saves connection data; the security rule $\mathbf{P}[\mathtt{PROVIDECARE}(u, \_)] \rightarrow \mathrm{AR}(u, ClinicalStaff)$ indicates that the action "provide care" is only permitted to the clinical staff (necessary condition). So, the sufficient condition requires that the user play the role "clinical staff" in the team where the patient is treated. The rule is: $\mathrm{AURT}(u, clinicalStaff, T) \wedge \mathrm{AWard}(t, go, \_) \wedge \mathrm{AObj}(p, go) \rightarrow \mathbf{P}[\mathtt{PROVIDECARE}(u, p)]$.

6. CONCLUSION

This paper describes a security model that covers the particularities of HCCS. It also achieves a good trade-off between respect of the least privilege principle and flexibility of the access control. This logical model can be exploited to verify the consistency of the security policy.

7. ACKNOWLEDGMENT

This work was supported in part and is experimented in the RNRT project "MP6" where partners are: Ernst & Young Audit, ENST-Bretagne, ETIAM, France Telecom R&D, LAAS-CNRS, MasterSecuritY, ONERA-DTIM, Supélec-Rennes, and UPS-IRIT. We thank particularly Philippe Balbiani from IRIT for his contribution to the logic section.

8. REFERENCES

[1] Resolution A/RES/45/95 of the General Assembly of United Nations: *"Guidelines for the Regulation of Computerized Data Files"*, 14 December 1990.

[2] Recommendations R(81) of the Council of Europe: *"on Automated Medical Data Banks"*, Strasbourg, January 23, 1981.
[3] Directive 95/46/CE of the European Parliament and the Council of the European Union: "*On the protection of individuals"*, October 24, 1995.
[4] Directive 97/66/CE of the European Parliament and of the Council of 15 on: *"the treatment of the personal data and privacy protection on the telecommunications sector"*, December 15, 1997, Official Journal L 24, 30-1-1998, p. 1-8.
[5] Directive 2002/58/EC of the European Parliament on: *"the processing of personal data and the protection of privacy in the electronic communications sector"*, July 12, 2002, Official Journal L 201, 31-7-2002, p. 37-47.
[6] D. Powell and R. Stroud (Eds.), *Malicious and Accidental-Fault Tolerance in Internet Applications: conceptual model and architecture*, MAFTIA project Deliverable D2, November 2001, <http://www.research.ec.org/maftia/deliverables/index.html>
[8] B. Woodward, "The computer-based patient record and confidentiality", *New England Journal of Medicine*, v 333 N° 21, 1995, pp 1419-1422.
[9] Audit Commission, *Ghost in the Machine - An Analysis of IT Fraud and Abuse*, Audit Commission Publications, United Kingdom, ISBN 1-86240-05603, 1998.
[10] R. Anderson, *Personal Medical Information Security, Engineering, and Ethics*, Personal Information Workshop proceedings, Cambridge, UK, June 21-22, 1996, Springer ISBN 3-540-63244-1.
[11] S.I. Gavrila, J.F. Barkley "Formal Specification for Role-Based Access Control", *Third ACM Workshop on Role-Based Access Control*, Fairfax, VA, USA. 22-23, October 1998.
[12] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn and R. Chandramouli "A Proposed Standard for Role-Based Access Control", *ACM Transactions on Information and System Security,* Volume 4, Number 3, August 2001.
[13] Roshan K.Thomas, "TMAC: A primitive for Applying RBAC in collaborative environment", *2nd ACM, Workshop on RBAC*, Fairfax, Virginia, USA, November 1997.
[14] Christos K. Georgiadis, Loannis Mavridis *et al.*, "Flexible Team-Based Access Control Using Contexts", *ACM Symposium on Access Control Models and Technologies (SACMAT'01)*, Chantilly, Virginia, USA, May 3-4, 2001, pp. 21-27.
[15] M. Willikens, S. Feriti, M. Masera, "A Context-related authorisation and access control method based on RBAC", *ACM Symposium on Access Control Models and Technologies (SACMAT'02)*, California, USA, June 3-4, 2002, pp. 117-124.
[16] P. Bonatti, E. Damiani, S. di Vimercati, P. Samarati, "An access Control Model for data archives", *16th IFIP TC11 International Conference on Information Security (IFIP/Sec'01*), Paris, France, June 11-13, 2001, pp. 261-276.
[17] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, "A Fine-Grained Access Control System for XML Documents", in *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, n. 2, May 2002, pp. 169-202.
[18] B.F. Chellas, *Modal Logic: An Introduction*, 295p., Cambridge University Press, 1980, ISBN 0-521-29515-7.
[19] Y. Deswarte, N. Abghour, V. Nicomette, D. Powell, "An Intrusion-Tolerant Authorization Scheme for Internet Applications", in *Sup. of the Proceedings of the 2002 International Conference on Dependable Systems and Networks (DSN2002),* Washington, D.C. (USA), 23-26 June 2002, pp. C-1.1 - C-1.6.
[20] A.K. Dey, G.D. Abowd (1999). *Towards a better understanding of context and context-awareness*. GVU Technical Report GITGVU-99-22, College of Computing, Georgia Institute of Technology.
[21] M. Covington, W. Long, S. Srinivasan, A.K. Dey, M. Ahamad, G.D. Abowd, "Securing Context-Aware Applications Using Environment Roles", *ACM Symposium on Access Control Models and Technologies (SACMAT'01)*, Chantilly (Va), USA, May 3-4, 2001, pp.10-20.
[22] E. Bertino, C. Bettini, E. Ferrari, P. Samarati, "A Temporal access control mechanism for database systems", *IEEE Transactions on Knowledge and Data Engineering*, v8, 1996.